分散形システムの試験・評価を行うテストベッドシステム

菊野　亨，吉田典可，池田　実，石田賢治
（　広島大学　　工学部　）

1. まえがき

最近，ＯＡ，ＦＡなどの分野へのローカルエリアネットワークの普及に伴い，分散形システムあるいは分散処理システムが注目を集めている[3]．しかし，分散形システムの設計には通信プロトコル，ネットワークＯＳなど解決すべき問題が残されている．

分散形システムにおけるシステムパラメータを決定するための性能評価も未解決な問題の１つである．性能評価法としては，従来，解析的な方法，１台の大型コンピュータ上で行うシミュレーションの方法などがある．これらの方法で分散形システムに固有の性質（例えば，高い処理能率，並列性など）を評価するには，所要時間，評価精度，あるいはコストにおいて困難であることが指摘されている[1]．

そこで，これらの方法の困難点を補完する試みとして分散形シミュレーションが提案され，その１つの実現手段であるテストベッド法への関心が高まってきている[2],[5],[7],[8]．テストベッド法では，複数のコンピュータを並列に用いて分散形システムのシミュレーションを行う．これまでに，リアルタイム処理システムに対するテストベッド[8]，ローカルエリアネットワークに対するテストベッド[2],[5]などが構築されている．

本稿では，分散形システムに対するテストベッド法に対する基礎的考察を行う．先ず，考察の対象とする分散形システム，テストベッド法についてのまとめを2.，3.で行う．4.ではシステム開発時の重要な課題である正当性にも留意したテストベッドシステムのアーキテクチャを提案する．更に，テストベッドに対しユーザが入力として与える，システム記述，及び，テスティング記述についても5.で例を用いて簡単に紹介する．

2. 準備

ここでは，対象とする分散形システムを明らかにする．2.1ではシステムを構築する立場から分散形システムの定義を与える．次に，2.2では制御機構に基づいた分散形システムの分類を行い，対象とするシステムのクラスを定義する．

2.1 分散形システムの定義

分散形システムをEnslow[3]に従って，ハードウェア構成，制御機構及びデータベースの観点から定義する．すなわち，次の(1)－(3)を満たすシステムを分散形システムであると定める（なお分散化の度合に応じて種々の分散形システムが規定できる）．

(1) 分散化したハードウェア構成…システムは複数のコンピュータ（以降，ノードと呼ぶ）と各ノード間を結合する通信回線から構成されている．

(2) 分散化した制御機構…各ノードは各々独立した制御機構を持ち，メッセージを使用してノード間の通信を行う．なお，ノード上の制御機構がすべて同種である必要はない．

(3) 分散化したデータベース…各ノードは直接利用できる独自のデータベースを持ち，かつ，通信回線を利用して他のノード上に格納されているデータベースを利用できる．

以降，上記(2)で説明した各ノード上の制御機構を分散形アルゴリズムと呼ぶことにする．

2.2 分散形システムの分類

対象とする分散形システムのクラスを明示するため，先ずシステムの分類を行う．分類基準として，ここでは分散形アルゴリズムに関する性質（すなわち，停止性，通信レベル，処理の形態）を採用し，これに基づいて分散形システムを分類する．

(1) 停止性…アルゴリズムのステップを有限回実行後に停止するものと，処理要求がある限り動作し続けるものに分ける．

停止する例としてはコンピュータネットワークの分散形ルーティングシステムがあり，非停止の例としてはコンピュータネットワークのフローコントロールシステムがある．

(2) 通信レベル…各ノード上のアルゴリズムの間で行う通信を，ノード間に存在する物理通信路を用いる低通信レベルと，ノード間に論理的に設定される論理通信路まで許す高通信レベルに分ける．なお，論理通信路は物理通信路の系列として構成される．

