教育・研究報告

システムマネジメント工学科における情報教育に見る情報専門技術者教育の問題点

荒　川　雅　裕*

1．背景

1970年代からコンピュータの高機能化を背景に，情報システムが大規模・複雑化し，将来，ソフトウェアの開発や運用のできる技術者の供給が困難となることが言われてきた．これは「ソフトウェア危機」と呼ばれたものであり，その解決方法として1990年頃から多数のオブジェクト指向に基づく開発技法が提案されてきた［1］．一方で，情報システムでは単なる数値や文字データだけではなく，画像や音声とともに多様なデータを扱うようになり，さらに，ネットワーク技術の発展による複数の機器間の情報交換の仕組みやそれを取り扱う人的な組織構造も技術者の検討事項となってきた．ネットワーク技術の発達は情報交換の対象を世界のあらゆる人とし，リアルタイムでの情報の利用を可能とした．このため，情報システムは大規模・複雑化しているだけではなく，機能の拡張や組織構造や業務の変更によって頻繁なシステムの改良に対する柔軟性が重要視されてきた．情報システムの改良を行う場合では短期間での対応が必要とされるが，該当するシステムの開発者がすでに存在していないケースや存在しても詳細は忘れており，技術伝承が十分行われていないことも多い．このような理由からも，情報システムの設計・開発では必要機能を実現するだけではなく，拡張性や汎用性を高く設計するとともに技術情報を継承できる仕組みの組み込みが必要である．設計者の視点からは，短期間での情報システムの開発・運用を実現するために顧客からの要求機能を分析，具現化し，拡張性や汎用性の高い構造を設計する能力が必要とされる．

現実の SE 系の業務では，情報システムの上流工程（要求分析と設計）が利益に強く影響し，下流工程（実装やテスト）は他企業や他部署で行うことが通常である．設計から実装・テストまでを一貫して同一の作業者が担当するケースはほとんど存在しないが，要求分析や設計においても実装を意識した設計を行わなくてはならない．このような背景のもと，1990年代初めには情報システムを設計・開発できる技術者が必要とされていたが，2000年頃でも国内の大学で情報システムの設計・開発に関するカリキュラムは「ソフトウェア工学」として存在する程度であり，専門性高く，定型技術とした教育はほとんど存在していなかった．

＊　環境都市工学部　教授

（当時，業務とする専門担当者に対して，UML （Unified Modeling Language）などの開発技法の教育カリキュラムが一部の企業で導入されてはいたが.）

著者が所属していた工学部システムマネジメント工学科では，情報システムの開発技術者育成を目的としたカリキュラムを2002年度より導入し，運用してきた．本原稿ではシステムマネジメント工学科における情報システムの設計・開発に関する教育カリキュラムの内容と著者らによる情報専門の基礎教育の運用例を示すとともに，運用から見られた情報教育の問題点を記述する.

2．システムマネジメント工学科における情報システム教育

工学部・システムマネジメント工学科は2002年より管理工学科が改編した学科である．管理工学科は経営工学に順ずる学科であり，数理計画工学，生産管理，情報処理の科目を中心として，経営活動の効率化に関する工学的な技術を教育していた．時代の変遷から，国内の経営工学系の取り扱う対象の問題やカリキュラムは MOT や MBA と言った専門大学院に移行し，それにあわせて経営工学系の学部や学科は応用情報系の学部や学科に改組していった．これは，製造業のグローバル化による日本国内での生産管理の必要性が薄れてきたことや革新的な製品開発が重要視されてきたため，生産管理などの経営工学的なカリキュラムが陳腐化してきたことにある．その一方で，2000年に発表された e-Japan 政策を背景に情報を利用した経営・社会活動が注目され，その中で情報利用の重要さが議論されてきたことで，情報を利用する立場の多くの学部や学科が設立された.

管理工学科では，それまでの学科の方針やカリキュラムを前提にして，社会現象や物理現象をシステムとして捉え，システムを管理する技法を学習する学科へと改組した．その中で，大規模で複雑化する情報システムを設計・開発する能力を習得するためのカリキュラムを学科の柱とした.

