OSの可視化ツールの設計と実現

5N－7

伊藤能康、早川栄一、並木美太郎、高橋延匡

（東京農工大学　工学部）

1. はじめに

情報工学科の講義で、教育者がスーパバイザコール(SVC)によるOSの起動を説明する場合は、ユーザがSVCを発行した時OS内でどのようなシーケンスによってそれが処理されるのかを、またタスク管理の場合は、タスクを管理するリストをどのように持ち、入れ替えるのかなどを学生に解説する。これらはシーケンスを示す図やリストを図にして説明するが、それだけでは学生は理解しにくい。それはOSの性質と説明の方法、学習の方法の三点で、次の問題があるからである。

(1) OSが非同期に動作している

(2) 連続的な処理を動きのない図として表記し、説明しにくい

(3) 学生が手を動かさないため、理解度が低い

これらの問題の解決するために、OSを可視化して見せ、学生が自分の手を動かして理解させたい。そこで我々の研究室で開発されたOS『礎石』を対象とした可視化ツールの設計を行い、実現した。本報告では、可視化ツールの設計とその実現について述べる。

2. 可視化ツールの目的

OSの可視化ツールを作成し、OSの講義において学生がOSの理解を手助けすることが、本研究の目的である。なお、学生とは情報工学科でコンピュータサイエンスを専門とする大学生である。

3. 可視化ツールへの機能要求

可視化ツールには、1.であげた問題点の解決を含めて、次のような要求がある。

(1) 割込み・例外発生時間を知りたい

割込みやSVCの発行タイミングや、SVCによって起きる処理の流れ、処理時間をタイミングチャートとして見ることができれば、問題点(1)のOSの非同期性についての理解を手助けすることができる。

(2) 割込み・例外発生元を特定したい

(3) 割込み・例外によってリンクなどがどのように変更されたか知りたい

どのタスクがSVCを発行し、それによってSVC発行前のリストが、SVC発行後にどのようにリンクが変更されたかをコマ送りにして見せることで、問題点(2)の解決策とすることができる。

(4) 可視化対象のOSを学習者が操作して、それを可視化したい

(5) 学習者が可視化ツールを操作して、可視化の停止や復帰を自由にしたい

対象OS上での操作によって、OSがどのような動作をするのかを知り、さらに、常にOSの連続した動作を見せるのではなく、学習者が必要なところでOSの動作を停止させて、自分が理解するまで、そのままの表示を保つことでの問題点(3)を解決することができる。

4. 設計方針

まずOSの教育を考えた場合、3.(4)の学習対象者が実際に自分の手で操作して、その結果を見ることが重要である。教育者が対象OSを操作して、それを学習者の目の前で可視化ツールを使って可視化するだけでは、学習効果があがらない。また、OSを解説した書籍とソースコードや擬似コードを見せながら教育することも考えられるが、OSの概念やモデルを示すためにソースコードを見せることは得策ではない。高機能で複雑なOSを解説しても、学習者が理解できない。さらに学習者が実際に利用するOSを可視化することで、学習に対する意欲を高めることになる。よって、OSの可視化ツールの設計方針を、次のように設定した。

(1) 実動するOSの可視化を行う

(2) 対象OSを変更しない

(3) 対象OSは基本機能だけを持ったOSとする

5. 設計

5.1 ユーザへの表示

可視化ツールで示すべきものは、OSのモデルである。可視化する時に、実際のOSからモデルだけを抽出する必要がある。よって次のようにする。

(1) 管理テーブルの抽象化

理解をしやすくするため、管理テーブルを抽象化して見せることにした。アドレスはポインタであるから、矢印で表記することができる。タスク管理テーブルはいろいろな値を保持しているが、値を見せる必要はなく、それの存在や指し示すものを見せるだけでよい。

(2) 割込みシーケンスの表示

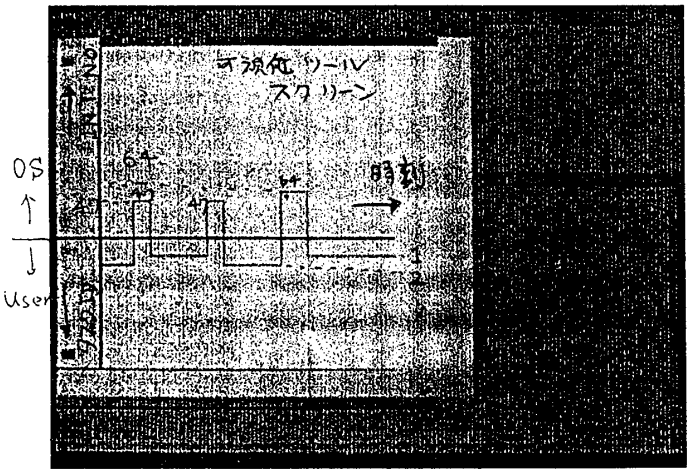
割込み・例外が発生した場合のシーケンスを表示する。これは時間軸を横軸としたグラフとして、表示す

