SE 14-1
(1976. 3. 26)

コンピュータ・システム シミュレーション
(システム・シミュレーター)

水谷 浩二 (日本アイ・ビー・エム株式会社)

1. はじめに

コンピュータ・システムの性能を評価する方法、あるいは、デザインに対するパフォーマンス資料の作成方法を分類すると、

(1) ベンチマーキング (実システム測定)
(2) シミュレーション
(3) モニタリング (ソフトウェア、あるいは、ハードウェアによるもの)
(4) 解析モデル (例えば、待理論を基礎としたもの)

などになる。

上記、各々の方法は、対象となるシステムの複雑性、及びそのシステムのおかれている段階、(例えば、設計中なのか、すでに本番稼働しているかなど)、目的としているシステムの性能評価の重要度、詳細要求度、期間、コストに応じ、方法の選択、あるいは、組合せを行うのが一般的である。

本稿では、上記のうち、シミュレーション技法の一例として、NASA のアポロ計画に付随するシステム開発の際に用いられたコンピュータ・シミュレーションを基礎として汎用化され、OS/MFT/MVT から OS/VS1, VS2, 及び DB/DC パッケージとしての IMS/CICS のシミュレーション用プログラムとして開発されるにいたった システム・シミュレーターについて紹介する。

2. システム・シミュレーターとは

システム・シミュレーター (以下 SS と略す) は、CSS (注) をシミュレーション言語として使用した汎用シミュレーション・プログラムといえる。

ベースとなっている CSS は、S/360, S/370 のアーキテクチャーがそうである様に、インタラプション・ストラクチャーを備えたコンピュータ・システム専用のシミュレーション言語で、一般的な事象にではなく、コンピュータ・システムを対象事象として開発されている。即ち、コンピュータ・システムのハードウェア構成要素各々を代表するマクロ言語が定義され、内部ロジック及び、ハードウェアの基礎的特性を言語機能として備えている。従って、使用者は、ハードウェアに相等するモデリングは必要ない。

(後述「CSS について」参照)

しかしながら、CSS によるシミュレーションを行う場合、ソフトウェア機能、(即ち、OS, DB/DC パッケージ, 使用者アプリケーション・プログラム) に関しては、使用者がそのロジックに基づきモデルを作らなければならない。近年の OS の機能拡大

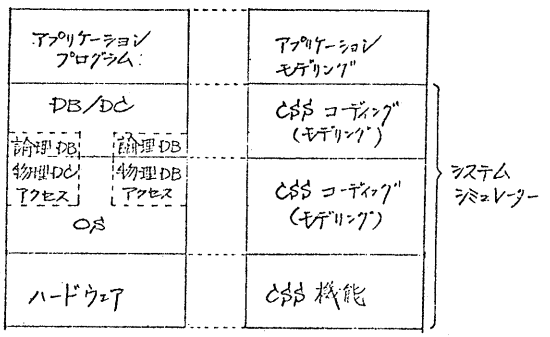


図1. 実システム モジュラ構成 と システム シミュレーター との対比

(注) CSS とは、
COMPUTER SYSTEM SIMULATION 言語

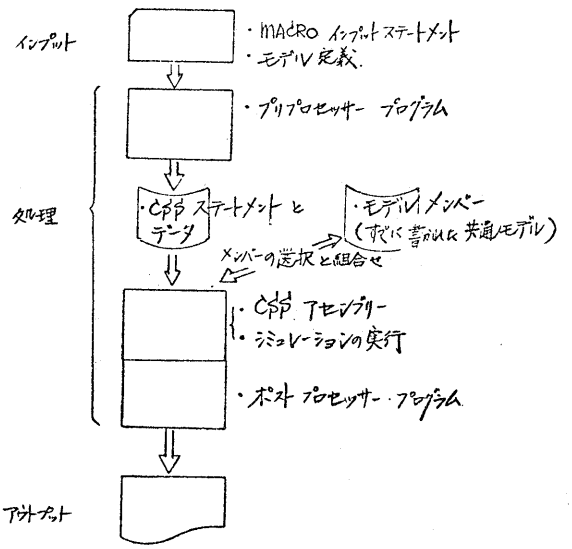
傾向、新しいアクセスメソッド，DB/DCパッケージの機能の増加，とオンライン・システムに代表されるトータルシステムを想定しての シミュレーションを行う場合，使用者にとって，正確なモデリングは不可能に近い状況になってきた。一方，トータルシステムを想定しての システム・シミュレーションの必要性が高まれば，高まるほど，このソフトウェア機能のモデリングに対する問題は注目をあつめ，現実のOSや，DB/DCパッケージがそうである様に、ソフトウェア・モデリングに対しても、共通システムのモジュール化、オンライン システム シミュレーションを想定して，使用者の立場に立ったマクロ・インターフェイスをもつ ハイ・レベルのシミュレーション・プログラムが要望されてきた。その結果として，プリ・プロセッサー，ポスト プロセッサーを備え，GPSS言語を知らずにモデリングのできる システム・シミュレーターが生まれてきた。

