コンピュータと教育　57－6
(2000. 10. 13)

たとえ話を用いたOSの学習支援システムの開発

及川　聡　　並木　美太郎

東京農工大学大学院工学研究科

〒184-8588　東京都小金井市中町 2-24-16
E-mail：satoshi@namikilab.tuat.ac.jp

あらまし　本稿では，オペレーティングシステム（OS）を学ぶ学生に対してOSの概念や動作などの学習を支援するための，たとえ話を用いた学習支援システムについて述べる．本システムでは，タスク管理の概念を学生に理解させることを目標とし，OSをこれから学ぼうとする学生に理解しやすいように，タスクによるプロセッサの仮想化，並行性，同期，排他制御に関する三つのたとえ話を用い，教材プログラムとOS内の動作をアニメーションで可視化して表示する．また，学習の進んだ学生には，教材プログラムと修正問題を提示し，OSの基本構造と簡単な実装の例を学ぶことができる．大学の講義で，本学習支援システムを用いて，前述の三つのたとえ話を学生に説明したところ，OSの理解に有効であることを確認できた．

キーワード　たとえ話による学習支援，オペレーティングシステム，概念，可視化

Development of a system to learn Operating System by examples

Satoshi Oikawa　Mitaro Namiki

Graduate School of Technology, Tokyo University of Agriculture and Technology

2-24-16 Nakacho Koganei, Tokyo, 184-8588 Japan
E-mail: satoshi@namikilab.tuat.ac.jp

Abstract　This paper describes a learning system by examples to understand fundamental concepts and basic structure of OS(Operating System) for undergraduate students. The system animates concurrency, synchronization and mutual exclusion with visual examples to explain a concept and facility of a task management and provides source programs of examples and the system for advanced students. We have evaluated the system in a OS course and confirmed its effects understanding OS's concepts.

key words　learning support by examples, operating system, concept, visualization

1．はじめに

大学でのオペレーティングシステム（以下，OS）の講義は OS の役割，基本的概念と実現方式を理解することを目的としており[3]，それらの解説が黒板や OHP，テキストなどで行われる．講義では概念の説明が大部分を占めるが，学生には仮想化による OS の抽象概念と実際の処理方式の対応が見えにくく，理解が困難になっている．また，非同期な割込みにより頻繁に OS 内部の状態が変化することも学生の理解の妨げとなっている．講義での黒板や OHP などを使った説明では，OS が行うリスト操作や表の参照，タスクの状態遷移などの解説に多くの書きこみがなされ，学生が板書しにくかったり，後でノートを見なおすときに処理の流れがわからず，結局忘れてしまう恐れがある．実際の OS を用いて学習したとしても，OS の実行そのものが目に見えず，処理の流れも非同期の割込みにより変わるので，学生にとって難しいものとなっている．教育用に開発された MINIX[1]はソースコードが 25,000 行以上もあり，修士の学生が実装を学ぶにはよいが，OS の概念を理解するには時間がかかる．このようなことから，OS の学習では初心者の理解を容易にするために，講義の説明に加えて，OS を使いながら概念をモデルで示す必要がある．

そこで本稿では，概念を学ぶためのたとえ話を用いた教材プログラムと OS の処理方式を同時に示すことによって OS の概念の学習を支援する環境を提案する．この環境では OS の役割や概念を説明するたとえ話を OS のアプリケーションプログラム（以下，AP）として実行し，その内容をアニメーションで表示する．処理方式を表示する学習対象 OS は，学生が容易に拡張，修正ができるように，実際の OS ではなく，ユーザプロセスを仮想プロセッサとして実行するプログラムとする．本システムでは，たとえ話の教材プログラムと学習対象 OS の処理方式を同時に表示することにより，学習者が常に概念のモデルを見ながら処理の流れを把握できることを可能にする．

本稿では，OS の中でも基本的な部分であるタスク管理部分の学習支援に焦点を絞り，その学習支援方法と設計，実現について述べる．

2．研究の目的

本研究では学習者が OS の概念を理解できるように，たとえ話や学習対象 OS の内部をアニメーションで示したり，学習対象 OS の変更，修正など学生が能動的に学習できる環境を提供することを目的とする．取り上げる内容はタスク管理部分とし，CPU を仮想化したタスク，タスクの並行性，同期，排他制御の概念を学習者に理解させる．ここでの学習者とは，大学でコンピュータサイエンスを専攻し，OS の講義を受講する学生であり，具体的には 3 年次の学生を指す．本システムでは学習者全員が OS の概念を理解することを第一目標としており，学習者の OS に関する知識は前提としない．

