計測自動制御学会東北支部第231回研究集会(2006. 10. 31)
資料番号 231-13

ペトリネットによるマルチOSネットワークシステムの構築

Construction of a Network System by Petri Nets

○有我祐一，津藤洋明，渡部慶二，村松鋭一，遠藤茂

○ Y. Ariga, H. Tsutou, K. Watanabe, E. Muramatsu and S. Endo

山形大学

Yamagata University

キーワード：ペトリネット(Petri Nets), ネットワークシステム(Network System), Java,
状態遷移シミュレータ(State transition simulator), リアルタイム制御(Realtime control)

連絡先：〒992-8510　山形県米沢市城南 4-3-16 山形大学 工学部 応用生命システム工学科
有我祐一, Tel & Fax: (0238)26-3764, E-mail: y_ariga@yz.yamagata-u.ac.jp

1　はじめに

近年，コンピュータにより構成された制御システムは様々な分野で用いられ，同時に利用者の要求に答えてシステムは日々高度化・肥大化している．その一方で，制御システムの構築や保守に必要な労力と時間の削減も望まれている．

一般的に，制御システムの動作の記述にはCやC++などのプログラミング言語が用いられている．このため，プログラミングの初心者にとっては，簡単な制御システムの構築でも苦労を強いられてしまう．また，制御内容が複雑になるほど，その内容を記述したプログラムが長く複雑なものになるため，プログラミングの熟練者にとっても相当の苦労を強いられる．さらに，プログラムの複雑化は，エラーが発生した場合に修正箇所を見つけ出すのに必要な時間が長くなるという問題点を引き起こす．どんなシステムでも潜在的なエラーを持っており，システム構築直後はもちろんのこと，運用時にもそのエラーが顕在化し運用停止となることが度々起こる．復旧させるまでの時間を可能な限り短縮するためには，保守と改変が容易であることが望まれる．

上記の問題に対処するためには，以下の3点の解決が求められる．

- 初心者でも容易に制御内容の記述が可能
- 視覚的に制御内容の理解が可能
- 制御内容の保守・改変が容易

これらの解決法の1つとして，本研究ではペトリネットを用いたコンピュータネットワークシステムを構築することを目的とする．ペトリネットは，離散事象システムの表現に適したグラフィカルモデルであり，システムの状態遷移が分かりやすいという特徴を持っている[(1)~(4)]．このため，初心者でも記述内容を容易に理解して記述することができる．また，視覚的に記述内容を理解できるためにエラー箇所の発見と保守・改変が容易になる．よって，ペトリネットの採用により上記の問題点の解決が可能となる．

本研究で構築したネットワークペトリネット環境をNetwork Petri-net for Java (以下Npj)と名づけた．システムの開発は，ネットワークの記述が容易なJava言語を選択した．構築したペトリネット制御システムは，

1)ペトリネットを記述・編集するネットエディタ
2)システムの状態遷移を確認するシミュレータ
3)機械システムのリアルタイム制御機能

以上の3つの機能を備えている．また，通常のペトリネットを拡張して，ネットの階層化やネットワーク機能，さらに独自の要素としてカラープレース機能の導入を行い，ペトリネットの表現の幅をより広いものとした．

