オンラインシミュレーション*

中　西　俊　男**

まえがき

実在する物，過程もしくはこれらによって構成される対象を何らかの形で表現し，実物ではなく，模擬表現を通して実験を行なうことをシミュレーションという．対象の表現のしかたは必ずしも具体的，全体的である必要はなく，調べようとするもののエッセンスを抽象的に表現してもよい．しかしたとえば，純粋数学の手法のように，きわめて抽象的な対象を，アルゴリズミックな論理，解析を通して表現し解くといった形態のものは，本質的にはシミュレーションと同等であっても通常その範ちゅうより除かれる．シミュレーションは，解析的に解くことがむずかしい，またはその過程に確率的要因がひそんでいて，簡明にして決定論的な表現ができない問題に対して用いられる．前理論的な実験法なのである．

現実の類似表現をモデルと称する．モデルには具体的な物で表現される物理的モデル（Physical Model）と数値の集合，変数などで構成される数学的モデル（Mathematical Model）とがある．前者によるシミュレーションは古来多くの現象の解明に用いられたスケールモデル（たとえば，風洞実験），類推モデル（たとえば，振動の研究のための電気的ネットワーク）による物理実験であり，後者によるシミュレーションは電子計算機の出現によって飛躍的に発展したディジタルシミュレーションである．アナログ計算機によるシミュレーションは，これらの中間地帯にあると考えられる．

最近デイジタルシミュレーションはますますよく用いられる傾向にあり，その手法の研究も盛んになってきている．今後，電子計算機利用に占める位置はいよいよ高くなると期待されるが，ここに一つ注意すべき問題がある．

物理的モデルは通常即物的であり，その実験は実際の時間に対応して行なわれることが多く，実験中人間との接触がきわめて密接に保たれる．アナログ計算機によるシミュレーションまでは，モデル構成，実験に際しての人間と機械の接触が比較的密接であり，人間の知恵のシミュレーションへの介入がある程度許された．ディジタル計算機の出現，特にその計算速度の飛躍は，シミュレーションと人間の接触を引き離し，シミュレーションの実行をわれわれの手のとどかない世界での出来事にしてしまった．われわれは，シミュレーションモデルの作成を終ると，その働きについて考える暇も，指示する暇も与えられず，無理矢理にシミュレーション結果を押しつけられるのである．

シミュレーションが，システム解析に際して有力な武器であることはみとめられながら，一方においてシミュレーションの効果に対する不信が根強く残っている．シミュレーションは対象システムが複雑であればあるほど，有効であるはずのものである．しかし複雑なシステムの正確なモデル化はきわめてむずかしい．また，はたしてシミュレーションが，意図したとおり，実行されたかどうかチェックするのも容易ではない．このような問題に対する検討を十分に行なわず，おざなりのモデルにより行なわれた，不完全なシミュレーションで事を済まそうとする傾向があることが，シミュレーション不信の一つの原因であることは否めない．

このような実情に対する対策として取りあげられたものが，オンラインシミュレーションである．ここでいうオンラインの意味は，シミュレーション実行中，人間と機械の接触を密にし，モデル作成，変更，計算結果の評価とそのモデルへのフィードバックなどが直接的に行なわれる形態ということである．いわゆるアナログ計算機によるシミュレーションは，本来この形のものが多く，たとえば微分解析器により境界値問題を解く過程などは，人間と機械の協力が基本となっている．そこでここでは，アナログ計算機によるオンラインシミュレーションではなく，現在等閑に付されているディジタル計算機でのオンラインシミュレーションを中心に論じることにする．

まず最初にシミュレーションの現状と問題点，オン

---

* On-Line Simulation, by Toshio Nakanishi (Systems Engineering Laboratory, Railway Technical Research Institute)

** 鉄道技術研究所　システム研究室

ライン化することで期待できる効果等について述べる．シミュレーションには通常，離散変化モデル（Discrete-Change Model）に関するものと，連続変化モデル（Continuous-Change Model）に関するものとあり，シミュレータもこれに対応して異なる系統のものが開発されている[1],[2]．いずれの場合もオンラインの特性を持たせることの重要性が認められ始めたのは比較的新しい．一つのシステムとして完成され，実用化されている例，もしくはその報告はきわめて少ないが，ここでは，連続システムシミュレーションの例（PACTOLUS[3]）や，MIT の Project MAC での例[4]（OPS-3）[4],[5]について触れてみることにした．

1. オンラインシミュレーションの意義

1.1 シミュレーションの現状と問題点

シミュレーションの対象となる領域はきわめて広範であり，シミュレーション手法も対象の特性に応じて最も有効であるべく，多種多彩である．しかし計算機によるシミュレーションの中核はシステム・シミュレーションで，その技術もこのところ著しい進展を見せている．