情報システムの設計・開発にはオブジェクト指向に基づく設計やプログラミング技法が有効とされていることから，本学科においてはオブジェクト指向言語である Java 言語を利用し，UML（Unified Modeling Language）を利用したシステムの設計法を情報関連のカリキュラムの中心とした．情報システムの設計とともに，維持管理するためにはオブジェクト指向に基づく分析，設計，実装の方法が有効であることもオブジェクト指向技術を導入した要因であった．当時，国内において Java 言語が広く利用され始め，続いて UML の標準図を利用したオブジェクト指向による情報システムの開発技法が提唱され，実務レベルでの利用が広まっていった（UML が OMG（オブジェクト指向の標準化のためのコンソーシアム）に提出，採用されたのは1997年である）．Java 言語や設計・開発の技術が実務レベルで広まったことと教育が可能なほどに UML を利用した設計・開発の方法論が定型化していたことが，オ

ブジェクト指向による情報システムの設計・開発の技術を導入する理由となった．（なお，以前はＣ言語を利用して，アルゴリズムを中心とした実習や数値計算法が情報系カリキュラムの中心であった．）

本学科の１学年の学生数は100～120名程度である．これだけの多人数を社会に送り出し，即，情報システムの設計・開発の業務に携われるほどのレベルの知識をつけるには学内での講義や実習時間だけではなく，自主学習の時間を課すことを必要としていた．このため，効果的に学生を学習させ，さらに学生が自主性を持って学習に取り組むような仕組みの導入が必要とされていた．本原稿ではシステムマネジメント工学科での情報教育に関するカリキュラムの特徴と著者による2003年度およびそれ以後数年間におけるカリキュラムの運用実施例を記述する．

３．オブジェクト指向技法の教育の問題点

情報システムは大規模・複雑化しているだけではなく，機能の拡張や，組織構造や業務の変更により頻繁にシステムの改良が生じている．情報システムの改良を行おうとしても，短期間での改良や該当するシステムの開発者が存在しない条件で扱われる場合が多い．このような理由から，情報システムの設計・開発では必要とされる機能を実現するだけではなく，持続的に維持するための拡張性や汎用性を高く設計する必要がある．一方で，設計者や開発者が異なっても継続的に業務を追行するためには，設計情報を残すことが必要とされる．これらの要件を容易に管理できる手法がオブジェクト指向に基づく設計・開発法であり，具体的な方法論および統一記法が UML である．

オブジェクト指向とは，データの集合に操作を付随させた“オブジェクト”ごとに処理を管理する技法である．一般に，オブジェクト指向によるプログラミングでは，構造化プログラミング（手続きや要素的な処理を単位に分けて複数用意し，大きな処理の流れから，それらの処理を選択する手法）に比べて，クラスもしくはオブジェクト単位で処理の特徴を考える必要があるため，学生には理解が容易ではない．

以前ではＣ言語を利用して情報教育を行っていたが，ＣやＣ＋＋では大規模なプログラムの実装においてメモリ管理の処理を避けることができず，メモリ管理に必要なポインタの取り扱いが生じていた．このため，Ｃ言語による教育においてもポインタの取り扱う時間を長く設定していた．しかしながら，ポインタはＣ言語の習得の難関項目であり，学生の理解の程度が極端に異なる．多くの学生がＣ言語によるプログラミング法の理解ができなくなり，プログラミングに拒否反応を示すようになる項目である．新しいカリキュラムではオブジェクト指向プログラミング言語である Java 言語を導入した．Java 言語ではガーベージコレクションの機能のため，プログラム内でメモリの管理が不要である．Java 言語ならばポインタ

の取り扱いはないため，ＣやＣ＋＋に比べて習得容易な言語と考えた．その他，Java 言語の利点としては，以下が挙げられる．

○ガーベージコレクション

○ OS に依存しないプログラム

○高いセキュリティ（ネットワーク関連のプログラム）

これらの利点から，当時，将来，現実の場で広く利用されるであろう言語として採用することとなった．

一方，Java 言語を利用した教育では，オブジェクト指向の技術（たとえば，継承や多態性など）が使いこなせるかが学生の理解のハードルとなる．Java 言語を導入して講義，実習を数年間行ったところ，プログラム作成に対して学生に以下の問題が見られた．

⑴処理の流れ（順列，繰り返し，分岐）は理解でき，いくつかのサンプルを示すことで習得できている．つまり，型にはまり，例題通りの処理ならば単純に理解でき，自分のものにできる．一方，処理を詳細に手順化できない．フローチャートに相当する処理の流れの図的表現が頭に浮かんでいない．

⑵しかしながら，プログラム化する大まかな処理の流れがわかっていても，１ステップごとの命令に表現できない．つまり，解答を導き出すプロセスの表現に飛躍が存在し，プログラム化することができない．

⑶クラスの存在意味，クラスとオブジェクトの関係が理解できない．例題を示し，利用方法をパターン化して説明すれば，ある程度の応用の利いたプログラムを作成できる．しかしながら，サンプルが存在していないとオブジェクトの作成を忘れる．（つまり，オブジェクトを作成する意味が理解できていない．）