The Design and Implementation of OS Visualizer

ITOH Yoshiyasu, HAYAKAWA Eiichi, NAMIKI Mitarou and TAKAHASHI Nobumasa

Faculty of Technology, Tokyo University of Agriculture and Technology

る。縦軸は“今、どの処理層にいるのか”、“どの割込みか”を表す。これによってOSの非同期な動作を見せる。以下は、タスク1とタスク2がSVC（ベクタ番号=47)を利用して、タスクスイッチを行っている時の画面である。



＜実行画面．割込みの可視化＞

(3) 割込み原因、対象の表示

割込みの発生した原因や、SVCを発行したタスク、その対象タスクを示すために、SVCを発行したタスクと(1)のシーケンスと対象となってタスクを“線”で結び、表示する。

(4) タスク管理の表示

前述のとおり抽象化し、タスク管理構造体（TCB）を“箱”、ポインタを“線”で表す。そして、SVCによってTCBのリンクが付け替えられる前後を表示して、OSによってどのように変更されたかを示す。

5.2 可視化ツールに必要な機構

可視化ツールを実現するためには、次のような機構が必要である。

(1) 割込み・例外・特権命令のトラップ

対象OSで割込み・例外が発生した場合には、それを検知した後、OSが変更した内部情報を検出しなくてはならない。そのためには、割込み・例外・特権命令のトラップをして、管理テーブルの変更箇所を検出しなければならない。

(2) OS内の管理テーブルの取得

変更箇所を知るために、OS内にある各種の管理テーブルを取得しなければならない。

(3) 可視化ツールの処理落ちの回避

可視化中にSVC発行時のデータを表示している間に、他の割込みを取りこぼしてはいけない。つまり、OSの対して発行される割込み・例外の処理落ちが起きないようにする必要がある。

(4) 対象OSの動作停止

学習者が実行した結果を見るために、OSの動作を一時停止させる。

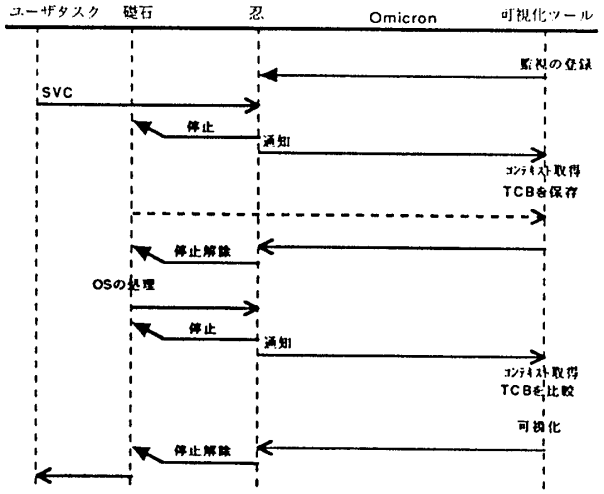
6. 実現

6.1 対象OSを『礎石』とする

前述の設計方針から、対象OSは基本的な機能だけを持ったOSである『礎石』を対象とすることにした。『礎石』には仮想割込み、SVC、タスク管理、排他制御、メッセージ通信といった基本機能だけが実装され、我々の要求を満たしている。

6.2 ハイパOS『忍』の利用

設計方針であげた『実動するOSの可視化』を実現するために、ハイパOS『忍』を利用することにした[1]。『忍』には、割込みの監視、特権状態の取得、CPU状態の取得、監視情報の報告など機能がある。可視化ツールではこれらの機能が必要となる。可視化ツールと『忍』、『礎石』間のデータのやり取りは、次のように行う。SVCが発行されてから、OSがタスクリストの変更を行った場合のシーケンスを示す。



＜図．情報取得シーケンス＞

6.3 実現した可視化ツール

現時点で、OSの理解を困難にしている“非同期性”を理解させるための割込みシーケンスの可視化部分を実現した。これはC言語のリストで約1200行である。

7. おわりに

本可視化ツールによって、非同期動作のために理解しにくいOSを、学生に理解されることが容易になった。今後は、ユーザインタフェース、評価、OSの基本概念のうち可視化できなかった項目への対応についての考察を行うことを考えている。

参考文献

[1] 清水他：OSデバッグ環境の考察とOSデバッグ用ハイパOSの実現，情報処理学会第34回プログラミングシンポジウム報告集，1G−7,1993

[2] Andrew S. Tanenbaum: MINIX OPERARING SYSTEM,ASCII BOOKS,1991