TR-289

Analysis of Design Specification in a Communication System by Means of Petri Nets

by
H. Hasegawa, W. Tanaka and K. Shibata
(Oki)

August, 1987

ANALYSIS OF DESIGN SPECIFICATION IN A COMMUNICATION SYSTEM BY MEANS OF PETRI NETS

Haruo HASGAWA, Wataru TANAKA and Kenji SHIBATA

Telecommunication Division
Oki Electric Industry Co., Ltd.
4-11-22, Shibaura, Minato-ku, Tokyo 108, Japan

ABSTRACT

We are developing an expert system -EXPRESS (EXPeRt system for ESS). EXPRESS designs automatically the specification for a communication system from users' requirements. Each requirement which means one of service features is converted into PSG (Partial Service Graph), and all PSGs are integrated into TSG (Total Service Graph). TSG is the final specification and the most fundamental document in programming communication software. PSG and TSG are described in Petri Net. They may have some errors or inconsistencies, which must be detected and resolved. The verification of the specification consists of syntactic part and semantic part. The syntax is verified in the process of designing the specification by utilizing an algebraic analysis technique of Petri Net. Besides, the analysis is applied to the integration of PSGs and discovery of new service features.

1. Introduction

In a communication system, user's requirements have become various and diversified recently, and it takes many hours of engineers and programmers to design an advanced communication system. A lot of manpower of high-class engineers are needed especially at the specification phase. We have proposed an expert system -EXPRESS (Expert system for an electronic switching system), which designs automatically the specification from users' requirements written in a natural language. In EXPRESS the specification is described in Petri Net and is verified syntactically by utilizing an algebraic method of Petri Net. This paper gives an outline of EXPRESS and describes an analysis of the specification.

2. Specification language for a communication system

SDL (Specification and Description Language) is recommended as a language for a communication system by CCITT (Comité Consultatif International Télegraphique et Téléphonique) and is used in many countries. SDL is based on specification languages which have been already used in a switching system. And now specification description for a communication system is reviewed from another point of view. In other words, though SDL plays the leading role in specification languages, a theoretical analysis like verification or simulation will be needed. Fromthis point, other languages become a matter of some concern.

Petri Net is known as a method of modeling and analyzing an asynchronous and parallel system. It has been used as knowledge representation and applied to logic programming recently, too. This is why we choose Petri Net as a specification language for a communication system.

3. Outline of EXPRESS

Fig.1 shows a system configuration of EXPRESS. The system consists of specification acquisition subsystem, knowledge base and specification verification subsystem.

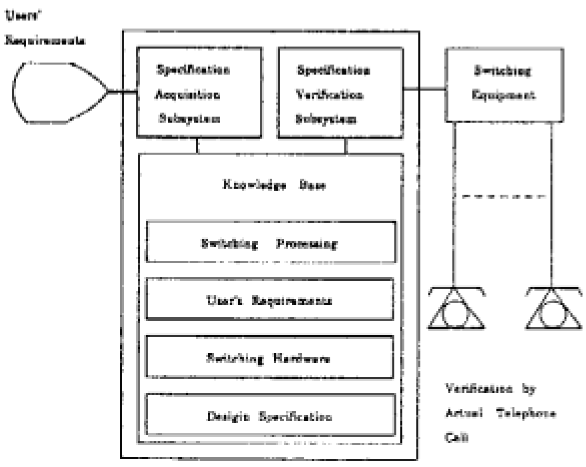


Fig.1 Configuration of the system

The function of them are as follows.

(1) Specification acquisition subsystem

When users input ambiguous requirements about service features of a switching system in a natural language, this subsystem understands the requirements by using the knowledge base. Each of users' requirements is usually fragmentary and there is a possibility that they have mutual inconsistencies. When users input the requirements which conflict with the knowledge base, this subsystem resolves them by interacting with users. It designs the consistent specification from users' requirements which may be ambiguous and mutually inconsistent.