本論文では，Npjの機能を紹介し，さらに応用例として遠隔制御システムを用いた実験の結果を示す．

2　Npjの概要

Npjは，Java言語で作成したためOSに依存せず，JavaVMをインストールしてあればどのようなOS上でもプログラムを動作させることができる．

Fig.1にNpjの起動時の画面を示す．起動時にはエディタとなるパネルが1つ生成される．Npjの動作やネッ

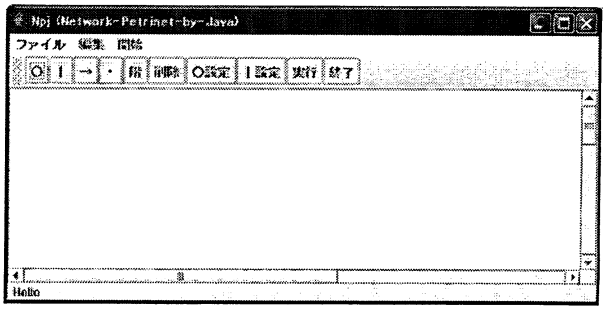

Fig. 1: An opening screen of Npj

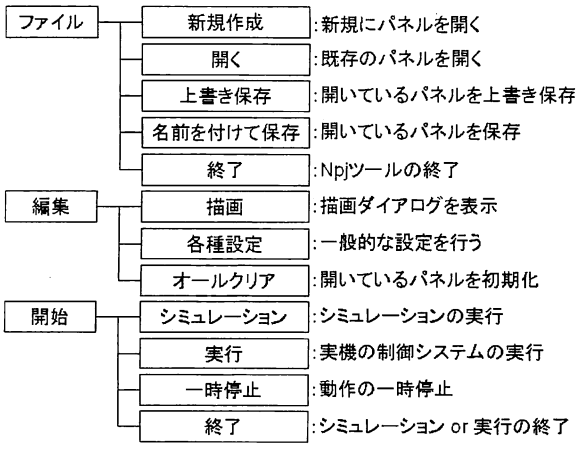

Fig. 2: Menu constitution of Npj

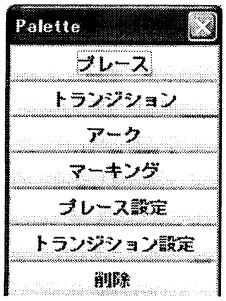

Fig. 3: A dialogue for drawing

トの描画・設定に必要な機能は，パネル内にあるメニューとタスクバー，およびダイアログボックスに用意されている．使用者はこれらを用いてパネル内の作業領域にネットを記述していく．

Npjのメニュー構成はFig.2のようになっており，ネットの記述からシミュレーションおよび実機の制御までの一連の機能を1つにまとめている．

以下，各機能について説明する．

2.1 エディタの基本機能

エディタは，ネットの描画および変更・修正が容易になるように設計されており，描画に必要な機能はパネル内のタスクバーと，Fig.3に示すダイアログボックスに用意されている．使用者はタスクバーかダイアログボックスで描きたいネット構成要素を選択し，パネル内で左クリックすることによって，クリックした場所に選択したネット構成要素を描画することができる．

プレースとトランジションは，ダイアログボックスで選択したのちパネルの作業領域を左クリックすれと描画することができる．

アークは

- 始点と終点は必ずプレースもしくはトランジションでなければならない．
- プレースを始点とした場合はトランジションを，トランジションを始点とした場合はプレースを，それぞれ終点として選択しなければならない．

という条件を満たさなければならない．そこで，始点と終点となる要素を選ぶと自動的にアークが描画されるようにした．このとき，始点にプレースを選択した場合には他のプレースを，始点にトランジションを選