ここでいうシステムとは，単体もしくはそれらの単なる集合に対する名称ではなく，構成要素間がある規則的な相互作用もしくは相互依存の関係によって結合されているような集合体を指す．システムにはいろいろな現象が存在し，その結果そこにいろいろな変化が生じる．システムの変化とはその中の対象の特性の変化，もしくは対象の間の関係の変化をいう．人間社会には多くのシステムが考えられる．シミュレーションの対象となるシステムは，その中でも特に規模が大きく，複雑な構成のものである．

システム・シミュレーションを実行するための第一の仕事は，システムの分析とモデル化である．現在のディジタル・シミュレーションにおける一つの問題点はモデル化にある．この場合のモデルには，物理モデルの場合と異なり，かなりの抽象化と省略がつきものである．システムが複雑であればあるほど，その本質的な部分を抽出し，計算機での取扱いが容易である形に記述することが必要である．

システムをどのように見，どのように表現するかはその本質を見失わない限り自由である．システムの変化をある連続量の変化と見る必要がある場合もあれば，ある種の省略により離散型の変化と見る方が，より本質を見きわめるに都合のよいこともある．このような自由度は，モデル化技術の発達にいくつかの方向を与えた．

現在モデル化とその表現を容易にするいくつかのシミュレーション言語が開発され，一頃と比べるとモデル化の仕事は格段にやさしくなっている．しかし，それにもかかわらずシミュレーションの効果が疑問視される原因の一つはモデル化過程のずさんさにある．

ところでこのずさんなモデル化の責任は，必ずしもシミュレーション言語の不備，もしくはモデル作成者の怠慢に帰せられない面がある．それはシミュレーションの対象にされるシステムの本質に由来しているとも考えられる．その本質とは

（1） システムの構成諸要素間の関係，その相互作用のしかたが，システムの表面的な観察からだけでは，究明しがたいものがある．

（2） 現実のシステムの変化過程に，人間の知恵が介入している，つまりヒューリスティックな過程が交錯している．

（3） 試行錯誤もしくはある種の予測に基づく判断の過程が含まれている．

などである．これらは必ずしも独立ではなく，現実にはこれらが融合した形でシステムのモデル化を困難にしている．

現在のシミュレーションにおけるもう一つの問題点はシミュレーション結果の評価と，そのモデルへのフィードバックの過程である．このことは，一つのシミュレーションが終了し，データもしくはモデルを変更してシミュレーションを繰り返す場合についてだけではなく，シミュレーション実行途中にも考えられる．

現在開発されているシミュレータでは通常時間の逆行は許されない．またそれが許され，途中での計算結果を参考にして，データもしくはモデル変更を行なうとしても，このような過程を洗いざらい取り入れ，完全に自動的に処理できるようにすることは至難である．

1.2 オンラインシミュレーションへの期待

上に述べた問題点は人間と計算機の協力態勢の不備に帰因するところが大きい．そこでシミュレーションをオンライン化する，つまりシミュレーション実行中もしくは繰り返しにあたって，人間と計算機が直接交渉できる態勢にすることにより，これら問題点がどの程度改善され得るかについて考えてみたい．

（1） モデルの作成，変更およびデバッグ

前にも述べたようにシミュレーションの対象となるほどのシステムは，その論理構成がきわめて複雑であり，直観的にモデル化することがむずかしい．特に，互いに並行して進む同時現象の間の関連を正確にとらえ，正しく記述することは容易ではない．さらにシミュレーションプログラムは，通常の数値計算プログラムのように，結果のわかっている簡単なチェックモデルによって，デバッグを済ますわけにはゆかない．したがって，モデル化およびプログラムデバッグなどに関しては，最初作成されたモデルについてシミュレーションを実行させ，その進行過程を人間が監視し，必要に応じて進行を止めて途中結果の検討，モデルの手直しを行なうことが必要になる．これらは，すべてオンラインシミュレーションの特徴であり，モデル作成変更，プログラムデバッグにおける問題点はシミュレーションのオンライン化によって相当に改善されると期待される．このようなシミュレーションの実行を可能にするためには

・シミュレーションの進行を何らかの形で表示するか，途中にブレークポイントを入れるかして，随時シミュレーションを中断し，再開できるようにすること．

・シミュレーション結果，もしくはその肝要な過程の進行状況を表示できること．

・随時プログラムもしくはデータの変更，訂正が可能であること．

・シミュレーション中断時点以前の任意もしくは指定された時刻におけるモデル状況を再現し，そこからシミュレーションを再開できる，すなわち時間の逆行が許されること．

などが必要である．このようなシミュレーションシステムが実現すれば，ダイナミックなモデルの作成，変更，プログラムデバッグが可能であり，より正確な，もしくはより有効なシミュレーションが実行できると考えられる．

（2） 試行錯誤過程の処理

試行錯誤は，論理もしくは直観で物事の是非が判断できないとき用いられる常套手段であり，複雑なシステムではこの形の過程が数多く見られる．オンラインシミュレーションでのモデル作成，変更も，いわば試行錯誤の効果に期待するところが極めて大きい．