一方，OS の概念をある程度理解し，処理方式を学習しようとする学習者のために段階的な学習を行う環境を提供する．処理方式の学習はソースコードのトレースや同期，排他制御のプログラミング，スケジューリング，システムコールの追加などソースコードの変更を行って理解を促す．

3．　学習支援システムへの要求

ここでは本システムの設計にあたり，OS の学習で理解の妨げとなる要因を分析し，本システムに求められる学習内容と方法について述べる．

3．1　課題分析

OS の理解を妨げる要因として，次のようなことがあげられる．

（1）OS の不可視性

講義では OS の基本概念や処理の流れが示されるが，実際に OS を操作しながら概念や処理の流れを学習することは難しい．普段 OS を使っているとき，OS の講義で取り扱うタスク管理やメモリ管理，割込みなどの様子はコマンドによって確認できるものもあるが，その結果がすでに過去の状態であったり，実体との対応が見えないものが多い．

UNIX では ps コマンドや top コマンドというプロセスの状態を示すコマンドがあるが，これによって表示される情報はプロセスに関する知識をあらかじめ持っていなければ理解できない．また，ps コマンドなどはファイルに書きこまれた情報を表示しているもので，必ずしもリアルタイムな情報とはいえない．OS 上でプログラムを実行させながら ps コマンドなどでプロセスの情報を表示しても，実際とは異なることがある．これらのコマンドではテキストでプロセスの情報が表示されるが，それぞれの情報と実体の対応が見えにくいので，学習に適しているとはいえない．

（2）動的な処理の流れ

OS の内部は常に変化するものであり，それを講義で説明するときに黒板やテキストなどでの説明だけでは限界がある．タスクの状態遷移やスケジューリングなどは時間とともに変化するものであり，それを黒板の図に次々と書き加えて説明していては学習者が混乱してしまう．OS 内で頻繁に行われるタスクの状態遷移，スケジューリングや表の参照，リスト操作などはアニメーションを用いて示すことにより学習者の理解を促すことができる．

（3）OS の規模

OS の処理方式を理解するには OS のソースコードを読むのが有効である．近年，ソースコードを公開している OS が増え，簡単にソースコードを入手し，それを読むことができるようになった．しかし，それらのソースコードは莫大な規模であり，大学の講義という限られた時間内で取り扱える分量ではない．教育用に開発された MINIX でさえも 25,000 行以上ものソースコードがあり，その内容をすべて学習することはこれから OS を学ぼうとする学習者にとっては難しい．

半年で行われる OS の講義を考えた場合，概念の解説などを考慮して，OS のソースコードの解説に割ける時間は 1，2 時限が限度である．OS の概念と処理方式までを学習するのであれば，学習者個人でも取り組める小規模な OS のソースコードが適している．

（4）計算機システムでの構造上の問題

OS は AP とハードウェアの中間に位置するソフトウェアであり，ハードウェアと密接な関係がある．OS は CPU や他のハードウェアからの非同期な割込みによって動作し，ハードウェアや AP の制御を行う．非同期な割込みやハードウェアの制御などは，計算機環境でも下層にあたり，学習者からその処理内容を見ることは困難である．

3．2　概念理解のためのたとえ話の重要性

OS の参考書や講義では，機能の説明をする際にたとえ話を用いている．特に同期を説明する「生産者消費者問題」，排他制御やそのときに発生するデッドロックを説明する「食事をする哲学者問題」などが有名である．OS は資源管理を行うプログラムなので，実社会の管理方法などをたとえることは，OS を理解するときに有効である．人間と計算機は計算のメカニズムが異なるので，人間が計算機のメカニズムを理解することは困難であるが，計算機が行う処理を，人間が普段行う仕事に置き換えれば理解が容易になる．

プログラムの実行単位であるタスクを「何らかの仕事をする人」とし，OS 内で行われる処理をたとえ話として示せば OS の機能は人間にとってわかりやすいものとなる．たとえ話を AP として実現し，その様子をアニメーションによって表示することにより，タスク管理の概念を学習支援することができる．