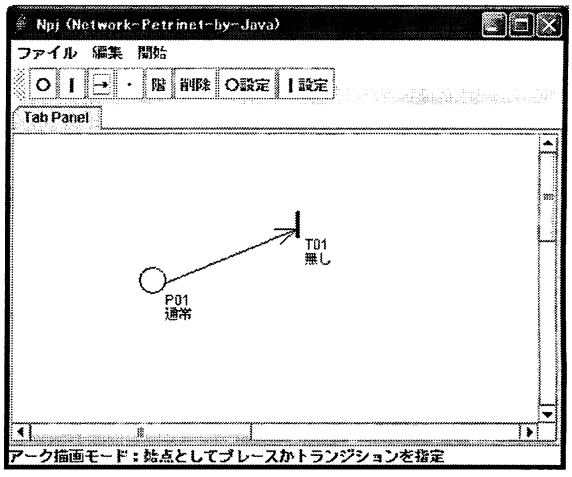

Fig. 4: An example of drawing

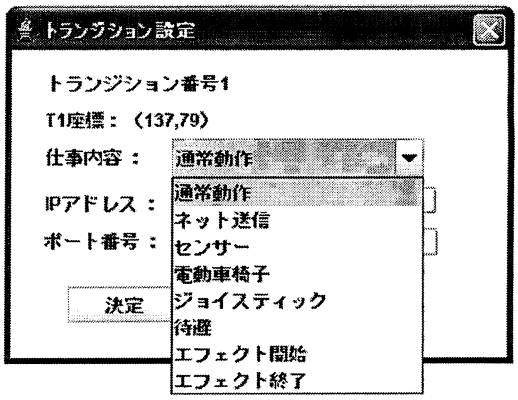

Fig. 5: A setting dialogue of transition

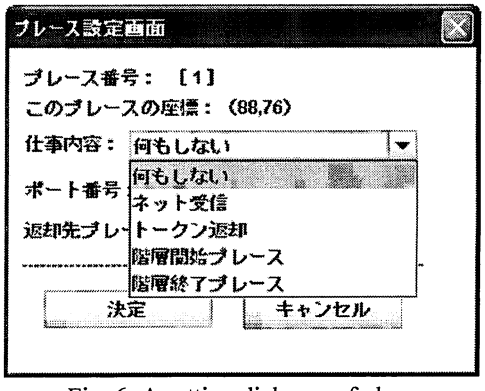

Fig. 6: A setting dialogue of place

択した場合には他のトランジションを終点として選択できないように自動化した．また，アークは後から任意の折れ線に編集可能となっているため，描画が進みネットが複雑になった場合でも再配置により見やすくすることが可能である．Fig.4 に描画例を示す．

描画後には各構成要素の定義を行う．トランジションの設定を行う場合は，描画ダイアログの「トランジション設定」を選択した状態で作業領域に描画されたトランジションをクリックする．すると Fig.5 のようなダイアログが表示されるので，あらかじめ作成しておいた仕事内容のリストから選択する．これで，発火の際に行う仕事をトランジションに割り当てることができる．

同様の方法でプレースでも設定画面(Fig.6)を呼び出

し，ネットワーク機能や後述するカラープレース，階層型ペトリネットのための開始・終了プレースとしての機能などを割り当てる．

2.2 カラープレース

従来のペトリネットではシステムの分岐を描くことが困難である．その解決手段として，本研究ではカラープレースを提案する．

カラープレースはプレースに色の概念を加えたもので，このプレースの色とトランジションの出力するトークンの色を対応させることにより分岐の概念を表現しやすいペトリネットとすることができる．

Fig.7 にカラープレースを使った例を示す．内容物の重さを量れるセンサーがついた 1 つの容器があるとし，この容器に水を注ぎ一定の重さになったら注水を停止するシステムを考える．図中の各要素の見出しには，トランジションに割り当てた仕事，もしくはプレースの色を括弧書きで表している．トランジション 1 が発火するとセンサーで重さ情報を取得し，その値によって出力するトークンの色を決定し出力する．プレース 2 と 3 は各々トークンを受け取るが，自身に設定されている色とトークンの色が異なる場合には，トークンを破棄する．Fig.7 の例では，赤色のトークンが出力された場合には停止，黒色のトークンが出力されたら注水を続行する．

このように，カラーペトリネットを導入することで，条件判断などの分岐を容易に表現することが可能となる．

2.3 階層型ペトリネット

一般的に，システムが複雑になるに従い，そのシステム表現も複雑化する．ペトリネットも例外ではなく，その特徴の 1 つであったシステム状態の把握の容易さが失われ，ネット管理も複雑になる．この問題点の解決方法の 1 つとして，Fig.8 に示すようなネットの階層化が挙げられる．本研究では，階層化するネットの一部を 1 つのトランジションにまとめることにより，表面上分かりやすい表現を可能とした．

階層を構成するために，次の 2 点を定義した．

- 1 つのトランジションにつき，1 つの階層パネルを持つ．
- 階層下のパネルの始点と終点はプレースとする．

1 つ目の定義により，同一のトランジションに対して階層化とカラーペトリネットを同時に定義できないことになる．また，2 つ目の定義はペトリネットの定義に基づき導入されたもので，下の階層パネルでは，ネットの開始は階層開始プレース，終了は終了プレースを設定しなければならない．

トランジションを階層モードに設定するためには，タスクバーにある「階」ボタンを押してからトランジションを左クリックする．これにより，下の階層となるパネルを呼び出し，通常のパネルと同様にネットを記述できる(Fig.9).