(2) Knowledge base

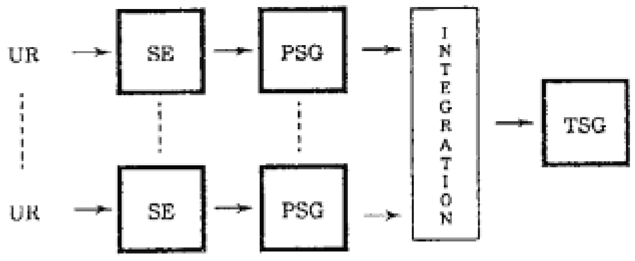
The knowledge base is stored with the specification which was designed and the knowledge necessary to design the specification. There is knowledge about switching processing, users' requirements, switching hardware and design specification.

(3) Specification verification subsystem

This subsystem works an interpreter according to the specifications with which the knowledge base are stored and controlls a switching equipment. This subsystem can verify semantically whether the specification satisfies users' requirements completely or not.

4. Integration of service graphs

Fig.2 shows the process in which EXPRESS designs the specification which doesn't have any inconsistency.



UR : User's Requirement
SE : Service Element
PSG : Partial Service Graph
TSG : Total Service Graph

Fig.2 The process of designing the specification

Each product of the process is explained below.

(1) SE (Service Element)

One of users' requirements is converted to SE at first. SE describes actions of terminals and states which change by the actions, from the beginning to the end of a service feature. Fig.3 shows an example of users' requirements and SE about station-to-station dialing in PBX (Private Branch eXchange).

1. A caller goes off-hook.
   Then he hears a dial tone.

2. He dials the number.
   Then he hears a ring back tone and a called party receives a ringing signal.

⋮

(a) Users' requirement (in a natural language)

```
1    ([tr(offhook,[agent(caller)])],
      [st(hear,[goal(caller),object(dt)])])

2.   ([tr(dial,[agent(caller),object(number)])],
      [st(hear,[goal(caller),object(rbt)]),
       st(receive,[goal(called),object(rgt)])])
```

⋮

(b) SE(service element)

Fig.3 An example of users' requirements and SE (station-to-station dialing)

(2) PSG (Partial Service Graph)

SE is converted to PSG which is described in Petri Net. In PSG an action is expressed as a transition, and a state before the action and a state after the action are expressed as a set of input places and as a set of output places respectively. EXPRESS can not only represent states but also make clear changing of states by moving tokens and firing them.

Fig.4(a), Fig.4(b) and Fig.4(c) show a case of the standard call, a case when the calling party dials a busy number and a case when the caller hangs up while the called station instrument is rung respectively.

(3) TSG (Total Service Graph)

All PSGs each of which is fragmentary are integrated into TSG, and TSG is a final design specification. Fig.4(d) shows TSG which is created from 3 PSGs. This example explains only about a station-to-station dialing service, but EXPRESS usually integrates all PSGs which mean many service features in PBX, including outgoing call, incoming call, call forwarding etc.

Fig.5 shows the internal expression of TSG. Each state corresponds to a place and the expression of TSG clarifies transitions which can be connected to a place and places which can be connected to a transition.

PSGs are integrated into TSG as follows.
(1) unify one PSG with TSG
(2) add the PSG to TSG if unification is not completed