低通信レベルの例としてはコンピュータネットワークの分散形ルーティングシステムがあり，高通信レベルの例としては分散形データベースの問い合わせシステムがある．

(3) 処理の形態…各ノード上で実行される処理がシーケンシャル処理に限られるものと，コンカレントな処理も含むものに分けられる．

以上，説明した3つの性質に基づいて分散形システムは形式的に図1に示す8つのクラスに分類できる．

本稿で考察の対象とするのは，図1中に斜線で示した4つのクラスである．このように定めた主な理由は，分散形システム上の1つのノード内の内部処理よりもシステムを構成する複数のノード間の相互作用に重点を置いたからである．

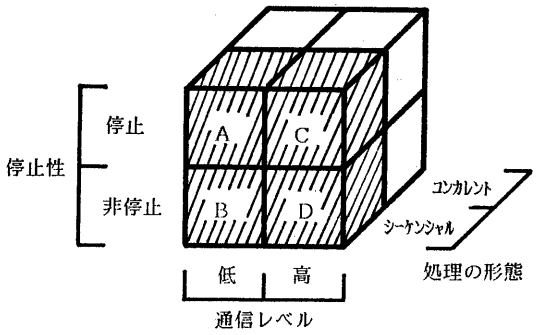

図1 分散形システムの分類

4つのクラスのそれぞれに属する具体的なシステムを紹介しておく．クラスAにはコンピュータネットワークのルーティングシステム，クラスBにはコンピュータネットワークのノードレベルのフローコントロールシステム，クラスCには分散形データベースの問い合わせ処理システム，クラスDにはコンピュータネットワークのエンドレベルフローコントロールシステムなどがある．

3. 分散形システムの性能評価法

ここでは分散形システムの性能評価法として注目されているテストベッド法の位置づけについて説明する．更に，提案するテストベッドシステムで行う性能評価と試験の項目のまとめをする．

3.1 分散形シミュレーション

分散形システムの試験，評価法としては，解析的方法，シミュレーションによる方法，プロトタイプによる方法が知られている．

これらには次の様な問題点がある．解析的方法の場合，対象システムが少し複雑になるとその適用が著しく困難になる．次に，シミュレーション（単一プロセッサ上で行う集中形シミュレーションのこと）を用いて高い精度を得るには多くの時間とコストが必要となる．プロトタイ

プの場合，多くのコストが必要で，しかも専用システムとなり他の分散形システムの評価への利用は困難である．

最近，上述の問題点を補完する1つの方法としてシミュレーションの機能を複数のコンピュータ上に分散化させた，分散形シミュレーションが注目されている．この方法では複数のコンピュータを用いた並列的なシミュレーションを行うため，高い精度を短時間で実現することができる．

3.2 テストベッド法

前節で説明した分散形シミュレーションは，各コンピュータへのシミュレーション機能の分散化の方法により，更に処理分散形とノード分散形の2つに大別される(7)．処理分散形では分散形システムで実行されるプロセスの種類に対応して1台のコンピュータを割り当てる．一方，ノード分散形では分散形システム上のノードに対応して1台のコンピュータを割り当てる．

処理分散形とノード分散形の比較結果を表1に示す．

表1 ノード分散形と処理分散形の比較

	負荷分散	通信量	スケジューリング	分り易さ
ノード分散形	困難	小	不要	大
処理分散形	容易	大	要	小

表1からも分かる様に，ノード分散形は処理分散形に比べると，負荷の分散は若干難しいが，ノード間の通信量が少なく，処理のスケジューリングが不要であって，しかもシミュレーションモデルとの対応が考え易いといった特長を持っている．本稿では，これらの特長に注目して，ノード分散形の分散形シミュレーションをテストベッド法と呼ぶことにする．

テストベッド法の使用例はこれまでにもいくつか報告されている(1)．その開発目的としては評価の精密さ，評価コストの低減化などがある．それらの大部分は特定の分散形システムに対する評価を意図して設計されたものである．これに対して本稿では汎用を目指して設計したテストベッドシステムを提案する．