⑷クラス間の関連が具体的な処理（プログラム）に結びつかない．（これは⑵のプロセスが理解できない，表現できないことに関連する．）

４．システムマネジメント工学科の情報関連カリキュラムの特徴

表１はシステムマネジメント工学科における情報関連のカリキュラムである．情報システムの設計・開発の技術習得を目的としており，２年次ではオブジェクト指向のプログラミング技法の基礎的な技術を講義と実習を通して教育している．３年次においては，実践的な教育を目的として情報システムの設計・開発の技法，および，Web 系情報システムに関する講義・実習を行っている．

２年次においてプログラミングおよびオブジェクト指向の考え方の基礎を学習し，３年次では情報システムの設計・実装の技術を学習する．プログラミング教育は２年次の講義と実習から本格化する．２年次において，基本的なプログラミング技術を身につけさせることを

目的として，e-Learning システムを導入した学習管理を行った。この原稿では著者が担当した２年次の教育カリキュラムについて記述する．

２年次のカリキュラムを表２，３に示す（なお，年度や進捗状況によって内容は変化する）．講義である「プログラミング技法」,「応用プログラミング」に対して，実習の「プログラミング実習Ⅰ・Ⅱ」が連動して行われている（なお，実習ではテストを定期的に複数回導入している．）．具体的には同一日の３時限目に講義，４，５時限目に実習を配置しており，授業で学習した内容を実習によって経験的な学習を行い，学生の理解が深めるよう用意している．表２，３の内容からわかるように，オブジェクト指向技術や基礎的なアプリケーション技術（GUI，ネットワーク，デザインパターンなど）を身につけるため，１年間として講義内容を広く設定している．

秋期の授業の内容において,「オブジェクト指向プログラミング」の項目が存在するが，ここではオブジェクト指向の考え方に特化し，そのあとに続く「デザインパターン」の基礎となる説明を行っている（実習では説明は行っていない）．デザインパターンはオブジェクト指向技術によるプログラムの構造の基本パターン群であり，データ構造や処理の基本構成を学習するには教科書的な役割をもつ．工学部の他の情報系学科に比較してもプログラミング等に関連するカリキュラムの時間数と内容は遜色ない程度の内容を学生に供給していると考えている．

講義「プログラミング技法」および「応用プログラミング」では自発学習促進スパイラル学習法を取り入れ，予習→講義→復習のサイクルを繰り返して学習させる方法を導入している．実習「プログラミング実習Ⅰ・Ⅱ」では，基礎的な問題に対する授業時間終了時での課題提出と応用問題に対して１週間後の課題提出を義務付けている．なお，授業時間内での課題は基礎的な問題を設定しており，教科書のサンプルプログラムを改良すれば解ける問題を設定している．応用問題は，授業内の課題を含めて，複雑化した問題を設定している．なお，授業中の課題は３～４問，応用問題は１～２問としている．講義「プログラミング技法」と「応用プログラミング」については，自発学習促進スパイラル教育法による教育を進めている．実習（２時限連続）では初めの約30分でプログラムの概要と課題の説明を行った後，各自が実習に取り掛かる．

次章では，講義「プログラミング技法」と「応用プログラミング」へ対する自発学習促進スパイラル教育法の導入例を説明し，その効果および問題点を示す．

表1　システムマネジメント工学科の情報関連カリキュラム

学年	科目名	実習・講義	内　容
1年 春期	コンピュータ 基礎実習	実習＆講義 （必修） 2時限	情報リテラシー教育（Word, Excel） コンピュータの基礎と構造（2進数，要素の説明，コンピュータシステムの説明など）
2年 春期	プログラミング技法	講義（必修） 1時限	Java を利用してオブジェクト指向言語によるプログラミング法の基礎の学習（処理の流れ，クラスの利用法など，プログラミング実習（春期）と連動）
2年 秋期	応用プログラミング	講義（必修） 1時限	Java を利用してオブジェクト指向言語によるプログラミング法の応用の学習（GUI，イベント処理，データベースの開発，プログラミング実習（秋期）と連動）
2年 通年	プログラミング実習 I・II	実習（必修） 2時限	Java を利用してオブジェクト指向言語によるプログラミング法の学習（プログラミング技法，応用プログラミングに連動）
3年 春期	ソフトウェア工学	講義（必修） 1時限	ソフトウェアおよび情報システムの設計・開発法の説明
3年 春期	情報システム実習	実習（必修） 1時限	情報システムの設計・開発・実装に関する実習（小規模，上流の処理に関する実習が中心）
3年 秋期	システム開発実習	実習（選択） 2時限	大規模・複雑化した情報システム（とくに，Web 系アプリケーション）の設計・開発の実習（主に実装に重点を置く.）
3年 秋期	分散情報システム	講義（選択） 1時限	Web データベースを中心に Web 系アプリケーションの開発技法の説明
3年 秋期	知能情報処理	講義（選択） 1時限	データ構造やアルゴリズムに関する講義