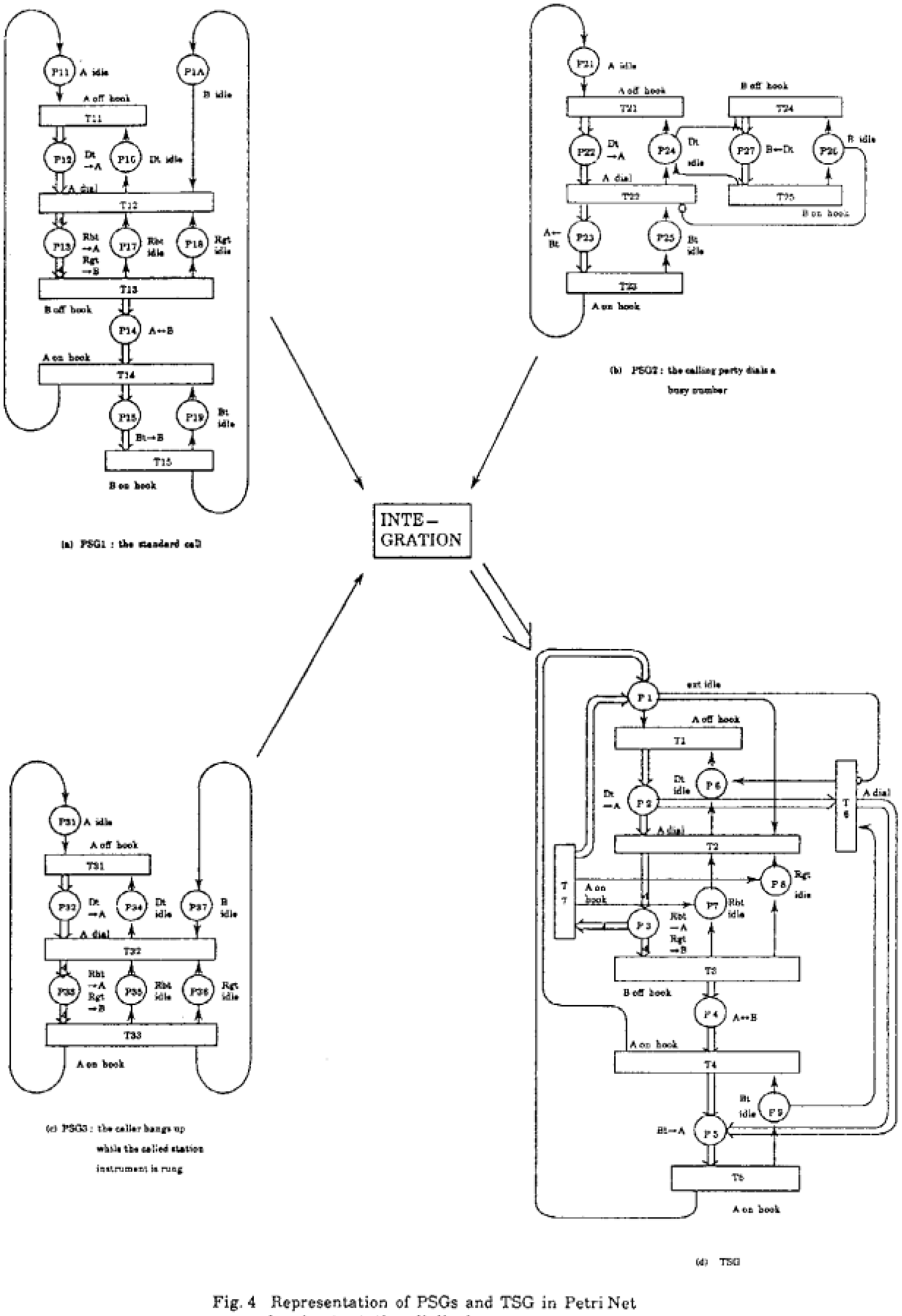


Fig. 4 Representation of PSGs and TSG in Petri Net
(station-to-station dialing)

(3) do nothing if unification is completed

```
place(Pt1, gpl(rel(token(Ttk1, extension),
                                    idle)),
             spt([Tt4, Tt5, Tt7]),
             snt(Tt1, Tt2])).

transition(Tt1, Arc1, offhook, transit
                            ([Arc1, Arc2]),
             spp([Pt1, Pt6]),
             snp([Pt2])).
```

Fig.5 An example of TSG expression

5. Matrix of Petri Net

The matrix of Petri Net is described below. We define D^- and D^+ which represent an input function and an output function of transitions respectively. Each matrix consists of m rows which correspond to transitions and n columns which correspond to places. An element of j-th row and i-th column of D^- means a multiple degree of arcs which go from i-th place to j-th transition, and an element of j-th row and i-th column of D^+ means a multiple degree of arcs which go from j-th transition to i-th place. When n-dimension vector μ, which is called marking, represents the number of tokens which exist in each place, the j-th transition Tj fires if the following expression is realized.

$$\mu \geq e[j] \cdot D^- \text{ .... (A)}$$

e[j] is m-dimension vector in which the j-th element is 1 and other elements are all 0. When Tj fires in the marking μ_0, the new marking becomes the following.

