システム記述評価システム：SDES†

伊藤 潔†† 田畑孝一††† 大野 豊†††

情報システムのソフトウェアの機能と性能を効果的に評価するシステム記述評価システム(SDES)を京都大学大型計算機センタの FACOM M-190 で開発した．

情報システムのソフトウェアの評価活動における困難さの多くは，評価活動がソフトウェアの製作活動と分離していることに帰因していると考えられる．SDES では，これらの活動を統合している．

SDES では，システム内に存在する，互いに逆の関係にある2つのタイプの実体——処理の主体となっている処理エンティティと，処理を受けている被処理エンティティ——に着目する．SDES では，まず，対象システムが処理エンティティ（プロセス）の側から作成される．次に，この記述の上に，被処理エンティティの振舞を，対象システムの評価のための記述としてオーバラップして与える．被処理エンティティを表わしその振舞をトレースするユニットをトラバーサと呼び，被処理エンティティの振舞の記述は，このトラバーサの経路・属性などの指定をすることにより行われる．以上の，互いに逆の関係にある2つの概念から，統合された記述を作成する方法を，双対プログラミングと呼ぶ．この統合された記述を作成し，その後，この記述を SDES 評価システムで実行し，機能と性能に関する各種の評価データを収集する方法をトラバーシング法と呼ぶ．

1. まえがき

情報システムのソフトウェアの製作の際には，その設計からプログラミングに至るあらゆる段階で，製作者は，製作中の対象システムが要求された機能を有していることと性能目標を満たしていることを，絶えず確かめながら製作活動を進めることが必要である．これは，対象システムの製作を迅速かつ円滑に進め，製作された対象システムを信頼性の高いシステムにする上で，必要不可欠な活動である．

情報システムのソフトウェアは，通常，多数の，並行プロセス[3)~5),7)]と呼ばれる並行処理モジュールから構成される．各並行プロセスは，システムの中で，他の並行プロセスと並行的に実行可能な自律的な実体であり，必要に応じて相互に関連・干渉し合う実体である．この並行プロセスから構成される対象システムの機能を評価する活動，すなわち，各並行プロセスの論理的振舞（処理手順）や，相互の論理的関係（同期・通信・待合せ・競合）の正しさを確かめる活動と，対象システムの性能を評価する活動，すなわち，対象システムが実環境の下で稼動する際の，各並行プロセスの時間的振舞（処理時間）や，相互の時間的関係（待合せ時間）を予測し，性能目標を満たしているかどうかを確かめる活動は，ソフトウェアの製作の早い段階から最終的な段階までのあらゆる段階で行われる必要がある．

製作進行中にシステムのソフトウェアの機能と性能を評価する活動には，一般に，あまり，組織的な方法が採られておらず，しかも，多くの時間や手数を要することが多い．

機能の評価のために，通常，机上テストやプログラムの検証理論に基づくアサーションテスト[8)]などの静的なテストや，トレーサなどを用いる動的なテストが行われる．対象システムが多数の並行プロセスから構成されているために，机上テストの際に，ソフトウェア自体が読みにくく，多くの時間が費されたり，データ値の範囲や変数間の関係の不変条件をアサーションとして指定し，この不変条件の成立を調べることに困難が多い．また，多数の並行プロセスからはトレース出力が交錯するために，動的なテスト出力の解析が困難となることも多い．

性能の評価のために，通常，待ち行列モデルを用いた解析や，シミュレーションモデルによるシミュレーションなどが行われる．これらのモデルは，対象システムの有する機能を抽出し，性能評価のために必要な情報を付加して作成される．モデルの作成の際には，対象システムの製作に用いられる言語表現とは異なる言語表現が用いられ，製作自体とは切離されて，モデルが作成されるために，かなりの時間や手数を要する．その上に，モデル作成時に書き誤りを生じたり，製作中の対象システムとモデルの詳細さの程度が異なるお

† System Description and Evaluation System: SDES by KIYOSHI ITOH, KOICHI TABATA, and YUTAKA OHNO (Department of Information Science, Faculty of Engineering, Kyoto University).