3.3 評価項目

分散形システムに対する性能評価として，本稿では次に示す(1)(2)の2項目についての評価を考える．以下では，2.2で導入した4つのクラスA～D（特に分散形アルゴリズムの停止性）に基づいて説明する．

(1) 処理時間，使用メッセージ量…クラスA，C
(2) スループット，転送遅延時間…クラスB，D

提案するテストベッドシステムでは，上述の性能評価の外に，次の(3)－(5)に示す項目について試験を行う．

(3) デッドロックの有無．
(4) ライブロックの有無．
(5) システムが設計された通り正しい動作をするか否かの判定．

これらはいずれも分散形システムの設計時には基本的，かつ，重要な問題である．なお，テストベッドシステムでは指示された内容のチェックを行うだけであり，性質についての証明を与えるものではない．

4. テストベッドシステム

4.1 システムの概要

ここでは，2.2で定義した分散形システムのクラスA～Dを対象に，試験，評価を行うテストベッドシステムのアーキテクチャについて説明する．図3にシステムの概要を示す．テストベッドシステムは対象システムのモデル化を行うDSM（Distributed System Modeler）と，これを制御，管理するTH（Testbed Host）

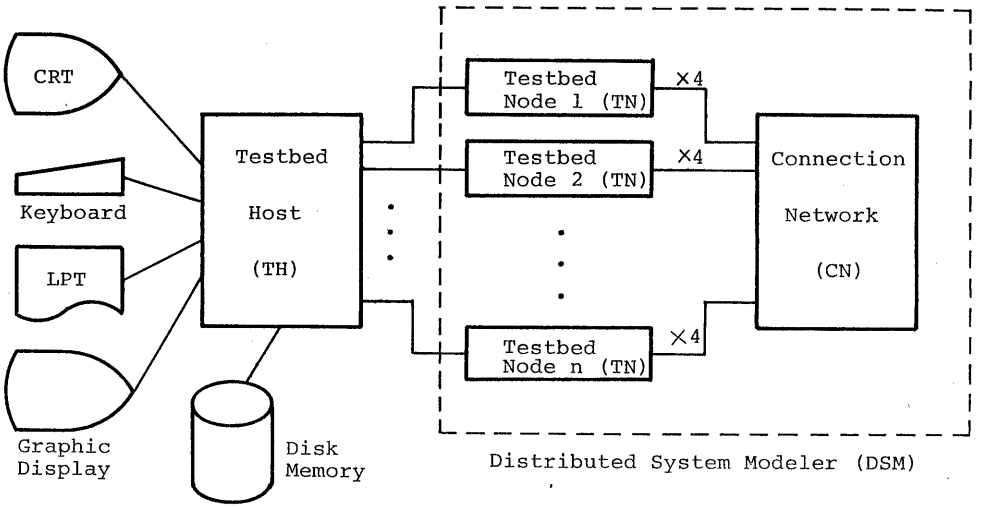

図3　テストベットシステム概要

から構成される．

ＤＳＭにはＴＮ（Testbed Node）とＣＮ（Connection Network)が含まれている．ユーザはＴＨ上でユーザ記述（5.で述べる）や操作コマンドを入力することによって分散形システムの試験，評価を行う．

ＴＮ，ＣＮは，それぞれ，対象システムを構成している１つのノード，通信回線に対応している．ＣＮではＴＮ間での通信を実現すると同時に，通信遅延の発生を行う．

分散形システム上で発生する遅延には，通信遅延以外に，ノードでの処理による遅延，処理の順番待ちによる遅延がある．この内，順番待ちによる遅延はＣＮで発生する．なお，ノードでの遅延は，通常，他の２つに比べて無視できる程度に小さいと考えられる．

ＴＨではシミュレーション時のＤＳＭの制御とモニタを行う．ユーザはＴＨが提供する次の①～④の機能を利用してテストベッドシステムを操作する．

①ユーザとの対話，及び，ユーザコマンドの解析

②コンパイル，及び，ＤＳＭへのロード

③ＤＳＭに対する制御

④ＤＳＭからのデータ収集，及び，データ解析

4.2　ハードウェア構成

先ず，ＴＨのハードウェア構成について説明する．ＴＨは汎用コンピュータにＤＳＭインタフェースとグローバルクロックジェネレータを付加した構成をとる（図4参照）．ＤＳＭインタフェースと各ＴＮ間の通信は全二重である．