表2　プログラミング技法およびプログラミング実習 I（春期）の各回の内容

回数	授業内容
1	プログラミング概要
2	Java 言語におけるプログラム作成法
3	変数の型
4	処理の流れ
5	配列1（1次元配列）
6	配列2（多次元配列と応用）
7	クラスの作成 1 （クラスの構成（フィールドとメソッド）とコンストラクタ）
8	クラスの作成 2 （内部クラス，匿名内部クラス）
9	メソッドの再定義と動的結合
10	インターフェース
11	パッケージと例外処理
12	マルチスレッドと並列処理1（基礎）
13	マルチスレッドと並列処理2（応用）

表３　応用プログラミングおよびプログラミング実習 II（秋期）の各回の内容

回数	授業内容
1	スレッド
2	GUI
3	グラフィックス
4	イベント処理1
5	イベント処理2
6	アプレット
7	ネットワーク
8	ネットワーク，入出力
9	入出力
10	オブジェクト指向プログラミング
11	デザインパターン1
12	デザインパターン2
13	デザインパターン3

5．自発学習促進スパイラル教育法

自発学習促進スパイラル教育法（以下，スパイラル教育法と略記）は，インターネット技術に基づく情報システムの利用を前提とし，予習→講義→復習のサイクルの中で授業の理解度を高め，学生への自発的学習へ‘強制力’を作用させることを狙った教育方法である．スパイラル教育法を運用するための情報システムでは予習と復習を支援する課題の提示と回答を収集する機能を利用し，また，講義の資料を事前配布する機能を利用した．以下に，スパイラル教育法の特徴を記述する．

5.1　カリキュラムの設計

スパイラル教育法を実施するには，科目の教育内容の構成を考慮した教材の配置と予習・授業・復習との関連の枠組みを予め設計しておく必要がある．今回の教育対象は「プログラミング技法」であるが，プログラミングの教育内容を宣言的な知識と手続き的な知識に分割し，それぞれの知識に関連させて，基礎問題から応用問題に対応できる論理的思考能力の育成を狙うカリキュラムを設計する．ここで論理的思考能力とは，宣言的知識に手続き的知識を適用して解を導出できる能力とする．

カリキュラムを設計するにあたり，教育内容を知識の特徴に分類し，構造化することを考える．

１サイクル内で習得する知識を論理的思考の必要性の有無により２種類の知識（知識 A, 知識 B と呼ぶ）に分類する．一つは論理的な思考を必要としない知識であり，主として宣言的知識に対応する知識をここでは「知識 A」と呼ぶ．知識 A は例えば語句の定義などが対応し，論理的な思考が必要な知識を得るための基礎となる知識である．習得に必要な時間は短

く，自主学習による習得は容易であると考えられる．

他方の知識は論理的な思考過程を経て獲得する知識を指し，手続き的知識やメタ知識に対応する．この知識を本論文では「知識Ｂ」と呼ぶ．知識Ｂの習得には長い時間を必要とするが記憶に長く留まり，知識Ｂは発展的な問題への対応が可能となると思われる．知識Ｂは，例えば，論理式の導出やプログラムのアルゴリズムなどを該当させる．

教育内容に含まれる知識を，知識Ａを下位，知識Ｂを上位とする階層構造に関連付けて一回の授業における学習順序を構成する．この構造化により学生の論理的な理解を高め，一般的な問題解決能力を身に付けることを期待する．構造化の具体的例は次節で示す．

学生に課す学習課題は，予習と復習に対して異なる種類の知識を対応させる．予習では，授業の事前知識として，知識Ａの習得を試みる．講義では知識Ａを知識Ｂと関連付けることや他の知識Ａを利用して知識Ｂを説明する．そして，復習において知識Ｂの確認を行なう課題を回答することで１サイクルの学生の学習を完結させる．これにより，授業前は暗記による知識であったものが論理的に他の情報と関連付けられ，知識Ｂとして理解できることが考えられる．

１サイクルの学習に要する時間（タイムバケット）は，通常の授業の時間間隔である１週間とし，学習内容（学習要素）は１サイクルで完結とすることが望ましい．サイクル内では知識Ａ，知識Ｂを階層的に関連付けて学習する．学期末テスト前の詰め込み的な集中学習を抑制するため，授業の開講期間において一教科の学習に必要な総学習時間を平準化する．すなわち，理解に必要な総学習時間を開講期間内で平均的に分散させる．タイムバケットごとに学習計画を立案し，予習と授業の教材を用意する．