†† 現在，上智大学理工学部機械工学科
††† 京都大学工学部情報工学科

それがある．この場合，モデルによる評価の結果は，対象システムの実際の振舞を正しく反映しない．

著者らは，情報システムのソフトウェアの製作をその評価に結びつけて，製作を効率的に進めるために，対象システムのソフトウェア自体の記述と対象システムの機能と性能を評価するための記述とを一体化し，この一体化された記述をコンピュータの援用により，実行し評価データを収集する，システム記述評価システム SDES (System Description and Evaluation System) を開発した．この SDES では，まず，PL/I を用いて対象システムのソフトウェアが記述され，次に，この記述の上に，対象システムで処理を受ける対象の振舞がオーバラッパして記述される．その後，SDES の評価システムが，対象システムを実行し，処理を受ける対象の振舞の記述を手掛かりとして対象システムの機能と性能を評価する．

PL/I により記述されるソフトウェアは，対象システムの中で処理を遂行する実体，すなわち，並行プロセスの側から，その処理手順と相互の関連や干渉を記述したものである．この記述は，実行可能な形式であれば，記述の詳細さは任意である．このことは，SDES を，対象システムのソフトウェアの製作の早い段階，すなわち，おおまかな製作の段階から，適用可能にしている．

対象システムの中で，処理を受ける対象，すなわち，入力データ・メッセージ・トランザクションなどの，振舞の記述は，それらの対象が生成され消滅するまでにいくつかの並行プロセスを渡り歩く経路の記述と，その経路上での，それらの対象とシステムの中の変数や資源とのインタラクションの記述とから成る．

2. 基本概念

システムの中には，互いに逆の関係にある2つのタイプの実体が存在する．一方は，処理を行っている，すなわち，処理の主体となっている実体であり，他方は，処理を受けている，すなわち，処理の対象となっている実体である．各々，処理エンティティ，被処理エンティティと呼ぶ．処理エンティティは，物理的には，CPU や入出力チャネルであるが，システムのソフトウェアを製作する立場から論理的にとらえるとプロセスに相当する．被処理エンティティの例として，システムへの入力データ，プロセス間で送受されるメッセージ，オンラインシステムへのトランザクションなどが挙げられる．

システムのソフトウェアは，通常，処理エンティティ内の処理手順や処理エンティティ相互の関連・干渉のアルゴリズムを記述したものである．処理エンティティは，システムの中に通常1つ以上存在し，その各各についてアルゴリズムが記述される．

処理エンティティは，他の処理エンティティと並行的に実行可能であり，通常は，与えられた処理手順に従って，その処理エンティティのみに関係する変数や資源に対して，自律的に，順序的処理を行っている．そして，必要に応じて，他の処理エンティティと共有する変数や資源を用いて，同期や通信などの，他の処理エンティティとの関連・干渉を伴う処理を行う．この同期や通信で，被処理エンティティが一方から他方へ移動する．

システムのソフトウェアの機能を評価するには，単に，各処理エンティティの処理手順を調べるばかりでなく，被処理エンティティに着目して，これが処理エンティティ間で正しく送受され，この被処理エンティティに対する一連の処理の流れが実現されていることを調べることも重要である．

処理エンティティは，与えられた処理手順に従って順序的処理を行っている際に，他と共有する変数や資源に対して，他と競合して処理を行うことがある．この場合に，共有する変数や資源に対して，排他的なアクセスが保障されなければならない．処理エンティティは，ある被処理エンティティの処理を行っている際に，共有する変数や資源に対して，他と競合して処理を行うのであるから，被処理エンティティの側から，排他的なアクセスが正しく実現されていることを調べることは有効である．

製作中のシステムを性能評価するために，GPSS[6] などの，トランザクションタイプのシミュレーション言語を用いたシミュレーションを行うことが多い．この場合に，作成されるモデルは，対象システムの中で処理を受ける被処理エンティティの，システムの中での振舞を，トランザクションの振舞として，記述したものである．表1には，対象システムのソフトウェアとそのシミュレーションのためのモデルの作成における記述方法の相違が例示されている．このように，ソフトウェアの作成における記述方法とは逆の立場からモデルを作成することは，多くの時間や手数を要する上に，対象システムのソフトウェアとそのモデルとが正しく一致せずに，モデルによる評価結果がシステムの振舞を正しく反映しないおそれを生ずる（図1上部

表 1 通常のプログラムとシミュレーションプログラムとの違い
Table 1 Difference between usual programs and simulation programs.

Example 1 A process STOREs a message to a memory area. *A message ENTERs a memory area.*
Example 2 A process LOADs a message from a memory area. *A message LEAVEs a memory area.*
Example 3 A process SENDs a message to another process. A process RECEIVEs a message from another process. *A message TRANSFERs from one process to another process.*

Statements in usual programs are written in normal letters. *Statements in simulation programs are written in italic letters.*

参照).