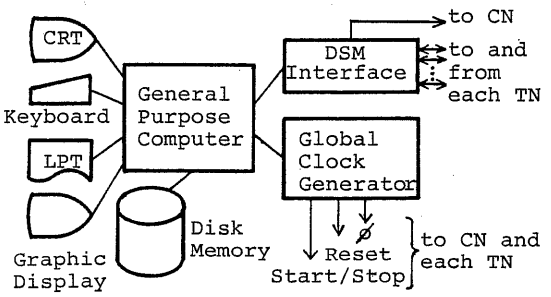

図4　テストベッドホスト(TH)

グローバルクロックジェネレータは各ＴＮとＣＮにグローバルクロックを供給する．このクロックはＴＮのデータ収集時に用いられる（例えば他のＴＮからデータを受信した時刻を記憶するために使

用される）．なお，各ＴＮはローカルクロックで動作している．更に，グローバルクロックジェネレータにはＴＮやＣＮ内のグローバルクロックカウンタを操作するための２本の制御線がある．

次に，ＴＮのハードウェア構成について説明する．ＴＮはプロセッサＰ１，Ｐ２，メモリＭ１，Ｍ２，グローバルクロックカウンタ，Ｐ１とＰ２を結ぶコモンメモリなどから構成されている（図５参照）．

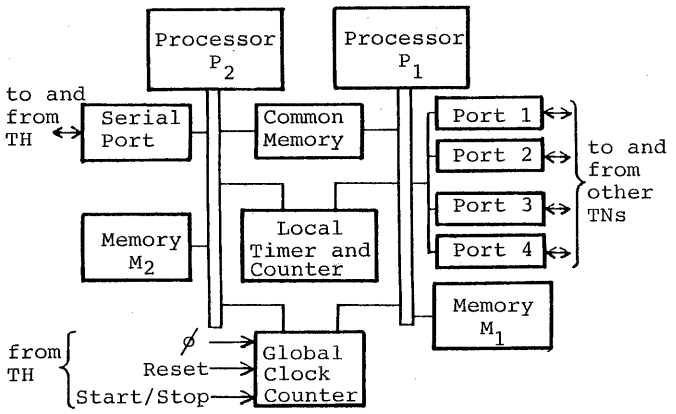

図５　テストベッドノード（ＴＮ）

コモンメモリはＰ１とＰ２間のデータの授受に使われ，ハンドシェーク法で制御されている．グローバルクロックカウンタはＴＨから送られてくるグローバルクロックの数え上げを行う．このカウンタの値を利用してテストベットシステム上のＴＮの動作の無矛盾性が保たれる．ローカルタイマ／カウンタはデータ収集のオーバーヘッドを減らすために使われる．例えば到着したメッセージ数をこのカウンタで保持することで，ソフトウェアの負担を軽減させる．

次にプロセッサＰ１とＰ２の動作について述べる．2.1で説明した低レベルの通信の場合にはＰ１のみが用いられる．一方，高レベルの通信の場合には，Ｐ２で分散形システムの試験，評価を行い，Ｐ１はこれに必要な低レベルの通信を支援する．Ｐ１は他のＴＮとの通信のために全二重のシリアルポートを４つ持っている．各ポートの通信速度は対象システムに合わせて変化させる．Ｐ２はＴＨとの通信のために全二重のシリアルポートを１つ持っている．

最後に，ＣＮのハードウェア構成について説明する．ＣＮはＴＮ間の通信回線を実現するコネクションスイッチ，通信遅延を実現するディレイユニット，ＴＨとの通信を行うシリアルポート，グローバルクロックカウンタ，及び，これらを制御するプロセッサから構成される（図６参照）．