ここでとり上げた試行錯誤はオンラインの処理に関するそれではなく，シミュレートしようとしているシステム内部の試行錯誤である．実際に，何かについて決定しようとするとき，机上でシミュレーションを行ない，その結果を見て判断するようなことが往々にしてある．このようなシステムのシミュレーションでは，実際のシステムにおける机上シミュレーションが比較的簡単で，単純な非シミュレーション計算におきかえ得るものならばともかく，メインのシミュレーションに相当する規模と複雑さを持っていて，むしろメインのシミュレーションを続行する形が望ましい時，問題を生じる．このような過程の正確な処理を行なうためには，やはり時間の逆行を含む各種のオンラインシミュレーションの便宜をとり入れることが必要である．

たとえば貨車操車場において，貨物列車組成開始時刻を定める過程などは一つの典型的な問題である．実際には構内作業ダイヤというものがあり，そこで定められた時刻に組成作業が開始されることになっているが，将来の自動化システムでは，労働時間の短縮，列車組成の合理化，組成線その他の支障の軽減などの観点から，最も効果的な組成開始時刻を定めることが望ましい．しかし組成内容，そのために要する時間は，操車場の全体的な状勢とその時間的な変化に関連するものであり，簡単な計算で算出できるものではない．このような過程の処理について，オンラインシミュレーションにかけられる期待は大きい．

（3） リアルタイムシミュレーション

オンラインシミュレーションには必ずしも直接的な関係があるわけではないが，きわめて関連の深いものにリアルタイムシミュレーションがある．リアルタイムシミュレーションと一口にいってもその内容は単純ではなく，いろいろな形態のものが考えられる．リアルタイムシステムの中で，リアルタイムプロセスの処理に利用されているシミュレーション，リアルタイムプロセスに関連する系の働きを事前に調査もしくは訓練するために行なわれるシミュレーションなど，いずれもリアルタイムシミュレーションの範ちゅうに入れられよう．

たとえば，人工衛星の回収地点を決定するために，その飛行中絶えず行なっている落下回収のシミュレーションなどは，リアルタイムシステムの中の，リアルタイム処理に関連するシミュレーションという意味で一つのリアルタイムシミュレーションと考えられる．

また人工衛星の打上げ，制御に関する諸々の機器，設備などがうまく実時間で動作するかどうかを見るために，実際に人工衛星を飛ばした時と同じ状態をつくり出し，シミュレーションによりチェックするといっ

たことも，リアルタイムシミュレーションと考えられよう．これは，リアルタイム処理がうまく行なわれるかどうかを全体のシステムのうち一部をモデルで表現して行なうシミュレーションで，人間の訓練を目的としたシミュレーションもその一例と考えられる．

最終的に人間と機械の協力態勢が基本となるシステムの設計のためのシミュレーションでは，特にマンマシンインタフェイスの部分の特性を調べたり，担当者を訓練したりするために，このような形のリアルタイムシミュレーションをオンラインで行なうことが重要である．このようなシミュレーションを行なうには，実用機器を直接連結したり，それらに全く同等な環境を適宜作ることができ，かつ各種ディスプレイ，ライトペンその他の入出力装置を自由に駆使してオンラインシミュレーションを実行できるようなシミュレーションシステムを実現させることが先決である．このようなシステムでは，計算機の働きを人間の歩調に合わせなければならず，現実にはタイムシェアリングシステムの実現が先行すべきである．この観点から完備したと考えられるシミュレーションシステムはまだ報告がなく，今後の研究に期待がよせられる課題である．

2. シミュレーションシステム

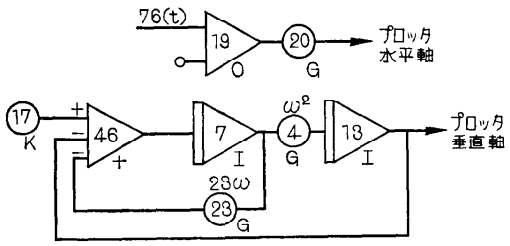
シミュレーションを行なうことを目的とした言語システムは数多く開発されているが，そのほとんどは，オンラインの使用ができないものである．連続変化モデルを取り扱う系統のものではオンラインの使用を考慮しているものに PACTOLUS があり，またより一般的なオンラインシミュレーションシステムとして Project MAC の OPS-3 の中のそれをあげることができるが，いずれも理想的な形で完成されたものではなく，今後の発展に期待されるところが大きい．しかし現段階で得られる数少ない試行例なのでここでとりあげてみることにした．

2.1 PACTOLUS

ディジタルなアナログシミュレータプログラムは，1955 年，Selfridge により初めてつくられて以来，その実用化されている数は驚くほどになっている[1]．代表的なものとしては DEPI, ASTRAL, DEPI 4, DYSAC, PARTNER, DAS, JANIS, MIDAS などがある．特に MIDAS はそれ以前の各種プログラムの特長をとり入れると同時にいくつかの創意工夫を加え最も優れたシミュレータと評価されている．しかし，これらはいずれもブロックオリエンティドな言語でシミュレーションモデルを表現し，計算を実行するという機能が主体で，アナログ計算機に見られるオンライン機能を十分代替する便宜を具えてはいない．MIDAS の後で開発された PACTOLUS はオンライン使用を考慮して開発されたという点で一つの進歩を示したと考えられる．