システムが稼動する際の性能を評価するためには，そのシステムの中で実際に処理を受ける被処理エンティティの導入が不可欠である．それゆえ，著者らは，システムの中の処理エンティティの側からの記述であるソフトウェアに，システムの性能を評価するために最小限必要な被処理エンティティの振舞の記述を付け加えて，性能評価を行う方法を考案した．

被処理エンティティを表わし，その振舞をトレースするユニットをトラバーサ (Traverser) と呼ぶ．このトラバーサが表わしている被処理エンティティの，システム中での振舞をトレースする様子についての記述を，処理エンティティの処理手順を記述したソフトウェアの中に与え，この記述を，製作進行中のシステムのソフトウェアの機能と性能を評価する手掛かりとする．トラバーサについての記述を与えられたソフトウェアは，システムの中で互いに逆の関係にある処理エンティティと被処理エンティティという2つの概念を，1つの記述の中に統合したものである．SDES では，SDES 言語を用いて，互いに逆の関係にある2つの概念を統合して，1つの記述を作成する．これを双対プログラミングと呼ぶ．この統合された記述を作成し，その後，その記述をSDES 評価システムで実行し評価データを収集する方法を，トラバーシング法[1),2)]と呼ぶ (図1下部参照)．SDES 言語は，処理エンティティの処理手順を記述するために用いられる PL/I と，被処理エンティティの振舞をトレースするためのトラバーサを記述するために用いられるトラバーサ言語とから構成される．

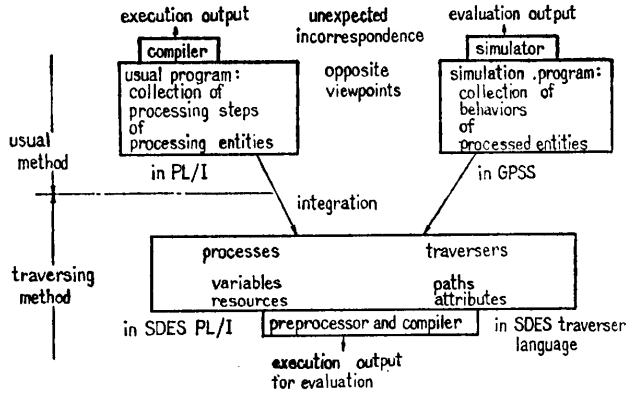


図 1 通常の評価手法とトラバーシング法
Fig. 1 Traversing method versus usual method in evaluation activity.

3. 双対プログラミング

3.1 双対プログラミングの概要

双対プログラミングは，次の5ステップから成る．

〔ステップ 1〕 PL/I により，システムのソフトウェアを製作する．これは，処理エンティティ，すなわち，並行プロセスの処理の記述である．この記述は，実行可能な形式であれば，その詳細さの程度は問わない．

〔ステップ 2〕 システムのソフトウェアの中で，機能と性能を評価する部分を定める．この部分を評価領域と呼ぶ．評価領域は，ソフトウェアの全体であっても一部であってもよい．

〔ステップ 3〕 定められた評価領域の中で，処理を受けている被処理エンティティの種類を数え上げる．各種類ごとにトラバーサを割当て，識別子を付ける．被処理エンティティの振舞の道筋を，トラバーサの経路として定義する．トラバーサの経路は，トラバーサの生成・消滅・移動箇所の指定により定義される．生成箇所は，対応する被処理エンティティの，評価領域の入口，あるいはその生成箇所である．消滅箇所は，逆に，評価領域からの出口，あるいは消滅箇所である．移動箇所は，対応する被処理エンティティがプロセス間の同期・通信処理などで送受される箇所である．

〔ステップ 4〕 システムのソフトウェアの機能を評価するために，各トラバーサの経路上に，トラバーサとソフトウェア内の変数や資源とのインタラクションを記述する．処理エンティティがある被処理エンティティを処理しているとき，ソフトウェア内の変数や資源を使用し，その値や状態に変化を与え

る．この変化の前後の値や状態をトラバーサの属性として定義する．そして，必要に応じて，トラバーサごとに，その属性の満たすべき関係式をアサーションとして与える．また，処理エンティティがある被処理エンティティを処理しているときに，他と共有する変数や資源を排他的に使用する場合，その使用の区間をクリティカルリージョンとして，トラバーサの経路上に定義する．

〔ステップ 5〕 システムのソフトウェアの性能を評価するために，各トラバーサの経路上に，トラバーサとソフトウェア内の資源とのインタラクションを記述する．資源は，3種類のタイプ—ファシリティ，ストレージ，キュー—のいずれかであるとみなせる．ファシリティは，CPU や並行プロセスなどの，容量が1である，すなわち，それを使用できるものが同時刻には1個である資源である．ストレージは，記憶装置やメッセージバッファなどの，容量が1より大である資源である．キューは，ファシリティやストレージのタイプの資源を使用するための待ち行列である．被処理エンティティが処理を受けているときに，これらのタイプのいずれかの資源が使用される場合，その資源の使用区間を，トラバーサの経路上に定義する．また，トラバーサの経路経過の計時区間を定義する．