コネクションスイッチの形式は回線交換クロスバを採用する．その理由は通信遅延に関し高い評価精度が保証できること，スイッチの複雑度が低いことなどがある．高精度の評価を保証するには対象システムに存在しない遅延をスイッチ部分で発生しないことが要求される．回線交換クロスバ形式はその遅延が極めて小さいと考えられる．更に，この形式のスイッチはＴＮの数を２０程度と仮定する場合，他の形式（例

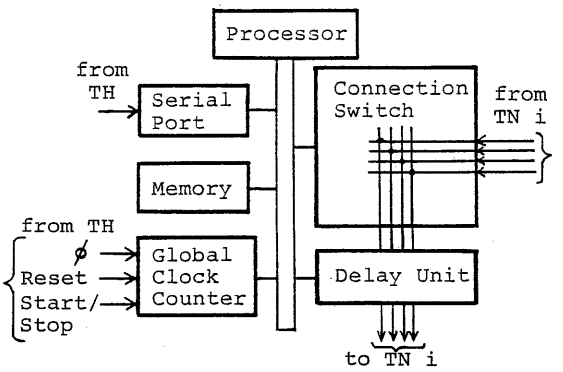

図６　コネクションネットワーク（ＣＮ）

えばbanyanやcube）に比べその複雑度は低い．その他，ブロッキングがない，完全結合となっているといった特長を持っている．短所としては，バス形式の持っている拡張性に欠けること，ＴＮの数が増加した場合にスイッチ部分の複雑度が急激に増大すること，などがある．

4.3　ソフトウェア構成

ソフトウェアはテストベッドシステムのハードウェア構成と対応しており，かつ，階層的なモジュール構造をとっている．図7にソフトウェア構成の概要を示す．

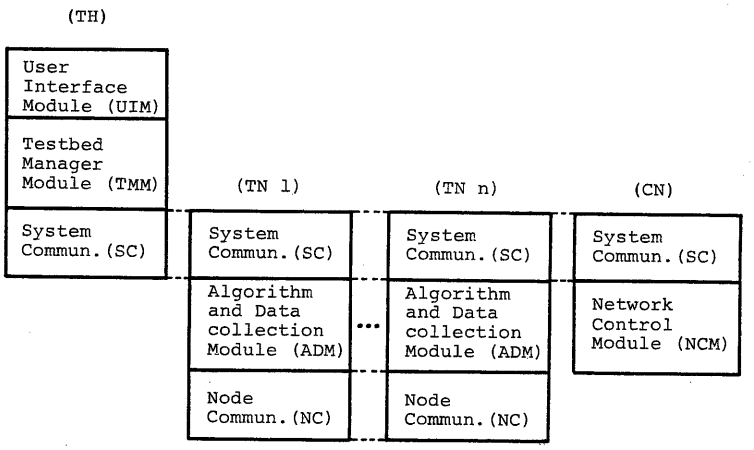

図7　ソフトウェア構成

テストベッドホストＴＨにはＵＩＭ（User Interface Module），ＴＭＭ（Testbed Manager Module），ＳＣ（System Connection）の３つのモジュールが載る．次に，テストベッドノードＴＮにはＳＣ，ＡＤＭ（Algorithm and Data collection Module），ＮＣ（Node Communication）の３つのモジュールが載る．更に，コネクションネットワークＣＮにはＳＣ，ＮＣＭ（Network Control Module）の２つのモジュールが載る．以下，それらについて詳しく説明する．

ＳＣ…ＴＨと各ＴＮ間，ＴＨとＣＮ間の通信を行うモジュールである．主としてテストベッドシステムによる試験，評価の初期時，あるいは，終了時に用いられ，アルゴリズムのロード，評価データなどの集計を行う．