2.4 ネットワークペトリネット

Npjで実現したネットワーク機能をFig.10に示す．こ

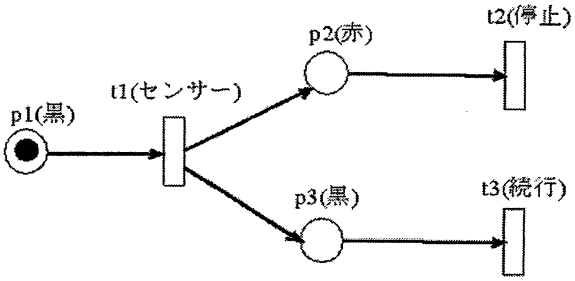

Fig. 7: A use example of color place

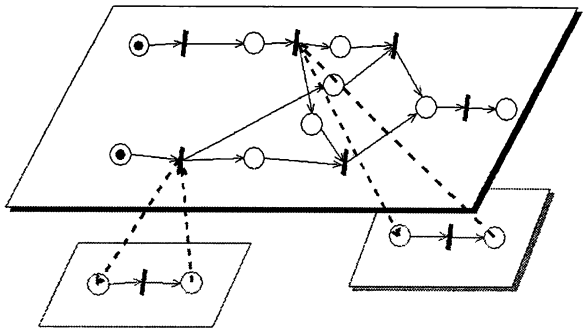

Fig. 8: A Hierarchical structure

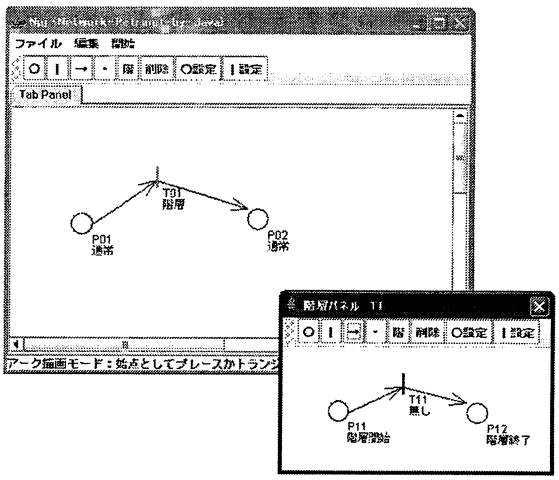

Fig. 9: Setting example of a hierarchy strucure

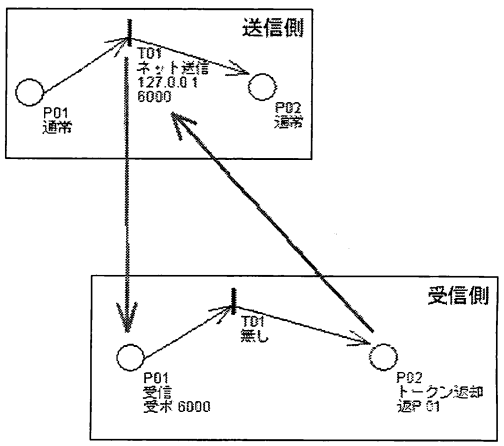

Fig. 10: Setting example of networking

の機能は，TCP/IPを用いた通信により，複数台のPCや機械システムを1つの制御システム上で制御することを可能にする．Npj上の通信の定義として，トランジションは送信のみ，プレースは受信のみとして機能を与えた．