$$\mu = \mu_0 + e[j] \cdot D \text{ ..... (B)}$$
$$D = D^+ - D^-$$

Therefore, when ignition series Tj_1, Tj_2 ..... Tj_k fire, the new marking becomes the following.

$$\begin{aligned} \mu &= \mu_0 + (e[j_1]+e[j_2]+ \text{.....} e[j_k]) \cdot D \\ &= \mu_0 + f(\sigma) \cdot D \text{ ...... (C)} \end{aligned}$$

6. Detection of syntactic inconsistencies

The specification must not have any inconsistency. There are two kinds of inconsistencies; syntactic one and semantic one. But, there is a possibility that the specification may include some inconsistencies, because users' requirements have already included some inconsistencies or they can be made in the process of designing the specification. EXPRESS can verify whether the specification satisfies users' requirements by working a switching system or not, but when inconsistencies are found at the phase, it takes a lot of manpower to correct them. Therefore, it is desirable to detect and resolve the inconsistencies in the process of designing SE, PSG or TSG.

Detection of inconsistencies consists of syntactic verification and semantic one. Syntactic inconsistency and semantic one are detected respectively. Semantic verification means verifying whether users' requirements are realized correctly or not, and it needs knowledge about a switching equipment, design specification etc. The specification can be verified semantically by means of knowledge in the computer, but it is verified mainly by a prototyping method.

On the other hand, syntactic verification does not need working of a switching system and can be made by an analytic method.

6.1 Syntactic inconsistencies

There are two cases in which TSG has inconsistencies. Some inconsistencies are what PSG has itself, and others are what are created in the process of integrating PSGs into TSG.

PSG has itself inconsistencies in the following case.

(1) After tokens pass all or some transitions, the initial state cannot be obtained again including the number of tokens in each place. This inconsistency which exists in PSG is succeeded to by TSG, after PSGs are integrated into TSG.

PSGs have mutual inconsistencies in the following case.

(2) A transition and a set of input places in one PSG correspond with those of another PSG, and a set of output places in two PSGs are different from each other.

(3) One PSG explains a users' requirement in detail and the other PSG doesn't. It is necessary to resolve inconsistencies before TSG is completed.

6.2 Detection of inconsistencies by means of Petri Nets

EXPRESS verifies the design specification by analyzing inconsistencies described in 6.1 by means of a matrix of Petri Net in the process of designing TSG. At first (1) can be verified by utilizing the expression (C). For example, the matrix D of TSG in Fig.4 is as follows.

$$D = D^+ - D^-$$
$$= \begin{pmatrix} -1 & 2 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & -2 & 4 & 0 & 0 & 1 & -1 & -1 & 0 \\ 0 & 0 & -4 & 2 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & -2 & 2 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 1 \\ 0 & -2 & 0 & 0 & 2 & 1 & 0 & 0 & -1 \\ 2 & 0 & -4 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

The next expression shows that the state returns to the initial state and the marking doesn't change when T1, T2 and T7 fire in sequence.

$$\begin{aligned}\mu &= \mu_0 + (1\ 1\ 0\ 0\ 0\ 0\ 1) \cdot D \\ &= \mu_0 + (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0) \\ &= \mu_0 \\ &= (2\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1)\end{aligned}$$

7. Integration by utilizing a matrix

EXPRESS integrates PSGs into TSG by unifying each paragraph of PSG with TSG. The integration can be realized by unifying each row and each column of PSG matrix with that of TSG matrix. D^- and D^+ of TSG are completed by adding places and transitions which don't coincide with those of TSG. For example, Fig.6 shows the process of making D^- of TSG in Fig.4. In this example, P11 and P1A in PSG1 are unified as the same kind of terminals.