システム・シミュレーター（SS）の実行ステップの概念図は下図の様になる。本稿では、このSSについて、IBM OS/VS1のモデルを中心にのべる。(注)

3. SSの構成

前述の様に、SSは、ハードウェア・モデル，システム・コントロール・プログラム（SCP），DB/DCパッケージ，に対応するソフトウェア・モデルを提供する。
使用者は、シミュレートしようとしているシステムに就って，各機能に対応するマクロの指定，必要モデルの定義，アプリケーション・プログラムのロジックをモデリングする事になる。これらが、SSに対してはインプットとなる。

図2 システム・シミュレーター 実行ステップ



SSの構成は 図-3 の様に大きくモジュール化されている。
各モジュールは、さらに、機能別にセグメント化されており、SSの実行の際には，使用者の指定に従い，必要なセグメント・モジュールをOSのライブラリーのメンバーとして選択できるので，部分的な変更に対しても，効率よい実行が可能である。

以下，実際のシステム（Real World）と比較しながらSSのインプット指定についてのべる。

(注). SSには現在，OS/VS1 SS，OS/VS2 SS，IMS SS，CICS SS がある。

SS モデル		ユーザー モデル（定義）		
ハードウェア	ソフトウェア	アプリケーション	システム 負荷定義	環境定義
CPU	システム コントロール プログラム (注)	トレース モデル	バッチ モデル	ターミナル動作
チャネル & 周辺入出力装置	DB/DC モデル (注) • プログラム • データセット • データマネジメント • タスクマネジメント • SVC	ユーザー モデル • プログラム • データセット • データマネジメント定義 • タスクマネジメント定義 • I/O 定義	ネットワーク 負荷 TP メッセージ	
ネットワーク, TPデバイス				

図3. SSの構成

実システムとしてのハードウェア構成（コンフィギュレーション）は，例えば図4の様なものを想定していただきたい。

4. SSのインプット.

実システムで，必要なハードウェア構成，使用されるソフトウェア（SCP コンポーネント）の選択，機能の指定を システム ジェネレーション時に行う様に，SSの場合も，実システムに合わせて，選択，指定するのが原則である。基本的なちがいは，シミュレーションが要求されている時点（例えば，オンライン ピーク時）に必要な システム構成のみを指定する事である。

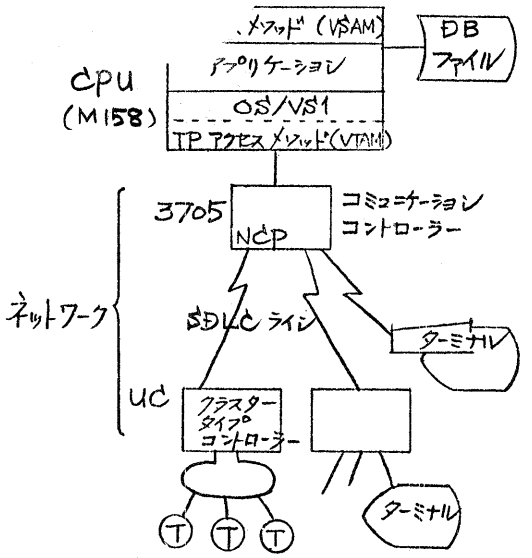


図4. コンフィギュレーション例

4.1 CPUの定義.

CPUの定義は，大きく分けて，ハードウェア，ソフトウェアの定義になる。（前述の様に，ハードウェアの各装置の基本的特性は，モデルの中に組み込まれている。

CPU ステートメント例.