ネットワーク機能を使用するためには，2.1で説明した方法でトランジションとプレースにネットワークの設定をする必要がある．

送信側となるトランジションは，Fig.5の設定画面で「ネット送信」を選択し，さらに送信先のIPアドレスとポート番号を入力することで設定が完了する．設定が完了すると，トランジション番号の下にIPアドレスとポート番号が表示される．

受信側のネットには，2 つのプレースを設定する必要がある．1 つは送信されてきたトークンを受け取るプレース，もう 1 つはトークンを送信側に返却するためのプレースである．トークンを受け取るプレースは，Fig.6 の設定画面で「ネット受信」を選択し，受け取るポート番号を入力することで設定され，プレース番号の下にポート番号が表示される．また，トークンの返却するためのプレースは，「トークン返却」を選択した後，送信先とつながっている受信用プレース番号を指定することで設定が完了する．このとき，プレース番号の下に返却先となるプレース番号が表示される．Fig.10 に示した例では，No.01 のプレースが受信用プレースとなるため，返却先として「P01」と表示されている．

2.5シミュレータ機能

記述したネットの成立性を検証するための機能が，シミュレータ機能である．ペトリネットはシステムの状態遷移を視覚的に表現可能であることから，シミュレーションにおいてもこれを実現しなければならない．本研究では任意に設定した時間間隔でトークンがネットを移動する様子を表示することで，視覚的に分かりやすい表現でネットの成立性を検証することを可能にした．

また，実機械の制御する機能を有したNpjでは，シミュレータに時間の概念を取り込む必要がある．一般的なペトリネットの考え方では，トランジションに割り当てた仕事にかかる時間を無視するか，全ての仕事は同時に終わると考え，トランジションの発火タイミングは全て一緒と考える．この場合，全てのトランジションが同じタイミングで，

1. 発火可能かチェックし，可能なら発火する．
2. 一定時間後に発火を終了し，出力プレースにトークンを渡す．

以上の処理を繰り返していれば，シミュレーションが成立する．しかし，本研究のように，トランジションに割り当てる仕事を「ネット送信」や「実機械の制御」などとした場合，処理時間が必要となり，かつ処理が完了するまでの時間は一定ではない．このため全てのトランジションの発火タイミングが同じとは考えられない．そこでNpjでは，発火している全てのトランジションにスレッドを持たせ，トランジションの仕事が終了するまでスレッドが動作するようプログラムを構成することで，シミュレーションと実機の制御との整合性をとった．

2.6リアルタイム制御機能

パネルにあるタスクバーより「実行」を選択することで，実機のリアルタイム制御を実行することができる．この機能については具体的な例を示しながら次章で述べる．

3　遠隔制御システムへの応用

3.1 システムの概要

構築した Npj を遠隔制御システムへ応用した．Fig.11 に今回 Npj を適用した電動車椅子の遠隔制御システムの構成を示す．2 台のノート PC を使用しており，USB カメラで撮影しているお互いの映像とヘッドセットに