以上のステップで，双対プログラミングされたソフトウェアは，SDES 評価システムによって実行され，機能と性能が評価される（図2参照）．評価結果が良好ならば，ステップ1に戻ってより詳細な記述を作成したり，ステップ2に戻って評価領域を変更する．評価結果が不良ならば，ステップ1に戻って記述に修正を加える．

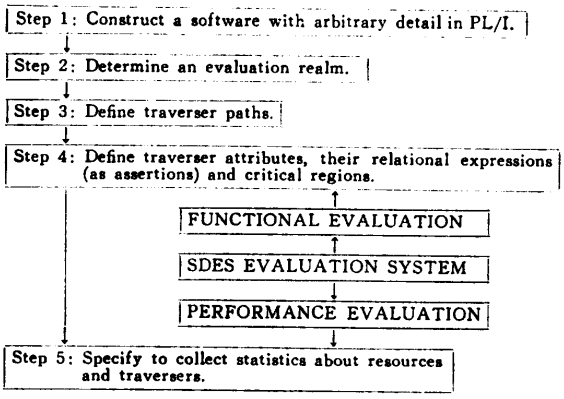


図 2 双対プログラミングのステップと SDES 評価システムの関係
Fig. 2 Relationship of steps in dual programming and SDES evaluation system.

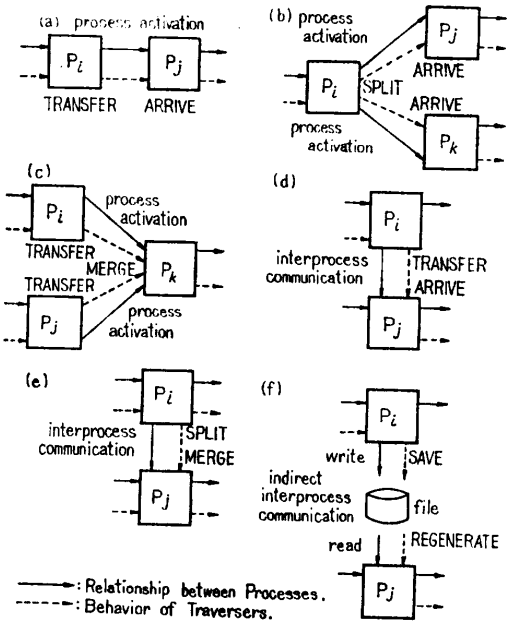


図 3 トラバーサの経路の定義原則
Fig. 3 Principles in specifying traverser paths.

3.2 トラバーサの経路

トラバーサの経路を定義するためのいくつかの原則を述べる．

〔A〕プロセスの起動（図 3(a)-(c) 参照）

〔A-1〕トラバーサの移動： あるプロセスが被処理エンティティを処理している際に，他のプロセスを起動してこの被処理エンティティの処理をそのプロセスに委ねるとき，起動の箇所でトラバーサを移動させる．

〔A-2〕トラバーサの複製： あるプロセスが複数のプロセスを起動して，ある被処理エンティティの処理をこれらのプロセスに分担させるとき，トラバーサを複製して各々のプロセスに移動させる．

〔A-3〕トラバーサの併合： 複数のプロセスの起動により，あるプロセスが実行を開始する場合，それらのプロセスから移動するトラバーサを併合する．

〔B〕プロセス間同期・通信（図 3(d), (e) 参照）

2つのプロセス間で同期を取合ったり，メッセージを送受する場合，一方のプロセスから他方のプロセスへトラバーサが移動すると考える場合と，2つのプロセス各々にトラバーサがあって，同期・通信の際に，一方のプロセスがトラバーサを複製し，これを他方のプロセスへ移動させ，他方がこれを併合すると考える場合がある．

〔C〕プロセス間の間接的同期・通信（図 3(f) 参照）

2つのプロセスがファイルを介してデータの送受を

行うことがある．これは，プロセス間の間接的な同期・通信と考えることができる．読み書きするレコードに特別なフィールドを用意し，このフィールドに，一方がトラバーサに関する情報を記入し，他方がこの情報を使ってトラバーサを再生する．