CPU	TYPE = 158	CPU タイプ
	CORESIZE = 1024K	メイン ストレージ サイズ
	NUCLEUS = 100K	ニュークリアス サイズ
	SYSQUE = 16K	SQA サイズ
	LPA =	パージェアブル リンクパック エリア
	SYSRES = (3330, 160)	ユニット・タイプ・アドレス

ストレージ

(注), OS, SCPのモデル作成の際には，実際の OS に対し，ソフトウェアの フックをかけ，タイミング 抽出を行い その結果をモデルに組み込んでいる。

```
CPU   SYSPAGE=(3330,161,162)   ページングデータセット
      OS=VS1                  オペレーティングシステム
      TP=VTAM,BTAM            TPアクセスメソッド   } ソフトウェア
      DBDC=CICS1.1            パッケージの使用の場合
      VTAMBUF=                VTAMバッファーサイズの指定
```

上記以外の3330(DASD)や周辺入出力装置は、ラミコレーションに必要なものだけ、JOBステートメント(後述)のデータセット定義時に指定する。

4.2. ネットワークの定義

ネットワークの構成も、CPUの外側(即ち、使用されるコミュニケーション・コントローラー、ライン、クラスター・コントローラー、ターミナル)の順に従って指定していけば良い。(但し、VTAMを使用してのネットワークでは、論理的な接続であるSE[illegible]リケーションプログラムと論理ユニットの接続)について指定する必[illegible]

基本的に、これらの[illegible]ーは、ハードウェア構成上、あるいはソフトウェア上のパフォーマ[illegible]要するパラメーターといえる。

以下、ネットワーク定義ステートメントの例を概略記述する。

```
3705      CONPROG=     : 3705のコントロールプログラム定義(NCPレベル)
          CHDELAY=     : アテンションインタラプト遅延指定
          BUFSPACE=    : バッファー・スペース/1転送(MAX CHARACTERS)
          ADAPTER=     : ハードウェアアダプタータイプの指定(TYPE1/TYPE2)
          CHANNEL=     : 接続チャネルタイプの指定
          SCANNER=     : 回線走査機構の指定

LINE      NUMBER=      : ライン数の指定(もし同一タイプのラインが複数個の場合まとめて指定可能)
          CHARSEC=     : ラインスピード(C.P.S.)
          LINTYPE=     : ラインタイプの指定
          PROPTIME=    : 伝播遅延時間の指定
          TATIMES=     : ライン・タンアラウンド時間の指定
          CONNECT=     : ラインが接続される相手の指定(どの3705か?)

TERMINAL  TYPE=        : ターミナル・タイプ
          APPL=        : ターミナルが会話するアプリケーションの指定
          RESPONSE=    : リスポンスに対する統計の必要性,(EACHか,GROUPか)
          CLUSTER=     : クラスタータイプの指定
          CHANNEL=     : 3705を経由ではなくチャネルに直接接続される場合のチャネルタイプ
          CONNECT=     : ターミナルが接続される相手方の指定(ライン)
          MSGRATE=     : nメージ到着/HOUR,(到着率)の分布指定(注)
          INTERARR=    : 到着間隔の指定
          MSGMIX=      : 全体のメッセージ発生に対する比率,
```

(注) FUNCTIONについては次ページ注参照

VTAM における ネットワークでは、クラスタータイプの ターミナルは、複数個の論理ユニット（複数個の END USER）を備え、各々は、独立に アプリケーション プログラムと メッセージの やりとりができる。SP に於いても、この指定を UC マクロと SESSION マクロで行う。

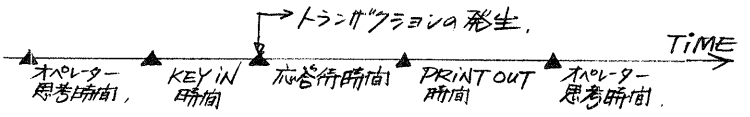
UC	TYPE=	: UC タイプ （3601, 3651, 3791）
	FRAMENUM=	: リスポンスを待たずに(SDLC) アウトプットを うける数 (NCP に於ける MAXOUT)
	PASSCNT=	: マルチドロップラインに於ける トランスミッションユニット数 (PASSLIMIT)
	RESPONSE=	: リスポンスに対する統計の必要性，メッセージ単位の統計 UC単位に統計をとる，全体として統計をとる，などの指定ができる。
	CONNECT=	: UC が 接続される先．