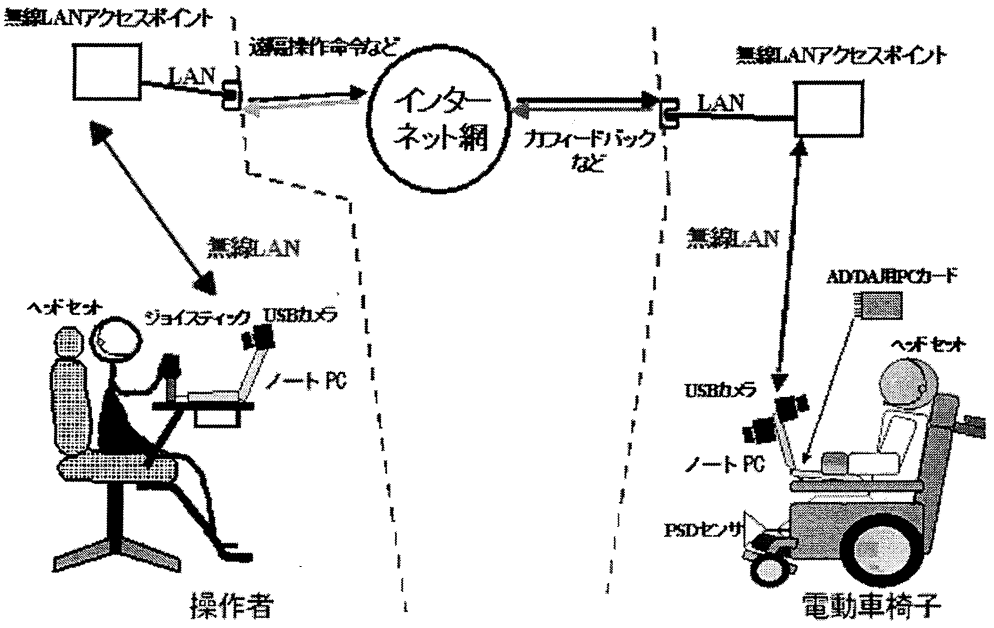

Fig. 11: Construct of a remote control system

よる音声通信でコミュニケーションしながら，インターネットを介した遠隔制御を行う．操作者は電動車椅子に搭載された USB カメラの前方映像を見ながらジョイスティックにより操作する．

操作者側のノート PC にはフォースフィードバック機能がついたジョイスティックが接続されている．ノート PC からフィードバック機能を制御することにより，ジョイスティックを持つ操作者に情報を伝えることができる．今回のシステムでは，電動車椅子からの任意の信号を操作側 PC が受信した場合にジョイスティックを制御するように，ローカルな制御系を構築した．

一方の電動車椅子側のノート PC には AD/DA 変換器を有した PC カードを搭載してあり，センサ信号の取り込みと車椅子のモータの制御のための信号出力ができるようになっている．車椅子側ノート PC には，車椅子のモータを制御するようにローカルな制御系が構築されている．通常は，操作側からの指令に基づき車椅子が動く．しかし，操作者は車椅子の前方 USB カメラの画像を見て操作をしているため，カメラに映らない障害物などがあった場合には操作者が認知できずに衝突してしまう可能性がある．そこで，前方に障害物検知のための PSD センサを設置し，障害物を検知した場合には車椅子を停止して退避動作(後退)するようにローカルな制御系を構築した．さらに，障害物を検知した場合，操作者側の PC へジョイスティックのフォースフィードバック機能を作動させるように信号を送信し，避動作が終了したらフォースフィードバック機能の停止の信号を送信する機能を持たせた．

3.2 モデル化

電動車椅子の遠隔制御システムをペトリネットでモデル化する。今回はカメラ画像と音声通信を除いた遠隔操作システムの中核部分のみをモデル化した。

まず，操作者側の事象を次のように定義した．

1. ジョイスティックの位置情報を取得する．
2. 電動車椅子側に座標情報を送信する．
3. 車椅子側からの情報に基づいてジョイスティックのフォースフィードバック機能を動作させる．
4. 車椅子側からの情報に基づいてジョイスティックのフォースフィードバックを停止する．

ジョイスティックの位置情報は 16bit の分解能を持つデータで，スティックが直立している位置を中心として±32735 の幅を持つ数値としてノート PC に取得される．車椅子にはその値をネットを介して送信する．また，車椅子側からフォースフィードバックに関する信号が送られてきた場合には，それに応じてフォースフィードバック機能の起動もしくは停止を実行する．以上の定義に基づいて操作者側のシステムのフローチャートを Fig.12 に示すように作成した．