PACTOLUS は一口にいってオンラインのコントロールと入出力が可能なブロックオリエンテイドのプログラムである．1627 プロッタとカードリードおよびパンチを具えた IBM 1620 計算機システムに対して開発されたものである．作成にあたっての最大の関心事は，オペレーションの融通性をいかに発揮させるかということで，プロッタ，タイプライタ，センススイッチなどがその目的に十分利用される形になっている．たとえば，プロットされるアウトプットを見て，適宜任意にインタラプトをかけ，モデル仕様，パラメータもしくは初期条件の修正を行なうことができる．

PACTOLUS は，標準的なアナログ計算機の持つ機能の他に，特殊な目的のアナログ回路に対応する命令を持っている．その種類と内容については参考文献3)に詳しい．ここでは所期の目的からはずれるので，これらについて詳しく述べることは避け，PACTOLUS のおよその模様を示す例題を示すにとどめる．



第 1 図　2次システムのシミュレーションダイヤグラム

第1図はある2次のシステムのシミュレーションダイヤグラムである．

ブロック 17 は定数の入力を示し，ブロック 46 はブロック 17, 13, 23 からの入力の和（+），差（−）をとる．ブロック 4 および 13 はゲインポテンショメータであり，7 および 23 はインテグレータである．ブロック番号は 1〜75 の任意の数をとることができる．ブロック 76 は特殊なもので，時間変数を示す．

このようなモデルの表示はカードデックを読ませて行なってもよいが，**第2図**に，タイプライタからこれを行なった場合の記録を示す．

```
                    CONFIGURATION SPECIFICATION
          BLOCK      TYPE     INPUT 1    INPUT 2    INPUT 3
           (17)       (K) ;    (    )     (    )     (    )
           (46)       (+)      ( 17 )     (-13)      (-23) ;
           ( 7)       (I)      ( 46 ) ;   (    )     (    )
           (23)       (G)      ( 7) ;     (    )     (    )
           ( 4)       (G)      ( 7) ;     (    )     (    )
           (13)       (I)      ( 4) ;     (    )     (    )
           (19)       (O)      ( 76) ;    (    )     (    )
           (20)       (G)      ( 19) ;    (    )     (    )

                    INITIAL CONDITIONS AND PARAMETERS
          BLOCK      IC/PAR1          PAR2          PAR3
           (17)    ( 0.0   ; )    (          )   (          )
           (13)    ( -100.0 ; )   (          )   (          )
           ( 4)    ( 1.0   ; )    (          )   (          )
           (23)    ( 0.8     )    (          )   (          )
           (20)    ( 20.0  ; )    (          )   (          )
           (19)    ( -5.0  ; )    (          )   (          )
(  0.05 ;  ) INTEGRATION INTERVAL
(  10.0 ;  ) TOTAL TIME
( 2.0   ;  ) PRINT INTERVAL
(20)  ;      HORIZONTAL AXIS
(13)  ;      VERTICAL AXIS


TIME        OUTPUT(46)      OUTPUT( 7)      OUTPUT(13)      OUTPUT(23);
  0.000      100.00000        0.00000      -100.00000        0.00000
  2.000      -30.57587       47.30857        -7.27098       37.84886
  4.000      -13.03217      -11.02521        21.85234       -8.82016
  6.000        9.20057       -6.96698        -3.62699       -5.57358
  8.000        .48782        3.84609        -3.55969        3.07587
 10.000      -1.96715         .50806         1.56070         .40645

                    INITIAL CONDITIONS AND PARAMETERS
          BLOCK      IC/PAR1          PAR2          PAR3
           (23)    ( 1.2 ;  )     (          )   (          )


TIME        OUTPUT 46       OUTPUT  7       OUTPUT 13       OUTPUT 23
  0.000      100.00000        0.00000      -100.00000        0.00000
  2.000      -23.43237       37.60320       -21.69146       15.12788
  4.000       -8.64924        -.65464         9.43481        -.78556
  6.000        2.27288       -3.39439         1.80038       -4.07327
  8.000         .74380         .11838         -.88587         .14206
 10.000        -.21880         .30537         -.14764         .36645

                    INITIAL CONDITIONS AND PARAMETERS
          BLOCK      IC/PAR1          PAR2          PAR3
           (23)    ( 0.4 ;  )     (          )   (          )


TIME        OUTPUT 46       OUTPUT  7       OUTPUT 13       OUTPUT 23
  0.000      100.00000        0.00000      -100.00000        0.00000
  2.000      -38.09478       63.26133        12.79025       25.30453
  4.000      -25.56784      -32.19054        38.38406      -12.87621
  6.000       30.08132      -12.01936       -25.77358       -4.80774
  8.000       -3.85580       20.56715        -4.37106        8.22686
 10.000      -11.54220       -5.06982        13.57013       -2.02792
```