SESSION	TYPE=	: セッションタイプ （セッションタイプは 次の アクティビティに 指定ともう）
	APPL=	: アプリケーション名
	CONNECT=	: 接続先
	MSGRATE=	: メッセージの 発生量，発生分布（注），(指数，一定，ユーザー指定など)
	INTERARR=	: 到着間隔
	MSGMIX=	: 全体に 於ける メッセージの 発生比率．
	SESSQ=	: 次の メッセージを このセッションでは 連続的に 出すか 否か？

4.3 ネットワーク負荷定義／ターミナル動作定義

TP システムにおける インプット メッセージ及び アウトプット メッセージの属性の定義，ターミナルに 於ける オペレータ の動作を定義する。メッセージの発生は、MSGIN，MSGOUT ステートメントにより，メッセージタイプ別に ランダムに発生させる事ができる。ターミナル動作の指定には，ハードウェアの I/O 速度の 外に，応答待時間，キーインまでの オペレーター 必要思考時間，(例えば，伝票確認時間)，キーイン時間，などの オペレーション順序に従って，また オペレーションタイプ（データエントリー，照会型，会話型），を考慮して指定する。

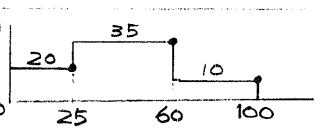
ターミナルに 於ける 動作例，



MSGIN	LENGTH=	: 固定長，もしくは 可変長の インプット メッセージの 数を指定する（FUNCTION ステートメント）
	SLENGTH=	: セグメントの 長さを 指定する．(UC タイプの場合 256) （注）
	TYPE=	: メッセージタイプの 指定．
	PERCENT=	: 複数個の メッセージタイプがあり，全体の MSGIN における パーセント を 指定する．
	BATCH=	: バッチタイプの メッセージである事を指定．
	INQUIRY=	: インクワイリ・タイプの メッセージである事を指定（次の メッセージは 必ず リスポンスの後に 送られる）
	THINK=	: キーイン・タイム と オペレーター 思考時間．
	APPL=	: 試験用に 対応する アプリケーション を 指定する
	MSGRATE=	: メッセージの 発生が 指数分布，一定分布，あるいは ユーザ指定分布に 従うかの指定

(注) FUNCTION ステートメント： DISCRETE（不連続分布） CONTINUOUS（連続）の 2種がある。

右図例では， FUNCTION D3, 25/20, 60/35, 100/10 となる



MSGIN　INTERARR = メッセージの発生間隔が，指数分布，一定分布，ユザー指定分布かの指定（FUNCTION）
RESP = ：実システムがVTAMである場合，レスポンスの要求の選択可．

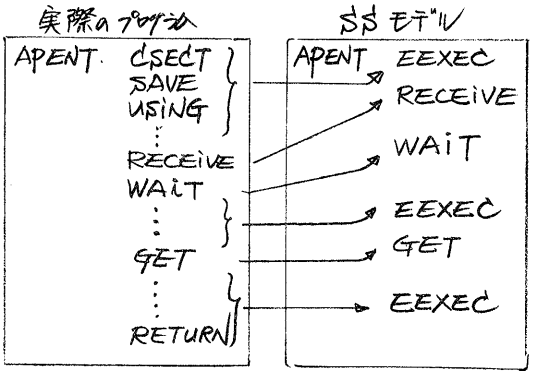
MSGOUT　LENGTH = ：固定長，もしくは可変長のアウトプットメッセージを指定．
SLENGTH = ：セグメントの長さを指定．
TYPE = ：メッセージタイプの指定
REPLY = ：インクワイリイタイプのメッセージアウトプットに対応．

4.4. アプリケーション・プログラム モデリング

SSにおいて，アプリケーション・プログラムをモデル化するには，トレース方法とセルフ方法の二種類があるが，本稿では，セルフ方法について記述する。(注
実際のアプリケーション・プログラムは，アプリケーション・ロジックと，コントロールプログラムに対する機能要求，（即ち，I/O要求など），及び，使用データ・セットの定義から構成される。モデルと，これらを対応させると次の様になる。