次に，電動車椅子の事象を以下のように定義した．

1. 車椅子に搭載された PSD センサーの出力値を AD 変換器を介して取得する．
2. 電動車椅子を受信したジョイスティックの座標情報を元に操作する．

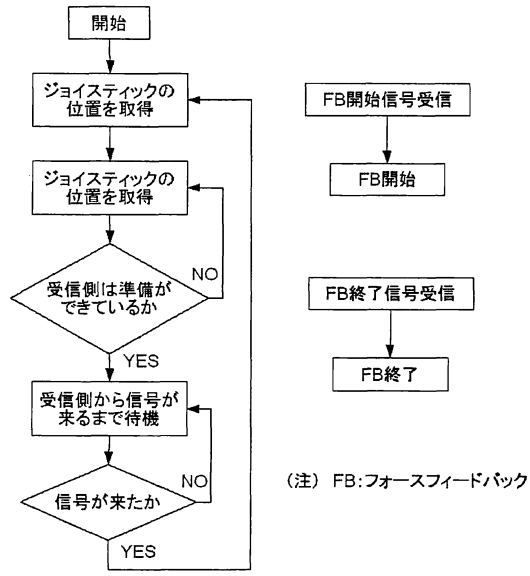

Fig. 12: Flow charts of the manipulator

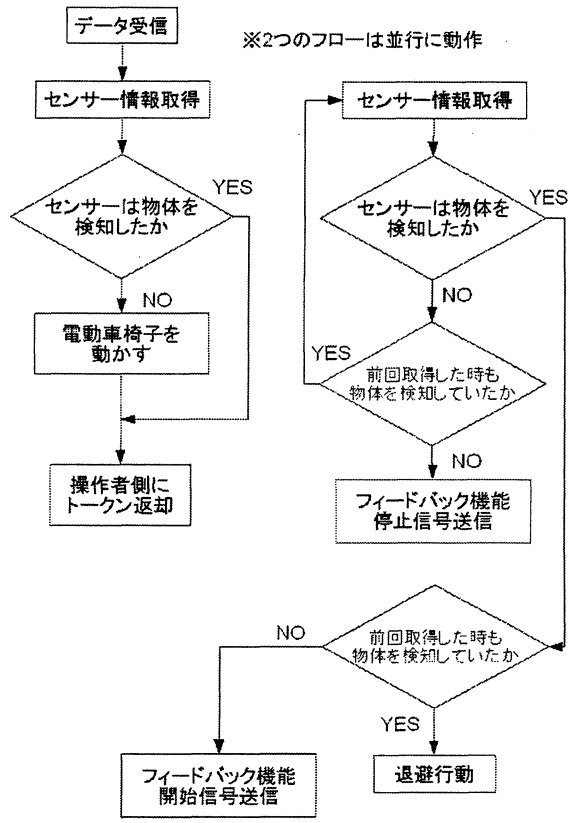

Fig. 13: : Flow charts of the electric wheelchair

3. 退避行動をとる．（車椅子を後退させる）
4. 操作者側にフォースフィードバック機能の動作開始命令を送信する．
5. 操作者側にフォースフィードバック機能の停止命令を送信する．

PSD センサの情報による動作は，その信頼性を高める

ために前回取得した結果も考慮して決定される．通常の動作中に前方に物体を検知した場合，まず送信側にフィードバック機能の開始を伝える．再度センサー情報を取得した際にも物体を検知した場合には，車椅子の退避行動を行う．退避行動を終了し，かつ送信側にフィードバック機能を停止する命令を送るには，2 回連続してセンサが物体検知しなかった場合のみに限られる．以上の事象の定義に基づいて，電動車椅子側のシステムのフローチャートを Fig.13 に示すように作成した．車椅子を駆動するフローとセンサー情報に基づいて動作を判断するフローは，並列に動作している点に注意が必要である．

3.3 ネット作成と実験

以上のように作成したフローチャートを基に，操作者側と電動車椅子側の双方のネットを作成した．Fig.14 の左側はジョイスティックの位置を検出してその情報を送信する部分である．制御の実行をやめないかぎり情報を送り続ける必要があるために，ネットはループしている．右側に位置するのは，それぞれフィードバック機能の開始と停止のためのネットである．Fig.15 の上段は，センサー情報の取得とその情報に基づいた動作の判断をするネットである．動作の判断は制御動作中は常に行う必要があるため，ループ構造になっている．下段は車椅子の動作をするためのネットである．