4．設計方針

本章では，OS の講義で学習者に OS の概念を理解させるための学習支援システムの設計方針について述べる．

4．1　学習のための目標と方針

OS の理解を妨げる要因の解決策として，本システムでは次に示す方針で学習支援を行う．

（1）OS の一般的な概念の学習支援を行う

大学の OS の講義で最低限理解しなければならないのは OS の概念であり，OS の概念は先に述べたとおり，重要な項目である．その概念を学習者の混乱をなくすために，特定の OS に特化しない範囲で学習支援する．

実際の OS は割込みやデバイス操作などハードウェアに依存する部分が多数存在するが，初心者の理解を妨げないためにハードウェアに依存する部分は取り上げない．

（2）たとえ話を用いた教材プログラム，OS 内の動作をアニメーションで表示する

OS 内部の様子が見えないということは，OS の理解を妨げている大きな要因である．OS 内部の様子を可視化し，視覚的に理解を促すことは学習効果があるが，単に内部の状況をそのまま可視化するだけでは，特に初心者に対しては表示している情報の意味を知ることができず，結局理解できないことになってしまう．

本システムでは OS の概念をたとえ話を用いて解説し，次にたとえ話と OS 内の動作をアニメーションで同時に表示することによって，実際にどのように動いているかを理解させる．

（3）学習者がパラメータやソースコードを変更して，OS の動作を確認できるようにする

たとえ話や OS 内の動作をみているだけでは単

に学習者が理解したつもりになってしまうおそれがある．学習者自身が自ら操作してその変化を実感できるように，本システムで能動的な学習をできるようにする．また，概念を理解した学習者がさらにOSの理解を深めるために処理方式と実装の学習としてソースコードの変更などを行う．具体的には，GUI などによってタスクの操作を強制的に行い，その結果を確認したり，教材プログラム，学習対象 OS のソースコードを変更し，その変化を確認できるようにする．

4．2　学習対象 OS の設計方針

学習対象OSは学習の取組みやすさを考慮してUNIX 上に 1000 行程度のユーザタスクとして実装する．1000 行という規模は 1，2 回の講義でプログラムの概要を説明できる程度であり，学習者個人でも取り組めると判断した．既存の OS のユーザタスクを仮想プロセッサとみなし，学習対象 OS をユーザプログラムとして実装することにより，学習者による学習対象 OS のソースコード変更や実行の確認が容易になる．

5．全体設計

5．1　学習支援システムの構成

先にも述べたように学習対象 OS は既存の OS のユーザタスクとして動作するプログラムを用いる．学習対象 OS がユーザタスクであれば，ソースコード変更に失敗した場合でも，計算機システムを破壊することなく，デバッグも容易である．学習対象OS上で動作して可視化される教材プログラムはスレッドとして動作させ，マルチスレッドによってマルチタスクの学習を可能にする．学習対象OSはスレッドを管理するプログラムとみなすことができ，以後，学習対象 OS をスレッドマネージャと呼ぶ．

スレッドマネージャにはタスクの生成，停止，消去，再開などのタスク管理の基本的な機能があり，たとえ話を用いた教材プログラムはスレッドマネージャ上で動作する．これらの動作や資源管理モデルは表示部によって表示する．スレッドマネージャの資源管理モデルは常に表示部で可視化され，ソースコード変更を行ったときも，その変化を可視化する．学習者は表示部の GUI によって教材プログラムやスレッドマネージャの制御を行う．たとえ話を用いた教材プログラムは，学習内容ごとに独立して実行する．

本システムの全体構成を図 1 に示す．

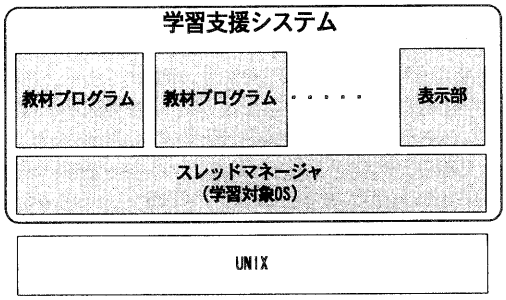

図 1．学習支援システムの全体構成図

5．2　本システムでの学習方法

先にも述べたとおり，本システムの利用対象は大学で OS の講義を受講する学生であり，本システムの利用も大学の講義，演習を想定している．本システムの目標は受講者全員が OS の概念を理解することであるが，概念を理解した学生のために処理方式の学習を行うための環境も本システムは提供する．本章では，本システムの利用を初心者向けから上級者向けまで順に述べる．