実システム	SS モデル（CSS）
・アプリケーション・ロジック ----------	CSS命令
・アプリケーション・実行時間（インストラクション数）	EEXEC命令（EEXECI命令）
・データ・マネジメント・マクロ ----------	SS・データマネジメント・マクロ，
・タスク・マネジメント・マクロ ----------	SS・タスクマネジメント・マクロ（SVC）
・データ・セット定義（DCB，ACB）----------	SS．DDステートメント
・ロード・モジュール定義 ----------	SS．プログラム・ステートメント

基本的に，アプリケーション・プログラムのモデル化においては，どの様な機能を行うかではなく，どの様な処理形態で実行されるかが重要である事は言うまでもない。（即ち，アプリケーション・プログラムのステップ数とその間に入る入出力要求の順序，アクセス回数などが重要である。（下図参照）



(1). EEXEC命令（注）
EEXEC命令には，実行時間を指定する方法と，インストラクションステップ数を指定する方法とがある。（EXECI命令）
どちらも，I/O要求や，SVCマクロの間にあるインストラクション数，もしくは，平均実行時間を算定し指定する。

(注1)，トレース方式というのは，SPAというソフトウェアモニタで実際のプログラムをトレースし，テープに記録したものをインプット負荷とするもの。
(注2)，EEXECとはEnableな状態でのCPU実行時間である。DISABLEな状態（スーパーバイザーモード）でのCPU実行時間は，DEXEC命令でシミュレートする。

EEXEC I あるいは EEXEC	INSTRUCTIONS, TIME,	{ PAGE/NOPAGE PFUNC/ PCONT/PNOTE PREF./PLIST }	アプリケーション ロジック ステップ数 あるいは、実行時間、ページングの 有無、ページング パターン などの指定。

EXEC命令でな、また、どの様なページ参照パターンで、実行されるかを指定し、(FUNCTION ステートメント など) ページング オペレーションをシミュレートする事も可能である。この機能は、ユーザーが指定するとも、SSがもつパターンを利用する事もできる。(OS/VS1 の場合には、2K単位に ページ番号をつけ、コントロールされる。)

(2). タスク・マネージメント・マクロ

タスク・マネージメントに関連するスーパーバイザー機能のうち、SSで提供するものは、右図の様なものである。実際のアプリケーションで使用する主要な、SVC をモデル化している。これらの機能をシミュレートする際も実システムで、これらのSVC マクロを出すときに必要とされるオペランドとして必要な情報は、SSでも必要とされる。

タスク	ATTATCH, DETATCH, CHAP ENQ, DEQ, POST, WAIT, LINK, LOAD, DELETE, XCTL, EXIT
ストレージ	GETMAIN, FREEMAIN
タイマー	STIMER, TIME, TTIMER
I/O	EXCP

SS スーパーバーザー マクロ. (VS1) 図5

(3). データ・マネージメント・マクロ.

データセットの OPEN/CLOSE I/O オペレーションの実行など、実システムでおこなわれるのと同じ様な、ユーザーインターフェイスがSSの場合にも提供される。従って、データ・セットの DASD上の配置、プログラムの中でだす I/O マクロの種類、順序など、実際のデータセット(データ・ベース)の設計や詳細な情報の指定が可能である。(SSでは、実システムにおける エラー処理機能は備えていない。)
図6. は、VSAM に於ける、データ・セットの定義と、I/O マクロの例である。

4.5. JOBストリームの定義

実システムにおける、JCLに対応して、また、データマネジメントマクロ要求時に定義する、DCB/ACB/DD に対応して、SSにおいても、この指定を行う事により初期設定(イニシャリゼーション)が行なわれる。
指定すべき、ステートメントは次の4つである。

```
ESDS1   ACB   ACSM=VSAM,UNIT=(3330,130)
                SPACE=    ,STTRACK=
                CISZ=     ,BUFND=
                RECSZ=    ,SIZE=
                DSTYPE=ESDS
*
OPEN    OPEN ESDS1
          :
        POINT ESDS1
        GET   ESDS1,SVRPL
          :
        PUT   ESDS1,SVRPL
          :
SVRPL   EQU   SV
```

図6. VSAM ESDS の例.