予習・授業・復習のサイクルでの教育実施手順と，学生に対して学習を促進する実施上の工夫について述べる．図１に実施にあたっての時間経過を，図２に予習・授業・復習の各フェーズにおける学生と情報システム間での情報の流れを示す．

⑴予習

教員は当該授業が行われる一週間前から教育支援システムに授業の資料を掲載する．学生はその資料を授業が行われる日時までにダウンロードし，資料を読み，教育支援システム上に出題されている予習課題を解答する．予習課題は知識Ａを対象とし，資料に記述している内容相当とする．ただし，資料の文章と同一の解答を避けるため，字数制限を設定する．これにより，資料の文章を自分の言葉に置き換えることで，学生は課題の意味を記憶し，知識を習得する．そして，取得した知識群は該当する講義のキーワードとして，講義を受けることとなる．また，資料を読み，内容が理解できれば，授業にも興味が湧き，学習意欲が増すことが期待できる．

⑵講義

学生は予習実施の際，取得した資料を持参し，講義を受ける．科目担任者は資料に基づき，論理的証明や問題を示すことなどにより知識 A を知識 B と関連付けて説明する．学生の授業中の反応から学習状況を推測し，復習課題の提出期限を設定する．授業中に授業内容を十分に理解しなければ，復習課題を提出できなくなるため，学生は授業に対する集中度を高め，理解に努めるようになる．

⑶復習

担任者は授業終了後，教育支援システム上に復習課題を掲載する．学生は決められた期限までに復習課題の解答を教育支援システムに提出する．復習課題は授業の内容に関連し，知識 B を対象とする．これにより，復習課題を理解して解答するには授業中に説明を聞かざるをえない．授業終了の一定期間後（１～３週間）に教育支援システム上に本人の解答および予習と復習の解答例を提示する．これにより，学生に自分の解答と解答例を対比させ，自分の解答内容を自主的に理解させる．さらに，提出された解答の共通した間違いや問題のポイントやコメントを解答例に併記することで理解の促進を図る．

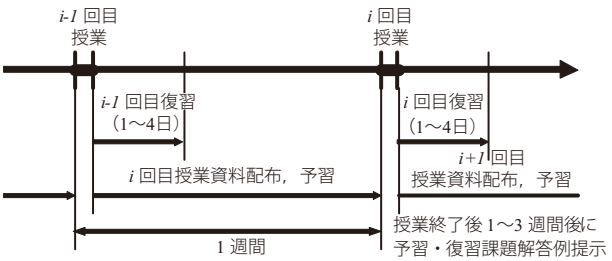


図１　予習・授業・復習サイクルのタイムチャート

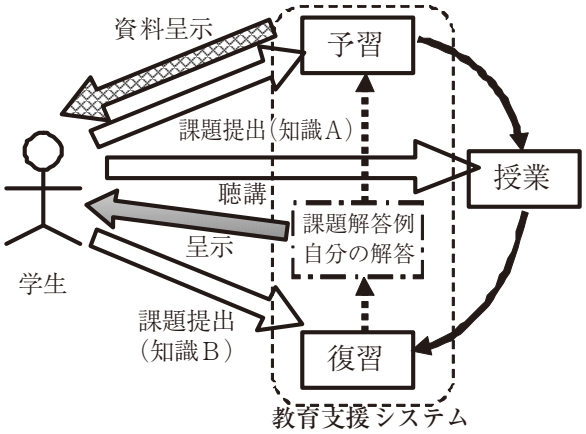


図２　学生と情報システム間での情報の流れ

5.2　教育支援システムと利用形態

スパイラル教育法の実践にあたっては，Web 型自発学習促進クラス授業支援システム CEAS［2］を利用した．CEAS はクラス授業支援と e-Learning システムの両方の機能を有しているが，スパイラル教育法を実施するにあたっては教材提示機能とアンケート収集機能を用いた．この機能で CEAS を用いる利点を以下に示す．

⑴教材提示機能

- 授業の回数ごとに教材を割付け分類できる．
- 教材の登録が容易である．

⑵課題解答に対するアンケート収集機能

- 授業の回数ごとに課題を出題し，解答を自動収集できる．
- 収集した解答結果を授業回数ごとに表形式で表示し，結果をテキストファイル出力できる．
- 学生は一度提出した解答を修正できない．
- アンケート機能の利用がWebブラウザ上に表示されるラジオボタンにより指定でき，アンケート実施の時間制御が容易に行なえる．
- 100名規模の解答結果を一覧できる．