3.3 SDES言語

SDES言語は，処理エンティティ，すなわち，並行プロセスの処理手順や相互の関連・干渉を記述するPL/Iと，被処理エンティティの振舞をトレースするためのトラバーサを記述するトラバーサ言語から成る．トラバーサ文には"$"が付けられ，PL/I文と区別される．トラバーサ文に付けられるラベルは，トラバーサの移動先を示したり，トラバーサの経路をトレースするために利用される．

トラバーサ文のオペランドには，トラバーサ識別名，ラベル，ファシリティ・ストレージ・キュー識別名，変数・資源識別名などが指定される．

トラバーサ文は，次の5種類に分類される．

(a) 経路指定文：トラバーサの経路を定義する．

(b) 属性指定文：トラバーサの属性を定義する．

(c) 検証指定文：機能評価のために，トラバーサの経路や属性のテストを指定する．

(d) 統計指定文：性能評価のために，統計情報の収集を指定する．

(e) 場合分け文：システム内の変数の値によってトラバーサの経路や属性の指定が異なるときの場合分けに使われる．

表2は，トラバーサ文の意味の一覧である．

表2 トラバーサ文
Table 2 Traverser statements.

path statement

GENERATE	トラバーサの生成
TERMINATE	トラバーサの消滅
TRANSFER	トラバーサの移動
ARRIVE	トラバーサの到着
SPLIT	トラバーサの複製
ASSEMBLE	同種のトラバーサの併合
MERGE	異種のトラバーサの併合
SAVE	トラバーサの保存
REGENERATE	トラバーサの再生

attribute statement

ATTACH	トラバーサ属性の定義
DETACH	トラバーサ属性の解放

verification statement

TRACE	トラバーサのトレースの指示
CHECK	トラバーサの通過の確認
TEST	トラバーサ属性の検証
LOCK	共有資源・変数に対する排他的アクセスの開始
UNLOCK	共有資源・変数に対する排他的アクセスの終了

statistic statement

SEIZE	ファシリティの使用開始
PELEASE	ファシリティの使用終了
ENTER	ストレージの使用開始
LEAVE	ストレージの使用終了
QUEUE	キューの使用開始
DEPART	キューの使用終了
MARK	経路経過時間の計時開始
TABULATE	経路経過時間の計時終了

case statement

CASE	トラバーサの経路と属性の指定の場合分け

4. SDES評価システム

SDES評価システムは，京都大学大型計算機センタのFACOM M-190 (OS IV/F4)上で開発され，TSS端末から使用できる．SDES評価システムのソフトウェアの基本構成を**図4**に示す．ユーザは，エディタを用いて，TSS端末の文字ディスプレイ上で，双対プログラミングを行う．プリプロセッサは，この双対プログラミングされた記述の中のトラバーサ文をPL/I文に変換し，また，PL/I宣言文によって書かれた各種評価用テーブルを作成する．すなわち，プリプロセッサの出力は，PL/Iプログラムである．このPL/Iプログラムは，PL/Iコンパイラによりコンパイルされた後，実行される．この実行中に，機能・性能の評価データが各種テーブルに記録される．評価結果編集プログラムは，これらのテーブルを参照して，評価データを編集し，文字ディスプレイ上に表示する．エディタとPL/Iコンパイラは，M-190のTSSに供給されているものを利用し，プリプロセッサと評価結果編集プログラムを新たにPL/Iにより開発した．

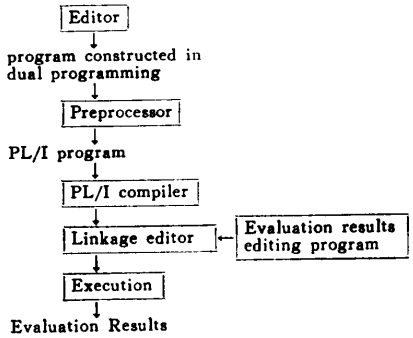


図4 SDESのソフトウェア構成
Fig. 4 Software organization of SDES.

4.1 機能評価手法

トラバーサの経路・属性・状態や，共有変数の状態

を管理し保存するために次のテーブルを用意する：

（a） トラバーサテーブル

（b） プロセス・トラバーサ対応テーブル

（c） トラバーサ行列テーブル

（d） クリティカルリージョンテーブル

（a）のエントリは，トラバーサの識別名・生成番号・現在位置・状態・属性から成る．このエントリは，GENERATE 文が実行されたとき生成され，識別名と生成番号が付けられる．このエントリの消去は，TERMINATE 文が実行されたとき行われる．現在位置は，進行中のトラバーサの記述内での位置を示す．属性は，ATTACH 文を実行したとき定義される．トラバーサの状態には，次の3種類がある（図5参照）．