• SYSDS ステートメント.	• システム データセットの定義 SVCLIB/LINKLIB/JOBQE/SYSPAGE
• JOB ステートメント	• ジョブ名, クラス, パーティション などの定義.
• EXEC ステートメント	• ジョブステップの定義. エントリーポイント, ストレージ サイズ
• DD ステートメント	• アクセスメソッド, サイズ, 配置 などの指定, (各々のデータセットに必要)

DCB と DD ステートメントの関係は, 実際の OS と同じに考えて良い。

4.6. モデルコントロール(シミュレーション・コントロール)の定義.

最後に, 実際のシステムにはないが, SS を実行するに必要なパラメーター定義のステートメントの説明をする。SS のプリプロセッサーだけでなく, ポストプロセッサーに対するオプションの指定も行う。
イニシャリゼーション時間, モデル実行時間, シミュレーション結果のオプションの選択(ポスト・プロセッサーに対するアウトプットの選択)
例えば下記の様なものである。

```
MDLCTL   MODTIME=300 ....  5分間実行.
         INITTIME=30 ....  30秒間 イニシャリゼーション (累積リセット)
         REALTIME=30 ....  30分間, 実時間をすぎれば 終了
         PRINTIME=60 ....  1分間に ポスト プロセッサーにコントロールがわたり統計
         PRINTRSP=GRAPH ..  をとる。応答時間は グラフでも OUTPUT する。
         TRACE=100 ....  最後から 100回の ブランチポイント を トレース アウトする
```

5. CSS について,

以上のべてきたのが, SS のインプット である。このインプット指定に従い, SS のプリ・プロセッサーは CSS ステートメントに翻訳し, CSS の実行を行う。

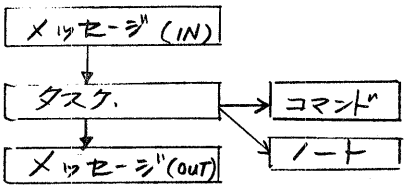
(1), 実システムと CSS との対応,

前にものべたが, CSS はハードウェア機能, (即ち、ディスク装置のメカニズム, 転送速度, 回転速度 等々の基礎的特性) を備え, その他に, インタラプション メカニズム, タイミング をもつ特殊な言語である。一般的なシミュレーション機能としては、クロック, Queue, 乱数発生, 統計用テーブルをもつ。オペレーティング システム との比較の上で重要なものは、一時的属性物 (トランジェント エンティティ) として, コンピュータ システム 所有な, メッセージ, タスク コマンド, ノート を備え, ユーザーに提供している。
実システムとの属性としての比較は 右の様になる。

実システムの属性	: CSS の属性
ハードウェア	ハードウェア定義, (Equipment)
ストレージ	RESOURCE
TCB	TASK
I/O コマンド	COMMAND
TCB Chain	TASK QUEUE
ECB	SVC
RB	DOUBLET.
CDE	NOTE
External インタラプト	iTIMER

(2) CSS プログラム構造（OSの動きの例）

OSとユーザー・プログラムの動きをシミュレートする例を記述する。OSの例として、インタラプトハンドラーと、スケジューラーが、ユーザー・プログラムをコントロールする例である。



CSS トランジェント・エンティティ

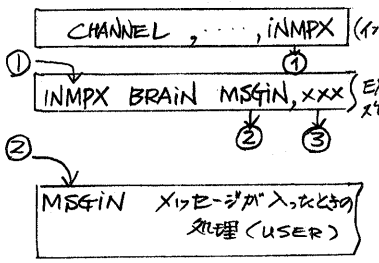
• インタラプト・ハンドラー

主に、I/Oインタラプトとタイマー・インタラプトをシミュレートする部分である。CSSでは、I/Oインタラプトに対してBRAIN（Branch on Interrupt Type）という命令を提供し、インタラプションの種類を解析し、各々のユーザー・イグジットにコントロールをわたしてくれる。

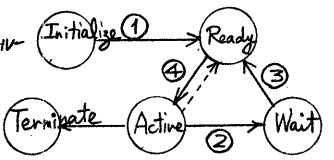
このBRAIN命令は、ディスク、テープのI/Oにもコミュニケーション・コントローラーのI/Oに対しても使用できる。タイマー・インタラプトは、ITIMER命令でインタラプトが生じる時刻とEXITルーチンを指定する事ができ、時刻に達すると指定EXITにコントロールを渡してくれる。