このネットを用いて実際に制御実験を行った結果，操作者はネットを介して電動車椅子を操作でき，障害物あっても衝突しないように車椅子が回避動作を行うことを確認できた．

4 おわりに

本研究では，初心者でも簡単に扱える制御システムの構築を目的として，ペトリネットを用いた制御システムを構築し Network Petri-net for Java (Npj)と名づけた．Npj は，カラープレースおよび階層型ペトリネットの機能を導入したことによりネット表現の幅がより広いものになった．さらに，ネットワーク機能を追加したことにより遠隔制御システムに応用させることが可能になり，その有用性は実験によって確認できた．

しかし，今回の実験対象はカラープレースと階層型ペトリネットを使うような複雑なシステムではなかったため，より複雑なシステムに適用して有用性を実証する必要がある．また，現在の Npj には可達性を判別する機能がないなど，初心者にとって真に扱いやすいシステムにはなっていないため，さらなる機能向上をすることも今後の課題である．

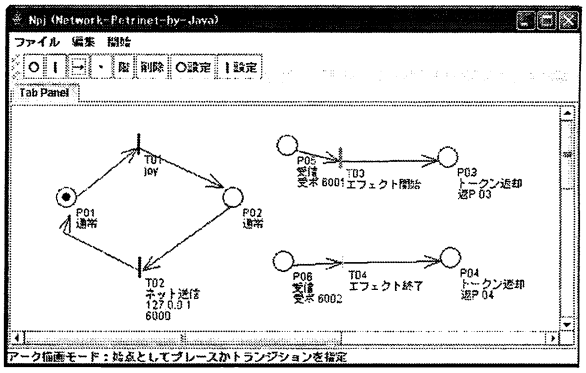

Fig. 14: Petri nets model of the manipulator

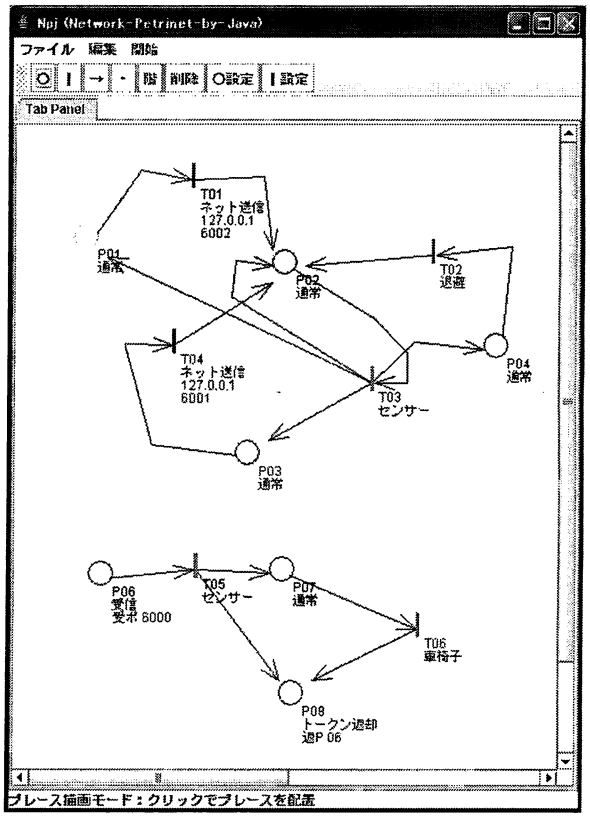

Fig. 15: Petri nets model of the electric wheelchair

参考文献

1) 椎塚久雄, 実例ペトリネット, pp.16/43 , コロナ社, (1992)
2) 村田忠夫, ペトリネットの解析と応用, pp.4/41 , 近代科学社, (1992)
3) 熊谷貞俊, 薦田憲久, ペトリネットによる離散事象システム論, pp.1/11 , コロナ社, (1995)
4) 奥川峻史, ペトリネットの基礎, 共立出版株式会社, (1995)