5.3　自発学習促進スパイラル教育法の実践

スパイラル学習法は2003年度秋学期の「応用プログラミング」から2005年度春の「プログラミング技法」までの２年次の講義に適用した．ここでは，スパイラル教育法を2003年度と2004年度の「応用プログラミング」に適用した例を示す．この科目を履修する学生は春学期でJava言語を利用してプログラミングの基礎を学習していることが前提であり，応用的な発展を目的として，GUI，イベント処理，アプレット，デザインパターンなどオブジェクト指向プログラミング技法の学習を目的とする．

表４と図３は各回の講義内容と内容の関連を示す．授業は13回で行なわれ，第13回目の授業終了後，２週間後に期末テストを行なった．スパイラル教育法は第１回目の復習から第13回目授業まで適用した．プログラミングの技術は知識と技術の積み上げにより習得する技術と考えられる．本科目も春学期での科目を基礎として学習する仕組みとなっているが，表４と図３に示すように講義内容の13回の中にも学習の階層構造が存在する．表中の「下位学習」の番号が下位階層に相当する授業であり，上位の内容を理解するには下位の内容の理解が必要となる．本科目ではデザインパターンを利用する小規模のシステム設計論も取り扱っている．この階層構造から，過去で学習した全ての授業の知識の積み上げによって習得される技術である．

各講義における内容の提示および説明の手段を設計するために，李正遠らによる知識の分類と教授・学習方法の関連付け［3］を利用する．図４は李正遠らによる学生の習得する知識と教授法，および学習法の関連構成を示す．図４において，手続き知識を習得するためには予備知識として宣言的知識や他の手続き知識を必要とする．これは暗記に頼って得た宣言的知識を他の宣言的知識や手続き的知識と関連付けることで，学生はその知識を手続き的知識として習得し，発展的で複雑な新しい問題に適応できる能力の習得が期待できる．図５に本科目において一つの講義で論理的な説明を行なうための内容の分類と教授方法の関連付けを示す．この図は図４を参考に作成しており，利用する授業内容の分類，分類を構成する説明内容，説明内容と知識Ａ，Ｂの関連付け，そして図４に示した教材知識との対応を示している．習得する授業内容を「目的」・「特徴」・「事例」に分類し，予習によって「目的」と「特徴」の概要を知識Ａとして学生に習得させる．そして，授業において，図に示す「目的」・

「特徴」・「事例」に関する詳細を他の知識と関連付けて説明を行ない知識Ｂとして学習する．この関連付けにより，学生は宣言的知識である知識Ａを手続き的知識である知識Ｂと組み合わせて理解すると期待される．

表４　2003年度　応用プログラミングの講義内容

回数	授業内容	下位内容
1	①スレッド	
2	② GUI	①
3	③グラフィックス	②
4	④イベント処理1	③
5	④イベント処理2	
6	⑤アプレット	④
7	⑥ネットワーク	⑤
8	⑥ネットワーク　⑦入出力	⑤，④
9	⑦入出力	④
10	⑧オブジェクト指向プログラミング	
11	⑨デザインパターン1	⑦　⑧
12	⑨デザインパターン2	
13	⑨デザインパターン3	