（1）概念の学習

まず，教官が講義で OS のタスク管理の概念を黒板や OHP，テキストなどで説明する．その後本システムを使い，黒板やテキストの説明では理解しにくい概念をアニメーションで表示する．概念を説明するために，学習者にわかりやすいたとえ話の教材プログラムを表示し，概念と動作の学習を行う．たとえばタスク間同期の学習では「生産者消費者」のたとえ話を使い，あるタスクは生産者タスク，もう一方は消費者タスクとし，それぞれの仕事をする人間として表される．同期の手段として使われるセマフォは遮断機で示し，同期の概念をアニメーションで表示する．その際，学習者には概念を理解することに専念させる．

概念を理解できたら，次に OS 内で実際に行われている処理方式を教材プログラムと同時に示す．表示はウィンドウを縦に二分割し，上方に教材プログラム，下方に OS 内の処理方式をモデルで表示する（図 2）．

OS 内の処理モデルを示すことにより，実行タスクが切り替えられたり，セマフォによって資源へのアクセスが制限されていることを学習者は見ることができる．概念の学習では，教材プログラムを用いて次に示すモデルを可視化する．

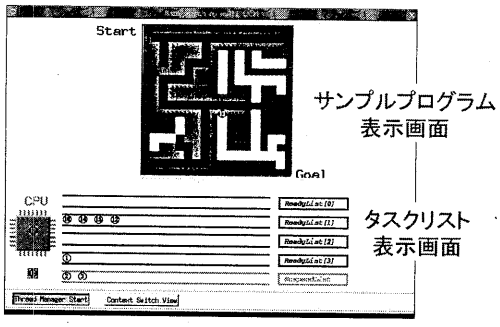

図 2. 実行画面

・教材プログラム
-CPU を仮想化したタスク
-タスクの処理内容
・OS 内の処理モデル
-Ready リスト，Suspend リスト
-各タスクの状態
-スケジューリング

また，OS の諸機能の必要性を理解させるために，問題のある場合の教材プログラムを示す．たとえば，同期をとらなければならない場合に，同期をとらない教材プログラムを表示し，学習者に問題点と同期の必要性を認識させる．

概念の学習では学習者の混乱を防ぐために，タスクのコンテキストといった詳細の情報は抽象化した図として示す．

この段階では，講義で教官がノート PC などで本システムの実行をプロジェクタで投影しながら説明を行う．教材プログラムと処理モデルをアニメーションで可視化することにより，学習者に OS の概念を直感的に理解させる．

（2）処理方式の学習

概念を理解した学習者は，具体的な OS の動作を理解するために処理方式の学習を行う．ここでは学習対象 OS のソースコードを読み，本システムの動作を見比べて OS の動作を確認する．また，概念の学習では触れなかったタスクのコンテキストの内容や，セマフォのデータ構造などをモデルで可視化する．

本システムでは OS の概念を理解させることを重要視しており，処理方式の学習は OS の講義で時間が余っているときに 1 時限程度の利用を想定している．

（3）プログラミング演習

OS の学習には講義での理論や概念の理解だけでなく，実際にプログラミング演習を行うことも重要である．プログラミング演習ではスケジューリングアルゴリズムの変更や，システムコールの追加，先に述べた同期をとっていないプログラムの修正などの課題を学習者に与える．

6．設計

本章では学習支援システムの各機能についての設計を述べる．

6．1　教材プログラムの設計

本システムにはタスク管理の概念を理解させるためにたとえ話を用いた教材プログラムが三つある．ここでは学習項目別に教材プログラムの設計について述べる．

6．1．1　タスクによる仮想化，並行性

タスク管理の基本である仮想化の概念と並行性を理解させるために迷路探索の教材プログラムを使う．この迷路探索は，タスクが迷路探索を行い，分岐点にたどりつくとその分岐点の数だけタスクを新たに生成し，その後の探索は生成したタスクが行うプログラムである．この教材プログラムは忍者の分身の術をたとえている．