（i） active 状態：トラバーサの進行があるプロセスの実行に伴われている，すなわち，そのプロセスがそのトラバーサに対応する被処理エンティティを処理中である状態．

（ii） wait 状態：あるトラバーサが他のプロセスから移動してくるトラバーサとの併合待ち，すなわち，そのトラバーサを伴っているプロセスが他のプロセスとの同期・通信待ちである状態．

（iii） free 状態：あるプロセスの実行に伴われて進行していたトラバーサが，移動のため，そのプロセスを離れ他のプロセスに到着するまで，トラバーサとプロセスの対応がなくなっている状態．

（b）のエントリは，プロセスの識別名，トラバーサの識別名・生成番号から成り，プロセスとそのプロセスに伴われて進行しているトラバーサとの対応を示す．このエントリは，GENERATE 文によりトラバーサが生成されたとき，あるいは，ARRIVE 文の箇所に到着しているトラバーサがプロセスと共に進行を開始したときに生成される．このエントリは，トラバーサが active/wait 状態にある間に，保持される．このエントリの消去は，トラバーサが free 状態になったとき，あるいは，TERMINATE 文を実行したときに行われる．

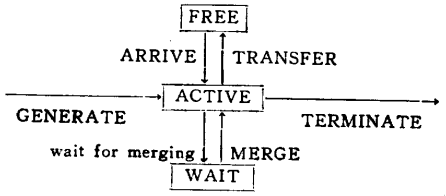


図 5 トラバーサの状態とその状態遷移
Fig. 5 Traverser states and transition.

（c）のエントリの構成は（b）と同じである．このエントリは，あるトラバーサがあるプロセスに移動するとき，移動先のプロセスが別のトラバーサを伴っている場合，そのプロセス自体が実行中でない場合，あるいは，そのプロセスに向けて移動するトラバーサが1つ以上存在する場合に，作られる待ち行列のエントリを示す．

（d）のエントリは，共有変数・共有資源の識別名，トラバーサの識別名・生成番号から成り，共有変数・共有資源が使用中であるか否かを示す．このエントリは，LOCK 文を実行したとき生成され，UNLOCK 文を実行したとき消去される．

経路テストは，あるプロセスの実行中に遭遇するトラバーサ文に指定されているトラバーサ識別名と，そのプロセスに伴われている，すなわち，プロセス・トラバーサ対応テーブルに記録されている，トラバーサの識別名との一致をとることにより行われる．不一致の場合，警告メッセージが出力される．また，TRACE 文で指定されたトラバーサについては，その識別名のトラバーサをオペランドとして有するトラバーサ文に遭遇したとき，その文のラベルをトレース記録としてトラバーサテーブルに記録する．

属性テストは，TEST 文に遭遇したときにその文で指定されているトラバーサの属性に関する関係式が満たされているか否かを，トラバーサテーブルに記録されている属性を参照して調べることにより行われる．

プロセス間で共有される変数・資源への排他的なアクセスのテストは，LOCK 文に遭遇したときに，クリティカルリージョンテーブルを参照して，その変数・資源についてのエントリの存在を調べることにより行われる．もし，そのエントリが，既に存在していれば，同時アクセスが起きたことになり，警告メッセージが出力される．

4.2 性能評価手法

トラバーサの経路・属性・状態を保存する上記の（a），（b），（c），のテーブルのほかに，次のテーブルを用意する：

（e） トラバーサ統計テーブル

（f） 資源統計テーブル

性能評価のために，（c）のトラバーサテーブルには，トラバーサの属性として，トラバーサが MARK 文を実行したときの時刻が記録される．このトラバーサが TABULATE 文を実行したときに，その時刻と記録された時刻との差が求められる．（e）のエントリは，

トラバーサの識別名，経路経過時間の総和・最大値・最小値，その経路を通ったトラバーサの個数から成り，上記の差が求められたときに更新される．(e)には，トラバーサの識別名，生成個数，消滅個数から成るエントリも存在し，これは，トラバーサの生成や消滅ごとに更新される．

(f)のエントリは，資源識別名，資源のタイプ（ファシリティ，ストレージ，または，キュー），その資源を使用したトラバーサの個数，使用時間の総和・最大値・最小値から成り，トラバーサの資源の使用の終了ごとに更新される．