ＮＣ…各ＴＮ間の通信を行うモジュールであり，テストベッドシステムによる試験，評価の実行時に用いられる．

ＵＩＭ…ユーザはＵＩＭを通してテストベッドシステムを操作する．

ＴＭＭ…ユーザ入力に基づくテストベッドシステムの初期化，テストベッドシステムに対する実行時のモニタ，実行後のデータ集計を行うモジュールである．

ＡＤＭ…分散形アルゴリズムの実行，及び，実行時のデータ収集を行うモジュールである．

ＮＣＭ…コネクションスイッチの制御，ディレイユニットに対する通信遅延の設定を行うモジュールである．

図８に各モジュール間のデータの流れを示す．ユーザからの入力は①の矢印で指すように，ＵＩＭを経た後ＴＭＭでコンパイルされ

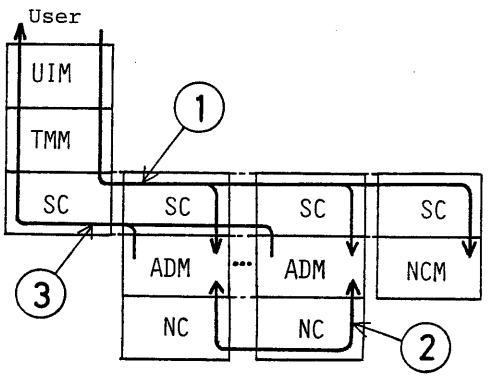

図８　モジュール間のデータの流れ

る．この結果はＳＣによって各ＴＮ上のＡＤＭ，及び，ＣＮ上のＮＣＭにロードされる．分散形システムの実行，及び，評価データの収集は，②の矢印で指すように，各ＡＤＭがＮＣを用いて通信することにより行われる．最後に，収集されたデータは，③の矢印で指すように，ＴＭＭで集計された後にＵＩＭを経てユーザに出力される．

5. ユーザ記述

テストベッドシステムに対する入力として与える，分散形システムの記述，及び，試験，評価条件の記述は図９に示す形式で行う．同図から分かる様に，ユーザ記述はモデル化に関する分散形システム記述と，試験，評価に関するテスティング記述に大別される．

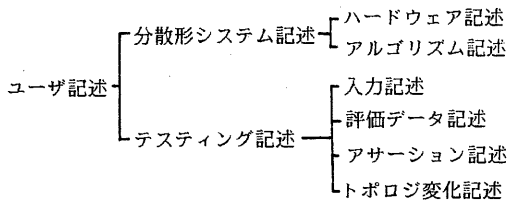

図９　ユーザ記述

5.1 分散形システム記述

分散形システム記述は更にハードウェア記述とアルゴリズム記述に分けられる．前者では分散形システムのハードウェア構成に関する部分を述べ，後者では分散形アルゴリズムに関する詳細を与える．

ハードウェア記述では，テストベッド上で使用するＴＮの数，各ＴＮごとのポート数，各ポートの接続状況，通信速度，及びその遅延時間を指定する．

次に，アルゴリズム記述では各ＴＮ上でのイベントの発生，処理手順を記述する．図10にPascal風な記述言語による記述例を示す．このアルゴリズムはコンピュータネットワークを無向グラフＧで表す時，Ｇ上で定義される１つの極大木を求めるものである(4),(6)．この記述

```
[[Algorithm Description]]
 -for each node-
  (Declaration)
     communucation level:low;
     state var: state:bool;
                father,vd,vp:integer;
                son:<integer>;
     temp  var:i,port:integer;
     array:  m1(1..vp):bool
             m2(1..vp):integer
     message:tree1 (event:up)
                v1:integer;
             tree2 (event:down)
                v2:integer;

(Initialization)
   state:false;
   son:=%; (*empty*)
   #i(=1..vp) do m1(i):false;
   father:=nil;

(Algorithm)
   event: dest
    {tree1.v1:=vd;
     #i(=1..vd) do transfer(tree1,i)}
   event: up
    {port:=eventport;
     m1(port):=true;
     m2(port):=tree1.v1;
     if state=false
      then {state:=true;
            father:=port;
            tree1.v1:=m2(port);
            #i(=1..vp except port) do
            transfer(tree1,i) }
      else {if #i(=1..vp) m1(i)=true
             then {state:=false;
                   tree2.v2:=m2(port);
                    if father<>nil
                     then transfer
                     (tree2,father)}
    }
   event: down
    {port:=eventport;
     m1(port):=true;
     m2(port):=tree2.v2;
     son:=uni(son,port);
     if #i(=1..vp) m1(i)=true
      then {state:=false;
            tree2.v2:=vd;
             if father<>nil
              then tranfer(tree2,
                   father)}
     }
-for each node end-
```