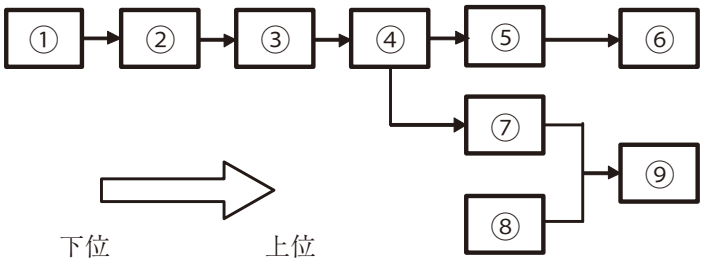


図3　講義「応用プログラミング」の学習階層構造（番号は表4の講義内容の番号を指す）

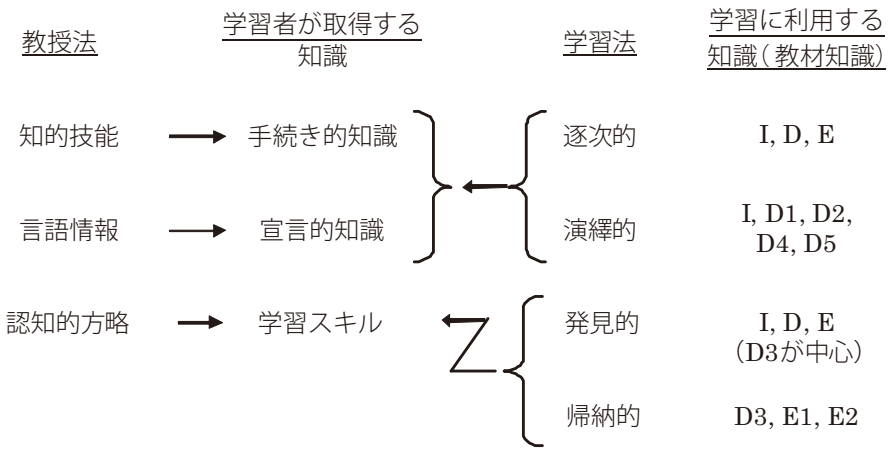


図４　学習者が習得する知識と教授法，学習法の対応

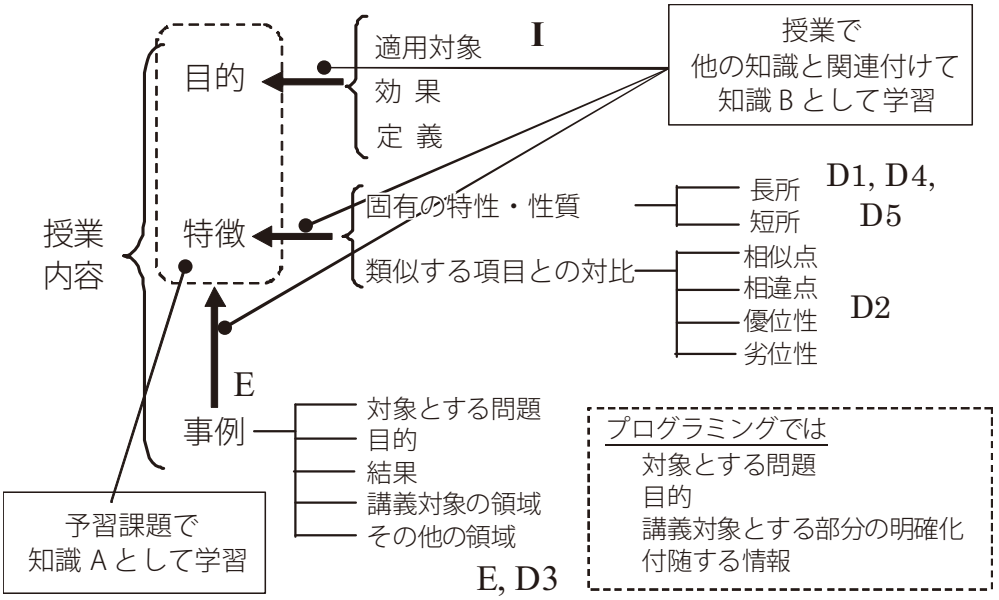


図５　授業内容の分類および説明内容と知識の関連付け

６．自発学習促進スパイラル教育法の導入の効果と問題点

スパイラル教育法の効果を評価するために，2003年度秋学期における出席と課題提出数，および期末（定期）テストの成績を調べた．図６は2003年度の各回における出席と課題提出数の推移である（履修登録者数は162名である）．この結果から予習・復習の課題提出者数に対して30％は授業に出席していない．

学生の学習効果を期末テストの結果から評価する．期末テストは２種類の問題から構成する．一方は予習と復習課題の内容からの出題であり，他方は「論理的な解釈を必要とする知識」を評価するための問題であり，講義では類似する問題は示していない．前者を問題Ａ，後者を問題Ｂとし，問題Ａを24問，問題Ｂを19問設定した．問題Ａについては予習・復習課題の全問題数104問から24問の出題となり，予習復習を自分で解いていれば高得点が取れるものと考えられる．問題Ｂはデザインパターンのプログラムに関する問題であり，クラス図の作成やプログラムの変更などの問題を含む．

図７(a)と(b)はそれぞれ問題Ａと問題Ｂにおける正答率の人数の分布を示す．問題Ａでは正答率70～80％の間で最大の学生数となる正規分布もしくはアーラン分布が得られている．問題Ａの正解率が60％以上の人数は毎回の授業の出席数と同程度である．問題Ａでは予習・復習の問題から出題されていることからも，この結果はスパイラル教育法を実践した学生の正答率が高いことを示唆している．一方，問題Ｂの正解率は70～90％の正答率が高くほぼ均等に人数が分布されている．

図８は各学生の問題Ａ，Ｂの正答率の散布図を示す．図８では両問題の成績間に正の相関が現れており，知識Ａを習得する程度が知識Ｂの取得に影響することが理解できる．この結