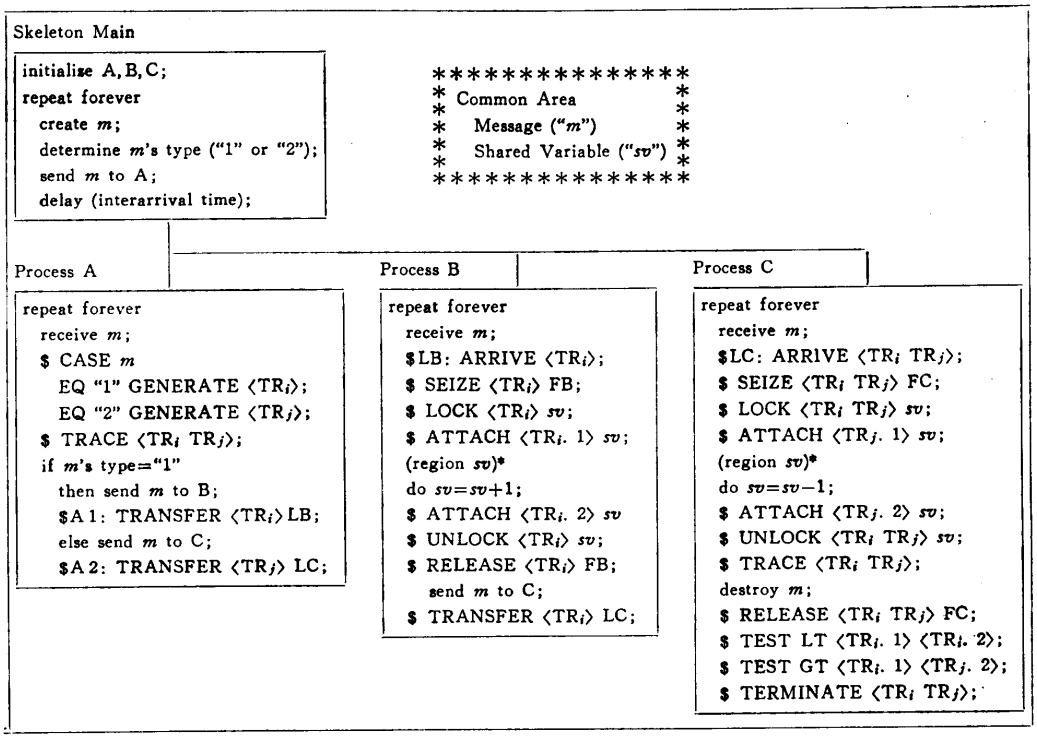
性能評価のために，トラバーサに対しては，その識別名ごとに，生成・消滅個数，経路経過時間の平均値・最大値・最小値などの統計が，資源に対しては，使用したトラバーサの個数，使用時間の平均値・最大値・最小値，使用率（ただし，キューの場合は，この項目はない）などの統計が，報告される．

5. 例　題

図6は，SDES 言語で，あるソフトウェアを双対プログラミングした例を示す．ただし，読みやすくするために，PL/I 文は英文風に書いてある．4つのプロセス，Skeleton Main, Process A, Process B, Process C が存在する．Skeleton Main は，メッセージの発生を模擬するプロセスである．2つの型のメッセージ，type-1 message, type-2 message が存在する．type-1 message は，Process A, Process B, Process C を通る．type-2 message は，Process A, Process C を通る．Process B と Process C は，共有変数 sv を参照していて，排他的なアクセスが行われるようにPL/I で書かれている．

Process A 内の CASE 文は，メッセージのタイプに従って2つのトラバーサ TR_i と TR_j を生成する．TR_i は，Process A, Process B, Process C の順に移動し，TR_j は，Process A, Process C の順に移動する．TR_i, TR_j は，対応するメッセージに対する処理が終了する Process C 内で，終了する．TR_i と TR_j の振舞を調べるために，TRACE 文が Process A と Process C 内に指定される．

共有変数 sv に対する排他的なアクセスを調べるために，LOCK 文と UNLOCK 文が Process B, Process C 内に指定される．sv の値の正しさを調べるために，Process B 内の代入文 $sv=sv+1$ と Process C 内の代入文 $sv=sv-1$ の前後に，各々，TR_i, TR_j の属性



(region sv)* may be missing.

図6 トラバーサ文を付加したソフトウェア
Fig. 6 A software overlapped traverser statements.