第 2 図　PACTOLUS によるシミュレーションプログラム例

使用者はまずセンススイッチ #1 をセットし，タイプライタからモデル仕様を入力することを報せる．計算機はこれに応じて

CONFIGURATION SPECIFICATION

BLOCK TYPE INPUT 1 INPUT 2 INPUT 3

(　)(　)(　)(　)(　)

とタイプアウトするので，使用者はタイプライタのローラを 1 行もどし，かっこ内に所定のデータを入れる．

第 2 図に示すようにすべての入力が終るとセンススイッチ #1 を切り，初期条件を入れるためにセンススイッチ #2 をオンにする．こうして入力されたデータは，同時にカードにパンチされ，あとで利用できる．

センススイッチの用途をまとめると，

センススイッチ #1: オンにすることにより，モデル仕様と，初期条件／パラメータの指定を行なうことができる．

センススイッチ #2: オンにすることにより，初期条件／パラメータの指定を行なうことができる．

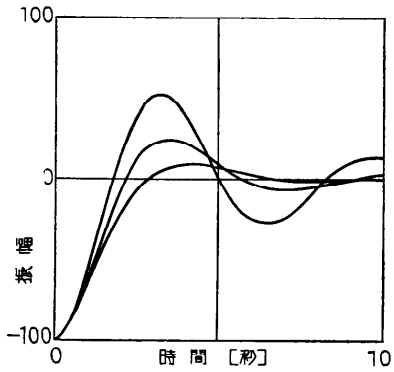
センススイッチ #3: タイミングおよび出力の指定を行なうことができる．

センススイッチ #4: プロッタの紙を動かし，新しいプロットのわくを準備する．

センススイッチ #3 は独立に使用してもよいし，セ

ンススイッチ #1 もしくは #2 と組み合わせて使用してもよい．センススイッチは常にいくつかの他のスイッチと組み合わせて用いられる．もしすべてのセンススイッチをオフにして計算を再スタートさせると，このプログラムは全く新しいシミュレーションが行なわれようとしているととり，モデル仕様の指定を読むことになる．

第2図の例ではゲインブロック 23 のパラメータを3とおり変えて計算を行なっている．おのおのの計算に 15 秒要した．**第3図**はこの結果をプロットしたものである．



第 3 図　2次システム例のプロット記録

PACTOLUS 開発の目的は，それより前につくられたいくつかのシミュレータの技術を評価し，アナログ計算機と同等の運用ができるディジタルアナログシミュレータをつくることであった．オンライン特性を持たせることがその中心的な課題であったが，必ずしも現段階で満足できる域に達したとは考えられない．3) の著者によれば，連続システムのディジタルシミュレーションは，その有意義な拡張の一端にとりかかった段階であり，適当な端末装置やディスプレイが十分利用できるようになれば，より効果的な進展が期待されると結んでいる．

2.2 OPS-3

1964 年春，MIT では新しい計算機システムに関するセミナーが開かれ，タイムシェアリングシステムとオンラインインタラクションについて討議された．その結果 OPS-1，続いて OPS-2 が作られたが，さらにその経験を生かし，同年秋これらの改良型 OPS-3 が開発された．OPS-3 はその開発にあたって特にオンラインシミュレーションのための機能について配慮されたので，ここでとり上げてみることにした．

OPS-3 のオンライン特性を理解するためには，OPS-3 についての概略の知識が必要であるので，簡単に紹介することにする．

OPS システムはそもそも，オンラインの計算，プログラミングおよびモデル作成のために開発された多目的のシステムである．それはオープンシステムでありモデュラ構成をとっている．使用者はそれぞれの目的に従って，それを自由に拡張整形できる．

OPS-3 システムは一口にいってあらかじめコンパイルされたサブルチーンであるオペレータで構成されたシステムである．オペレータには，サブルチーンの変数に相当するパラメータがあり，オペレータの機能に融通性を持たせている．たとえばAを行列とすると，

PRINT A SUM I 2 DIFF J 3 によって，A (I+2, J−3) がタイプアウトされる．

OPS-3 にはつぎのような五つのモード，すなわち使用者のとる状態がある．

（1） Execute

たとえば PRINT とか，SET といったオペレータを順々に一つずつ実行しているような時のモードである．

（2） Store

いくつかのオペレータで，あるプログラムを構成する時とるモードである．

（3） Store and Execute

オペレータを一つずつ実行し，かつ同時にそれらよりなるプログラムを構成する時のモードである．

（4） Run

複合オペレータを中断せずに実行するモードである．

（5） Guide

システムについて何らかの情報を得たいと思う時はこのモードがとられる．

これらのモードへの出入りはシステムコードにより実行される．システムコードはすべて，ドット（・）で始まるが，たとえば

.X: Execute Mode に入れ．

.S: Store Mode に入れ．

.SX: Store-and-Execute Mode に入れ．

のごとくである．

システムコードは，この他いくつかのシステムへの命令を実行する．たとえば

.LA: line A を現在の line とせよ.
.DA: 現在の KOP の，現在の line から始まるA個の line を削除せよ.
.TABC: センススイッチ A, B, C をオンにせよ.
.TL: line trace をオンにせよ.
.RESUM: 最後のインタラプトポイントからプログラムを再開せよ.