忍者の分身の術は自分が立っている場所をすばやく移動することによって，相手にあたかも自分が複数いるかのように見せかけるものである．場所の移動がすばやくないと実体が相手に身抜かれてしまう．また，次の瞬間に立つ位置を間違えると分身の術は失敗する．このたとえ話はラウンドロビンで実行タスクを切り替え，正確にコンテキストスイッチを行う OS のマルチタスキングと同じようなことがいえる（図 3）．

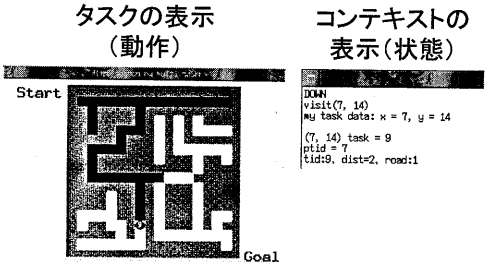

図 3. マルチタスクによる迷路探索

この教材プログラムを実行すると，複数のタスクが同時に迷路を探索しているように見える．ここで，ラウンドロビンのクォンタムタイムを長く設定し，プログラムの確認を行うと，常に一つのタスクしか実行されず，実行タスクは切り替えられているということを学習者は見ることができる．

本システムではタスクが迷路を探索している動作表示と，タスクの位置，進行方向，状態などを示す状態表示の二つの表示を用いてタスクの概念を理解させる．

6．1．2　タスク間同期

タスク間同期の概念を理解させるために，貯水槽の例を用いた生産者消費者問題の教材プログラムを使う．このプログラムは貯水槽を共有資源として，貯水槽に水を入れるタスクを生産者タスク，貯水槽から水を取り出すタスクを消費者タスクとし，同期のモデルを示している．

消費者タスクが必要とする分の水が貯まっていない場合は，遮断機によって消費者タスクは貯水槽にアクセスできなくなり，遮断機がなくなるまで寝て待つ（図 4-a）．貯水槽に消費者タスクに必要な水が貯まると，生産者タスクは遮断機によって水が止められる(図 4-b)．

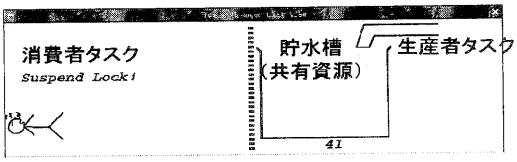

図 4-a．消費者タスクのサスペンドロック

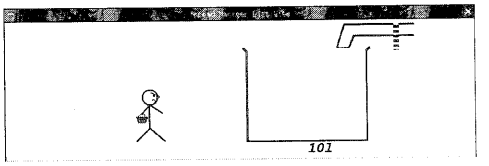

図 4-b．生産者タスクのサスペンドロック

このプログラムでの同期はセマフォを用いており，遮断機はセマフォのたとえとなっている．学習者には同期をとらなかった場合の例を見せ，同期の必要性を理解させる．この例では貯水槽は 10 リットルの水を貯めることができ，消費者タスクは 10 リットルの水を必要としている．同期をとらかなった場合は，消費者が 10 リットル貯まる前に水を持って行く効率の悪さを示すことができる．

教材プログラムのアニメーションを見て同期の概念が理解できた学習者は，P 操作，V 操作をボタンで操作することによって教材プログラムがどのように変化したかを確認できる．また，先の同期をとらなかった場合を解決させるプログラミング課題を与え，同期の実現を理解させる．

6．1．3　排他制御

排他制御は二つのタスクが同じ共有資源に同時にアクセスした場合のたとえ話である預金残高問題のプログラム使う．このプログラムは，一つのタスクが共有資源にアクセスしているときに，他のタスクからのアクセスを避ける排他制御の概念を学習者に理解させる．

ここでは，共有資源を銀行の預金口座とし，二つの異なったATMから口座にアクセスする様子を示す（図 5-a）．一つのタスクが口座にアクセスしているとき，遮断機によってもう一方のタスクが口座にアクセスできないようにロック操作を行う．ロックされたタスクが口座にアクセスしようとしたとき，遮断機によってアクセスできず，遮断機がはずされるまで寝て待つという様子をアニメーションで示す（図 5-b）．また，排他制御を行わなかった場合の例を示し，排他制御の必要性や排他制御の実現を理解させる．