• スケジューラー

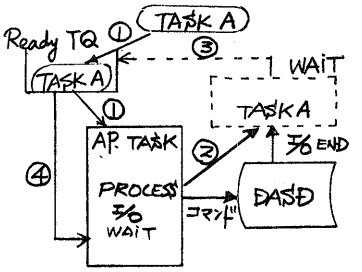
OSのタスク・ディスパッチャーに相当する部分で、インタラプションの処理終了後、あるいは、WAITマクロが出された後に、このスケジューラー・ルーチンにコントロールが渡される。スケジューラーでは、Ready Taskの有無、優先順位をチェックし、その高い順にコントロールを渡す。



インタラプト・ハンドラー



OSにおけるタスク・ステータス



CSSにおけるタスト・ステータス

6. SSのアウトプット

最も大切な事であるが、SSは、その大量のアウトプットを使用者の指定に従って、アウトプットする。使用者にとって、重要な事は、必要とする統計の選択と、要求する測定点を指定する事である。

ポスト・プロセッサーは、要求された統計を選択しアウトプットする。

(1)、リソース使用率、

CPU，入出力装置，コントロールユニット，チャネル，アプリケーションプログラム，SVCロードモジュール

TOTAL RESPONSE TIME IN MILLISECONDS FOR 3791.INQU

RANGE	0 ---------- 10 ---------- 20 ---------- 30 ---------- 40 ---------- 5
0	|
250	|
500	|
750	|***
1000	|******
1250	|**********
1500	|**********
1750	|**********
2000	|***
2250	|**********
2500	|**************
2750	|***
3000	|**********
3250	|******
3500	|******
3750	|***
4000	|
4250	|
4500	|***
4750	|
5000	|***

REMAINING FREQUENCIES ARE ALL ZERO

などの使用率、使用回数などを各々アウトプットする
(2). CPU タイム分析.
ロード・モジュール毎、ユーザー・タスク毎、プログラム機能別に DISABLE/ENABLE に分け、CPU タイムを分析アウトプットする。
(3). Queue 統計.
コマンド、タスク、メッセージ、各々の Queue に対して、平均、最大のキューの状況、待時間、をアウトプットする
(4). 応答時間
ターミナル、ライン、コントロールプログラム、等々の指定(ネットワークの指定)に従って、平均、パーセンタイル、標準偏差、及び分布を、使用者の指定に基づきアウトプットする。(グラフのアウトプットも可能である ···· 第ページ図参照)

7. おわりに.

以上のべてきた様に、PS は、その使用者が、シミュレーションの専門化である必要がないという前提にたち、汎用性のあるプリ・プロセッサー、ポスト・プロセッサーを備え、使用可能になっている。

PS の開発時の精度目標は、±10%の誤差範囲をめざし、またその範囲内での検証例も報告されている。しかしながら、シミュレーションを行う必要のある時点は、システム設計の初期段階が多く、PS へのインプットと詳細設計後、あるいは、完成後とは、異なるのが通例で、検証例はまだ数多くない。
また、実際のシステムのシステム・ライフの検証においては、その実システムをハードウェア・モニタ、及び、ソフトウェア・モニタで、データをとり、これをシミュレーションの中に組み込み、予測する事も多い。この場合には、かなりの作業量を要し、PS そのものの精度だけでなく、ユーザーのモデルの精度を、現実のプログラムに即し、上げる必要があり、やはり相当の作業量となる。この分野では、最近は、ソフトウェアによる、ネットワーク・アクティビティ・シミュレーターにより、コミュニケーション・コントローラーの外から、メッセージを発生させる方法が本格化してきた。

システム性能評価本来の目的である
- システムボトルネックの発見、
- システム限界の予測(確認)
- チューニング要素の定量化

等々の方法は、その目的意識を明確化する事、システムのおかれている環境を理解し、コストを考慮した上で、シミュレーションか、実測か、あるいは他の方法かを判断する事が重要である。

なお、本稿で、紹介したシステム・シミュレータ・プログラム、および、関連マニュアルは、IBM 社外には、提供できないこと、ユーザーのシステム設計に利用する場合には、IBM 社員によって使用され、その結果のみをユーザーに提供する事を、お断り申し上げておく。本稿の内容については、コンピューター・シミュレーションの一つの考え方と実際例として、ご理解いただければ幸いである。
(今回の記述は、VSPS Version 2 に従って記述した。また紙面の都合上、DB/DC パッケージに関する記述は省略した。)