など，合計 19 の命令がある.

OPS-3 ではいくつかのオペレータで複合オペレータをつくることができる．これを KOP と称する．各オペレータのラインは通常 10 きざみで番号がつけられる．一例をあげると

```
KOP SOLVEX
10 SET A=1.3, B=−5, C=A
20 SET X=(−B+SQRT.(B*B−
                    4*A*C))/2*A
30 PRINT X
```

のごとくである.

OPS-3 を理解するには，この他いくつかのオンライン特性を説明する必要があるが，紙面の都合上省略し，以後シミュレーションを中心に述べることにする.

シミュレーションモデルは，アクティビティを表わすオペレータおよび KOP で構成される．アクティビティ実行順は必ずしも事前に明らかではないので，通常のサブルチーンコールの形でフローのコントロールを行なうわけにはゆかない．GPSS, SIMSCRIPT などでもそれぞれにタイミングのコントロールルチーンがあり，シミュレーションの進行をコントロールしている.

OPS-3 では AGENDA と呼ばれる特別の KOP を設け，アクティビティのダイナミックスケジュリングを行なっている．アクティビティが完了すると，コントロールは AGENDA に返えされ，AGENDA がつぎに実行するアクティビティを指定する．アクティビティは指定された時刻もしくはある特定の条件が満たされたとき実行されるが，これらはすべて AGENDA で計画される．シミュレーション実行中にも，アクティビティの計画，削除，再度自分自身もしくは他のアクティビティの計画を行なうことが許される．アクティビティは DELAY もしくは WAIT などの命令により，一定時間もしくはある条件が満たされるまで，シミュレーションでの時間を消費することもできる.

OPS-3 のシミュレーションシステムでは，シミュレーションの実行過程よりもむしろモデル作成過程におけるオンライン機能の充実に重点をおいている．この方針により，インタープリティーブなものとコンパイル的なものとの結合システムが生まれた．すなわちオペレータはコンパイルされたプログラムで，効率よく実行される．しかし KOP はインタープリティーブに実行され，細部にわたってトレースすることが許される．ただし，一たんモデルが完成すると，KOP として書かれたすべてのアクティビティをコンパイルしてオペレータとし計算効率をあげることができる．シミュレーションモデルのオンライン作成では，あらかじめ部分部分のテストを行ない，あとで全体を構成するのがよい．構成順としては，外部から内部へ浸透するとか，階級的に行なうとかいろいろ考えられるが，OPS-3 ではこのような処置が可能である.

OPS システムではモデルの各部分は解放されており自由に修正できる．AGENDA 内部もまた自由にコントロールできる．すなわち，システムスイッチを使って AGENDA のどこからシミュレーションを実行するかとか，実行時間，終了の条件などを指定できるし，コンソールから AGENDA を変更，追加することによりシミュレーションの進行を修正できる．さらに，任意の時点でインタラプトをかけ，プログラムもしくはデータを変更してシミュレーションを再開することも自由である.

手広いトレース機能も完備しており，たとえば実行された KOP の名前，同じく line 番号，実行されたオペレータのパラメータと結果，シミュレート時間の動き，シンボルでリファーされた変数の値などすべてトレースできる．トレースはシステムスイッチによりコントロールされるが，このスイッチはコンソールからセットしてもよいし，KOP の中でダイナミックにセットしてもよい.

シミュレーション実行に際してその中枢の働きをなす AGENDA について少し詳しく述べておく.

AGENDA は通常アクティビティへのコールを持っており，その AGENDA における順位は，line 番号の順である．line 番号はシミュレート時刻に対応している.

AGENDA の初めにはアクティビティの条件付コールがおかれ，あとに無条件コールが続く．通常 AGENDA は頭から実行され，最初条件付コールについて実行すべきかどうかテストされる．もし条件が満たされていればそのアクティビティが実行される．条件付

コールについての実行が終ると初めて，最初の無条件コールが実行され，システム変数 TIME がそのコールの line 番号まで進められる．変数 TIME は常にシミュレートされる時間の現在値を示す．