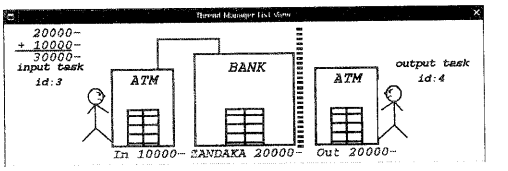

図 5-a．預金残高問題

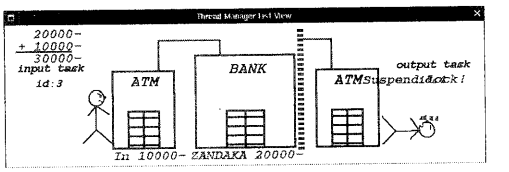

図 5-b．預金残高問題

6．2　タスクリスト表示画面の設計

OS の内部動作の可視化を行うために，スレッドマネージャ内のタスク管理モデルを教材プログラムの下に表示する（図 6）．ここではスレッ

ドマネージャのタスクリスト内のタスクが表示され，スケジューリングはアニメーションで示される．タスクは①，②のように表され，数字はタスク識別子を示している．コンテキストを読み込んで実行中のタスクはCPUに割り当てられていることを示し，色つきで CPU 上に表示する．Ready 状態，Suspend 状態のタスクはコンテキストを読み込んでいないので白地で示す．また，左下には任意のタスクを設定し，そのタスクが実行状態のときだけ表示されるようにする．こうすることによって，ラウンドロビンでクォンタムタイムの時間を短くすると点灯しているように見え，クォンタムタイムを長くすると時間に応じて点滅する．

このタスクリスト画面は，教材プログラムのアニメーションを見て学習者がタスク管理の概念を理解したあとに，OS 内で行われているタスク管理の様子を理解するために使う．

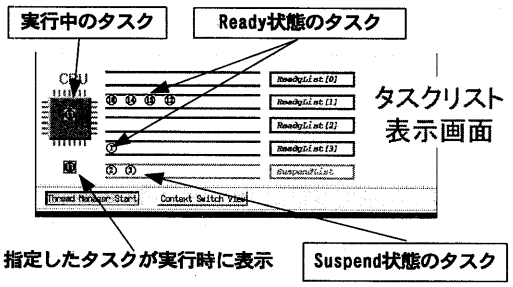

図 6．タスクリスト表示画面

7．評価

これまで述べた学習支援システムを実現し評価を行った．評価は本学学部 3 年次の講義「オペレーティングシステム」で学習支援システムの説明と教材プログラムを見せ，タスク管理の概念を説明したあと，学生にアンケートを依頼した．アンケートに答えた学生は当日の講義参加者 56 名である．評価で説明した項目は次のとおりである．

・分身の術のたとえ話を用いたマルチタスクによる迷路探索
・同期の説明のための生産者消費者問題
・排他制御の説明のための預金残高問題
・タスクリスト表示部でのスケジューリング

また，アンケートには学習支援システムの有効性とたとえ話のわかりやすさを中心に，次のような項目を設けた．

（1）学習支援システムを見て，理解できた部分（図 7-a）
（2）わからなかった部分
（3）たとえ話を使った教材プログラムのわかりやすさ（図 7-b）
（4）各教材プログラムの動作
（5）タスクリスト表示部は役立ったか
（6）本システムの有効性（図 7-c）
（7）学習支援システムに欲しい機能
（8）学習支援システムに取り上げてほしい分野