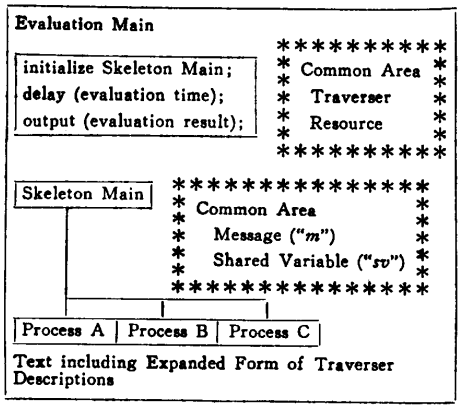


図 7 図6の前処理後の形式
Fig. 7 Preprocessed from of Fig. 6.

```
(a) HISTORY OF TRI 1*)
      POINT A 1     AT 72671396**)
      POINT LC      AT 72671418
(b) **SIMULTANEOUS ACCESS FOR "sv"
    **TRAVERSER IN THE REGION
        TRI 4
    **TRAVERSER ATTEMPTING TO ACCESS
        TRJ 12
(c) HISTORY OF TRJ 24
      POINT A 2     AT 71999363
      POINT LC      AT 71999368
      **MUTUALLY EXCLUSIVE ACCESS FOR "sv"
        IN SAFETY
(d) HISTORY OF TRI 24
      POINT A 1     AT 71997644
      POINT LC      AT 71997674
      **TRI. 1<TRI. 2 IS NOT SATISFIED
        TRI. 1=-3   TRI. 2=-5
(e) **EVALUATION TIME   81685 MS
    **TRAVERSER STATISTICS**
      1 TRAVERSER ID. TRI
          GENERATION COUNT  72
          MEAN FLOW  31900   MAXIMUM FLOW  61904
      2 TRAVERSER ID. TRJ
          GENERATION COUNT  71
          MEAN FLOW  19059   MAXIMUM FLOW  39460
    **FACILITY OR STORAGE STATISTICS**
      1 ID FB   TYPE FACILITY   CAPACITY 1
          NUMBER ENTRIES  17
          AVERAGE CONTENTS  0.6315
          MEAN USE TIME  3034
          MAXIMUM USE TIME  3478
          MINIMUM USE TIME  9
          AVERAGE UTILIZATION  0.6315
      2 ID FC   TYPE FACILITY   CAPACITY 1
          NUMBER ENTRIES  48
          AVERAGE CONTENTS  0.0355
          MEAN USE TIME  60
          MAXIMUM USE TIME  526
          MINIMUM USE TIME  9
          AVERAGE UTILIZATION  0.0355
*): instance number   **): time
```

図 8 評価結果
Fig. 8 Evaluation results.

を ATTACH 文で定義する．各トラバーサの属性の満たすべき関係式を Process C 内の TEST 文で指定する．

Process B と Process C の稼動率を知るために，各プロセスをファシリティ FB, FC とみなす．

図7は，図6を前処理したものを表わす．トラバーサ文はすべて PL/I 文に変換され，評価用の各種テーブルが付加されている．Evaluation Main は，評価結果編集プログラムである．

図8は，評価結果である．(a)は，トラバーサのトレース情報を示す．(c)は，共有変数 *sv* に対する排他的なアクセスが正しく行われたことを示す．(b)と(d)は，共有変数 *sv* に対する排他的なアクセスの指定が PL/I 文により正しく行われていなかったために，同時アクセスが検出され，トラバーサの属性に誤りが生じたことを示す．(e)は，トラバーサとファシリティに関する統計情報である．

6. むすび

情報システムのソフトウェアの製作と評価とを一体化するシステム記述評価システム SDES の概念とそのシステム構成について述べた．今後，さらに種々の情報システムのソフトウェアを SDES により製作・評価して，SDES の有効性を確かめたいと考えている．

参考文献

1) 伊藤，田畑，大野：並行プロセスシステムの機能・性能評価手法，第 18 回情報処理学会全国大会 (1977).
2) Itoh, K., Tabata, K., and Ohno, Y.: An Evaluation System for Concurrent Processes by the Transversing Method, Proc. 3rd UJCC, pp. 41-45 (1978).
3) Hansen, P. B.: Operating System Principles, Prentice-Hall (1973).
4) Hansen, P. B.: The Architecture of Concurrent Programs, Prentice-Hall (1977).
5) Hoare, C. A. R.: Monitors: An Operating System Structuring Concept, Commun. ACM, Vol. 17, No. 10, pp. 549-557 (1974).
6) General Purposed Systems Simulator/360 User's Manual, IBM.
7) Wirth, N.: Modula: A Language for Modular Multiprogramming, Software-Practice and Experience, Vol. 17, pp. 3-35 (1977).
8) Owichi, S. and Gries, D.: Verifying Properties of Parallel Programs: An Axiomatic Approach, Commun. ACM, Vol. 19, No. 5 (1976).

(昭和53年12月6日受付)
(昭和54年3月15日採録)