AGENDA でアクティビティの計画を立てるオペレータに SCHED があり3とおりの型がある．

（1） SCHED NAME AT T

（2） SCHED NAME IMMED

（3） SCHED NAME WHEN CONDITION

(1) では NAME というアクティビティがパラメータTの現在値の時刻にスケジュールされ，(2) では同じく TIME の現在値で AGENDA におかれる．ただ

```
KOP ARRIVE
10 SET Q = Q + 1
20 DRAW DT EXPONE 5
30 SCHEDK ARRIVE AT SUM TIME DT
40 SCHEDK STAT IMMED
50 RETRNA

KOP STAT
10 IF Q.LE. QMAX
20 GOTO 50
30 SET QMAX = Q
40 PRINT QMAX
50 IF S .GE. 5
60 WAIT LESS S 5
70 SET S = S + 1
80 SCHEDK SERVIC IMMED
90 RETRNA

KOP SERVIC
10 SET Q = Q - 1
20 DRAW STIME RANDOM 5 50
30 DELAY STIME
40 SET S = S - 1
50 RETRNA
```

第 4 図 OPS-3 によるシミュレーションプログラムの例

しこの場合そのアクティビティは直ちに実行されるという意味ではない．直ちに実行させるには SCHED を使わず直接そのアクティビティを呼ばなければならぬ．(3) のオプションでは，NAME というアクティビティが条件付コールとして，TIME の現在値に等しい line 番号を付され，同じもしくはより小さい line 番号のコールの後，かつより大きい line 番号のコールの前におかれる．条件は Prefix notation でつぎのように指定される．

SCHED NEXT WHEN OR LESS A B GREATER C 500. アクティビティ NEXT は A<B もしくは C>500 の条件が満たされたとき実行される．

この他シミュレーションに関連して，SCHED オペレータでのパラメータの送り込み，アクティビティの優先順位付け，PRINTK AGENDA による任意時刻での AGENDA のチェック，ローカルバリアブルの使用，乱数もしくは各種確率変数の発生，FIFO, LIFO によるスタックの処理，入出力その他いろいろな機能が付与されているが，詳細については参考文献4)，5) を参照されたい．

最後に OPS-3 におけるシミュレーション例（**第4図**）を掲げ，シミュレーション進行に従って AGENDA がどのように変遷するかを見ることにする（**第5図**）．

第4図は複数サービスが行なわれている待行列モデルの例である．三つのアクティビティに対応する KOP, ARRIVE, SERVIC および STAT がある．KOP ARRIVE はサービス待ち行列の数（Q）を増加させる．サービスを受ける人の到着はポアッソン過程（平均到着間隔は 5）でシミュレートされている．最後に KOP STAT をスケジュールし，コントロールを AGENDA に返えす（RETRNA）．KOP STAT は待行列の人の最大数のテーブルつくり，5人のサーバのうち誰かがあいているかどうかをチェックする．もしあいていなければあくまで待つ．その後サーバの数（S: 現在サービスを行なっている人の数）を1ふやし，KOP SERVIC を直ちにスケジュールして AGENDA にコントロールを返えす．KOP SERVIC は，行列の人数を1だけ減じ，サービス時間を計算し，サービス終了後サーバの数を1減らして，コントロールを AGENDA に戻す．

このシミュレーションは，ARRIVE を AGENDA の line 1 にセットして始められる．ARRIVE は自分自身のスケジュールを行なったのち，STAT をスケジュール，STAT はサーバが空いている時だけ SERVIC をスケジュールする．

第5図のリスティングは，シミュレーション進行に伴う AGENDA の変遷を示す．

AGENDA の思想は融通性に富むものであり，シミュレーションにだけ応用されるだけではなく，タイムシェアリングの他に，他のリアルタイムオペレーションも AGENDA メカニズムで考えることができる．MIT ではつぎの OPS として，オペレーションや応用範囲などでより効果的なシステムを計画しているという．

KOP AGENDA
1 CALLAK ARRIVE
TIME = 1

The initial AGENDA

KOP AGENDA
1 CALLAK STAT
9 CALLAK ARRIVE
TIME = 1

KOP ARRIVE has just been executed; it scheduled KOP STAT and rescheduled itself.

KOP AGENDA
1 CALLAK SERVIC
9 CALLAK ARRIVE
TIME = 1

KOP STAT has just been executed and has scheduled KOP SERVIC.

KOP AGENDA
9 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
TIME = 9

KOP SERVIC has been interrupted at the DELAY statement on line 30.

KOP AGENDA
9 CALLAK STAT
20 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
TIME = 9

The beginning of the second cycle of calls, as KOP ARRIVE has just been executed for the second time.

KOP AGENDA
9 CALLAK SERVIC
20 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
TIME = 9

KOP AGENDA
20 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
TIME = 20
•
•
•

Three cycles have been skipped for brevity.

KOP AGENDA
33 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 33

This is the end of the fifth cycle. All five servers are now busy.

KOP AGENDA
33 CALLAK STAT
37 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 33

KOP AGENDA
33 CALLBK STAT .L 70 LESS S 5
37 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 37
•
•
•

Note the conditional call at the top of the AGENDA. KOP STAT has been interrupted because there is no free server.

Several cycles have been skipped.

KOP AGENDA
33 CALLBK STAT .L 70 LESS S 5
37 CALLBK STAT .L 70 LESS S 5
39 CALLBK STAT .L 70 LESS S 5
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLAK ARRIVE
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 47

The end of the eleventh cycle. There are now 6 arrivals in Q waiting for service.