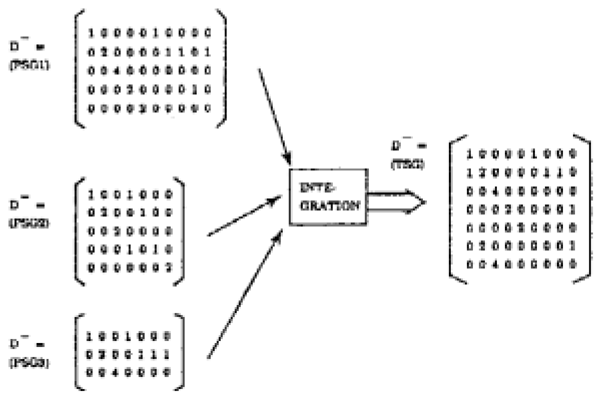


Fig.6 An example of integration of matrix D-

8. Presentation of new service

The system proposes a new route from the Petri Net graph of the design specification. The system finds fireable transitions and make them fire on the condition that all resources like terminals and trunks are idle. And then the system can discover new service features which users didn't think about. Fig.7 shows the process. In Fig.7 (a) and (b) are integrated into (c), when $P_2=P_5$. Then two new services are created:
$P_1 \rightarrow T_1 \rightarrow P_2 \rightarrow T_5 \rightarrow P_6$, $P_4 \rightarrow T_4 \rightarrow P_2 \rightarrow T_2 \rightarrow P_3$

Besides, the system can find a new service feature by adding a new transition and connecting places without adding a place. This is completed by utilizing knowledge. Fig. 8 shows the example.

A station can hang up while he is hearing a dial tone. $(P_1 \rightarrow T_1 \rightarrow P_4)$
A station can hang up while he is hearing a ring-back tone. $(P_2 \rightarrow T_2 \rightarrow P_4)$
↓
A station can hang up while he is hearing a busy tone. $(P_3 \rightarrow T_3 \rightarrow P_4)$

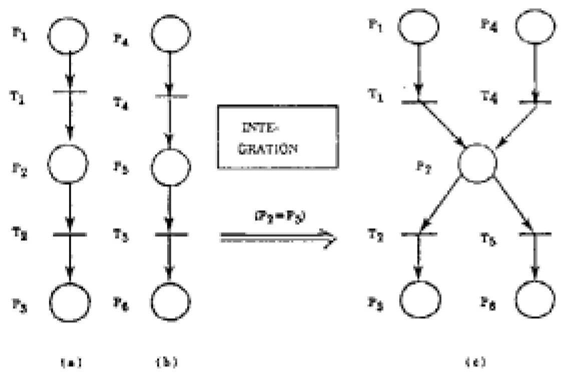


Fig.7 Representation of a new service (ex.1)

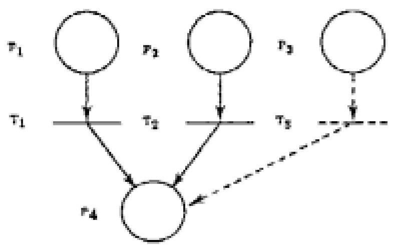


Fig.8 Representation of a new service (ex.2)

In this case, the system informs users of a new service at CRT etc. It depends on users whether the new service can be allowed or not.

9. Conclusions

In this paper we have analyzed the design specification by utilizing Petri Net matrix and applied it to detection of inconsistencies, integration of PSGs into TSG and discovery of new service features. It is very important to detect and resolve various inconsistencies in the process of designing the specification from users' requirements, and this is one of methods of analyzing them. Moreover, we are going to investigate an introduction of a colored token to the system and a problem that the matrix doesn't decide the sequence of ignitions and doesn't have every information about Petri Net.

ACKNOWLEDGMENTS

This research is supported by ICOT (Institute for New Generation Computer Technology). We would like to thank Ph.D. Hidenori Itoh for many helpful suggestions on this paper.

REFERENCES

[1] J.L.Peterson: "Petri Net theory and the Modeling of Systems", Prentice-Hall (1981)

[2] T.Murata and D.Zhang: "A High-Petri Net Model for Parallel Interpretation of Logic Programs", Proceedings of the IEEE Computer Society 1986 International Conference on Computer Languages (1986), pp.123-132

[3] CCITT Recommendation: "Functional Specification and Description Language (SDL)", CCITT REDBOOK (1985)

[4] H.Hasegawa, K.Shibata and S.Yuyama: "A Study on Synthesis of Design Specifications in a Communication System", Proceedings of the 33rd Annual Convention IPS Japan, 5T-9(1986) [in Japanese]