図10　記述例

は宣言部，初期部，アルゴリズム部の３つの記述に大別される．先ず，宣言部では通信レベル，データ構造（状態変数），外部手続き，発生するイベント，使用するメッセージなどの宣言を行う．次に，初期部では状態変数に対し初期値を定める．アルゴリズム部では各イベントごとに対応する処理手続を定義する．今の場合，３種類のイベント dest，up，down に対する処理手続きが与えられている．

5.2 テスティング記述

テスティング記述は更に入力記述，評価データ記述，アサーション記述，トポロジ変化記述の４つに分けられる．以下では，図11に示す記述例を用いて説明する．

入力記述では分散形システムへのテスト入力データを与える．今の場合，時刻０にイベント destがノード２に発生する．次に，評価データ記述では収集すべき評価データ（収集すべき条件，内容など）の指定を行う．アサーション記述では分散形システムの動作の正当性に関する試験の目的で，チェックすべき条件をアサーションとして記述する．最後にトポロジ変化記述ではテスティングの途中に発生させるシステム形状に対する変化に関する記述を行う．

```
[[Input Description]]
 -for node 2-
  <time:0> (event: dest);
 -for node 2 end-

[[Evaluation Data Description]]
 *Data 1*
  -for each node-
   (Declaration)
    external procedure: gettime
   (Data)
    <state=>true> gettime;
  -for each node end-

[[Assertion Description]]
 Local:
  -for node 3-
   (Declaration)
    external procedure: assert
   (Assertion)
    <time=>10> assert;
  -for node 3 end-

[[Topology Change Description]]
 <time: 100> fail: node1=node2.
```

図11 記述例

6. むすび

本稿では分散形システムの試験，評価を行うテストベッドシステムのアーキテクチャを提案した．今後の課題としては，(1)対象システムが大きくなっても十分対応できるコネクションスイッチの開発，(2)ユーザ記述言語のシンタックスの詳細を決定すること，及び，(3)提案したアーキテクチャを実際に構成し，使用すること，などがある．

文献

(1) Berg, H. K.:"Distributed systems testbeds: Experimentation with distributed systems", IEEE Computer, pp.9-11 (Oct. 1982).

(2) Brayer, K. and Lafleur, V.:"A testbed approach to the design of a computer communication network", IEEE Computer, pp.14-23 (Oct. 1982).

(3) Enslow, P. H. Jr.:"What is a "Distributed" data processing system?", IEEE Computer, pp.13-21 (Jan. 1978).

(4) 池田 実:"Testbed architecture for evaluating distributed systems",広島大学大学院修士論文 (Mar. 1984).

(5) 井村，他："マイクロプロセッサを用いたローカルエリアネットワーク・テストベッド"，昭和59年度信学会秋全大，1789 (Mar. 1984).

(6) Segall, A.:"Distributed network protocols", IEEE Trans. on Information Theory, IT-29, 1, pp.23-35 (Jan. 1983).

(7) 渡辺，他："並列処理事象型待ち行列網シミュレータD-SSQについて"，信学技報 IN 83-54，pp. 73-78 (Aug. 1983).

(8) Williams, T. G. et al.:"A hardware architecture for a flexible distributed computing testbed", Proc. 3rd Int'l Conf. on Distributed Computing Systems, pp.404-409 (Oct. 1982).