KOP AGENDA
33 CALLBK STAT .L 70 LESS S 5
37 CALLBK STAT .L 70 LESS S 5
39 CALLBK STAT .L 70 LESS S 5
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLAK STAT
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 47

KOP AGENDA
33 CALLBK STAT .L 70 LESS S 5
37 CALLBK STAT .L 70 LESS S 5
39 CALLBK STAT .L 70 LESS S 5
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLBK STAT .L 70 LESS S 5
49 CALLAK SERVIC .L 40
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 49

The number of arrivals in Q is now 7.

KOP AGENDA
33 CALLBK STAT .L 70 LESS S 5
37 CALLBK STAT .L 70 LESS S 5
39 CALLBK STAT .L 70 LESS S 5
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLBK STAT .L 70 LESS S 5
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 49

Note the first server is now free (line 49 has disappeared). S is down to four.

KOP AGENDA
37 CALLBK STAT .L 70 LESS S 5
39 CALLBK STAT .L 70 LESS S 5
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLBK STAT .L 70 LESS S 5
49 CALLAK SERVIC
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
TIME = 49

The arrival at the head of the Q has now been scheduled for service (line 49). Q is now 6.

第 5 図　AGENDA の変遷（1）

```
    KOP AGENDA                    All five servers are
37 CALLBK STAT .L 70 LESS S 5     now busy again. The
39 CALLBK STAT .L 70 LESS S 5     simulation has
40 CALLBK STAT .L 70 LESS S 5     reached a steady state
41 CALLBK STAT .L 70 LESS S 5     condition. The pattern
45 CALLBK STAT .L 70 LESS S 5     of the last cycle
47 CALLBK STAT .L 70 LESS S 5     will continue.
51 CALLAK SERVIC .L 40
55 CALLAK SERVIC .L 40
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
68 CALLAK SERVIC .L 40
TIME = 51
     •                            Several cycles have
     •                            been skipped.
     •

    KOP AGENDA                    The Queue has
40 CALLBK STAT .L 70 LESS S 5     shrunk to 4, but a
41 CALLBK STAT .L 70 LESS S 5     new arrival is about
45 CALLBK STAT .L 70 LESS S 5     to be processed.
47 CALLBK STAT .L 70 LESS S 5
57 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
68 CALLAK SERVIC .L 40
91 CALLAK SERVIC .L 40
93 CALLAK SERVIC .L 40
TIME = 57

    KOP AGENDA
40 CALLBK STAT .L 70 LESS S 5
41 CALLBK STAT .L 70 LESS S 5
45 CALLBK STAT .L 70 LESS S 5
47 CALLBK STAT .L 70 LESS S 5
57 CALLAK
61 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
68 CALLAK SERVIC .L 40
91 CALLAK SERVIC .L 40
93 CALLAK SERVIC .L 40
TIME = 57

    KOP AGENDA                    The Queue builds up
40 CALLBK STAT .L 70 LESS S 5     again. This pattern
41 CALLBK STAT .L 70 LESS S 5     will continue
45 CALLBK STAT .L 70 LESS S 5     indefinitely.
47 CALLBK STAT .L 70 LESS S 5
57 CALLBK STAT .L 70 LESS S 5
61 CALLAK ARRIVE
62 CALLAK SERVIC .L 40
63 CALLAK SERVIC .L 40
68 CALLAK SERVIC .L 40
91 CALLAK SERVIC .L 40
93 CALLAK SERVIC .L 40
TIME = 61
```

第 5 図 AGENDA の変遷（2）

あ と が き

本文中 PACTOLUS および OPS-3 によるプログラム例は，参考文献 3) および 5) より採録したものである．

なお本文の内容については，穂坂衛東大教授ならびに鉄道技研大野豊氏に貴重な参考意見を賜わった．ここに謝意を表したい．

参 考 文 献

(1) Daniel Teichroew and John Francis Lubin: Computer Simulation—Discussion of the Technique and Comparison of Languages. Com. of the ACM, Vol. 9, No. 10, Oct. 1966. pp. 723～741.

(2) John J. Clancy and Mark S. Fineberg: Digital Simulation Lauguages, A Critique and a Guide. AFIPS Conference Proceedings, Vol. 27, Part 1, 1965. pp. 23～36.

(3) Robert D. Brennan and Harlan Sano: "PACTOLUS"—A Digital Analogue Simulator Program for the IBM 1620. AFIPS Conference Proceedings, Vol. 26, 1964. pp. 299～312.

(4) Martin Greenberger, Malcolm M. Jones, James H. Morris, Jr. and David N. Ness: On-Line Computation and Simulation. The M.I.T. Press. 1965.

(5) Martin Greenberger and Malcolm M. Jones: On-line Simulation in the Ops System. Proceedings—A.C.M. National Conference, 1966. pp. 131～138.

（昭和 42 年 8 月 31 日受付）