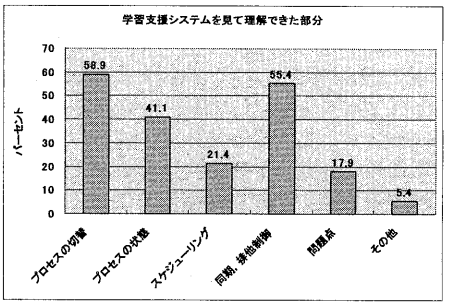

図 7-a．本システムで理解できた部分

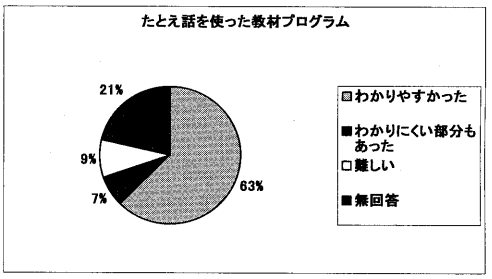

図 7-b．たとえ話を使った教材プログラム

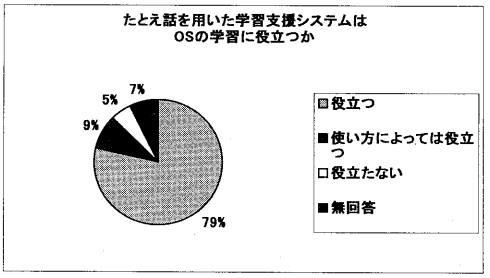

図 7-c．本システムの有効性

8．考察

学習支援システムを見て理解できた部分であるが，プロセスの切替と同期，排他制御が半数以上の学生が理解できたようである．これらは三つの教材プログラムでそれぞれ最低限理解して欲しい部分であり，ほぼ満足できるものである．アンケートの中には「あいまいな部分を確認できた」というものがいくつかあった．

今回の評価は静止画を用いての説明であったが，その説明でも理解できたという回答が多いことから，OS の講義で教官が概念の説明をしながら学習支援システムでその動作を示すことは有効であることがわかる．

学習支援システムを見てもわからなかった部分としてプロセスのリストやスケジューリングが多かったが，この部分はあまり説明しなかった部分である．今回，学習支援システムを用いて学生に理解して欲しかったのが，プロセスの並行性と同期，排他制御の流れであるので，プロセスのスケジューリングやリストについては，実行画面を見せて実行状態のプロセスと実行可能状態，停止状態のプロセスの位置を簡単に示しただけであった．しかし，プロセスリストのアニメーションと具体的な説明ができれば，少なからず理解できる部分であると思われる．他にも，システムコールやプロセスの切替の詳細，プロセスの情報などは今回の説明であまり触れなかった部分である．処理方式部分の学習も説明すれば，違った結果になるだろう．

たとえ話を使った教材プログラムのわかりやすさは半分以上がわかりやすいと答えた．講義では預金残高の話をしていたようだが，生産者消費者問題もわかりやすかったという回答が多くあった．この部分も実際に動いているところを見せていなかったことが影響するが，半分以上がわかりやすいとこたえているので，やや安心できる結果である．

しかし，教材プログラムの内容が少なからずわからなかった学生がいるのが現実である．教材プログラムの内容を理解してもらうためにも，説明や教材プログラムの質の向上をしなければならない．

学習支援システムがOSの学習に役立つかどうかは多くの学生が役立つと答えた．ただし，中には使い方によって役立つという意見もあった．OS の概念を知るには適切で身近なたとえ話で動きを説明し，OS の内部の動きも同時に見せることが有効である．

9．おわりに

本稿ではコンピュータサイエンスを専攻し，OS の講義を受講する学生に OS のタスク管理の概念や動作をたとえ話を用いた教材プログラムや処理モデルの可視化によって学習を支援するシステムの開発について述べた．このシステムによってこれまで講義で説明が難しかったOSの概念がアニメーションによる可視化によって視覚的に説明できるようになり，理解が容易になった．

現在は，学習者が教材プログラムやスレッドマネージャを制御するインタフェースの実現と，割込み，システムコールを学習するための教材プログラムの設計中である．今後はこれらを実現し，講義や演習で用いることによって評価を行う．

参考文献

[1]A.S.Tanenbaum：オペレーティングシステム 第 2 版　設計と理論および MINIX による実装，1998，プレンティスホール

[2]清水謙多郎：オペレーティングシステム　情報処理入門コース 2，岩波書店，1992

[3]（社）情報処理学会：大学の理工系学部情報系学科のためのコンピュータサイエンス教育カリキュラム J97（第 1.1 版），1999，（社）情報処理学会

[4]Thomas D.Wagner, and Eugene K.Ressler : A Practical Approach to Reinforcing Concepts in Introsuctory Operating Systems, Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education, pp.44-47, 1997

[5]Mark A.Holliday:System Calls and Interrupt Vectors in an Operating Systems Course,Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education, pp.53-57, 1997

[6]Daniel A. Canas:GRAPHOS"A Graphic Operating System", Papers of the 18th SIGCSE technical symposium on Computer science education, 1987, pp.201-205