果は学生の論理的思考能力に個人差があるものの，問題Ａ（知識ＡとＢ含めて）の内容の理解により，応用問題である問題Ｂにも対応できていることが考えられる．

図９は異なる年度（2001年度と2003年度）の得点分布の比較を示す．2001年度と2003年度の得点を直接比較することは困難であるが，2001年度の問題は2003年度と類似する内容であり，問題Ａと問題Ｂに類似する問題を2003年度と同様に分割して出題している．図９は2001年度の問題の配点を2003年度と同様にして作成した人数の分布である．2003年度では問題Ａでの正答率が高く，問題Ｂにおいても高い正答率に分布しているため，2001年度の分布に比べて優れた結果が得られている．2001年度では予習復習やレポートは課していないため，40～70％程度の正答率の人数が多く，正答率の間での差は2003年度に比べて小さい．2003年度では予習復習の課題から出題していることから問題Ａの点数が高いことは妥当な結果であるが，問題Ｂの成績の分布と2001年度の成績分布を見ても2003年度の分布が優ることから，スパイラル学習法により基礎的な知識だけではなく応用力も習得していることが考察される．

図10(a)，(b)は翌年（2004年度）の「応用プログラミング」の期末テストの得点に対する人数の分布である．この年の問題も2003年度と類似する問題を設定している．2004年度の結果においても，問題Ａに対する正答率（図中は得点）は70～90％に人数が多く，正規分布あるいはアーラン分布に類似する．一方，問題Ｂでは得点間での人数の差は問題Ａの場合に比べて小さい．さらに，問題Ｂでは低い得点（図10では30～40）の人数においてもピークが存在する．この傾向も含めて，2004年度の問題Ａ，Ｂの得点，総得点の人数の分布は2003年度の結果と類似する．また，2004年度における問題ＡとＢの得点に関する散布図には正の相関が見られるものの，2003年度の散布図に比べて分布に広がりを持っていた．これは，前年度に比べて応用問題の解答に対する効果が現れなかった学生が多かったことを示している．しかしながら，いずれの分布もスパイラル教育法の導入以前の2001年度の成績に比べて優れた結果が得られている．2003年度，2004年度の両年度においても、総得点，とくに問題Ｂにおい

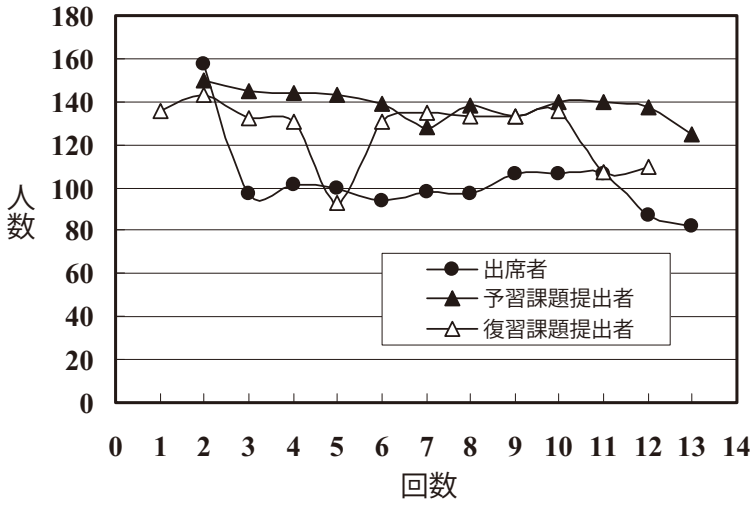
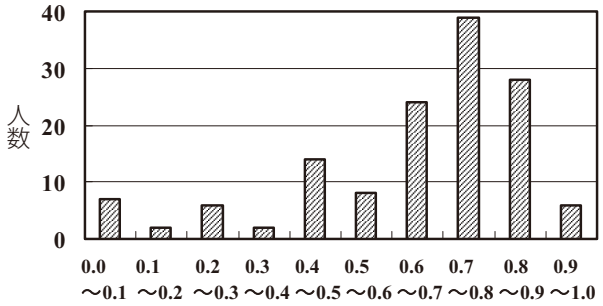
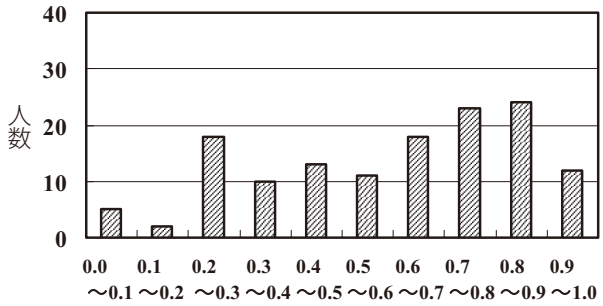


図６　出席者数と課題提出者数の推移

て低い得点を取っていた学生は予習復習の課題を提出していない，もしくは一部しか提出していない学生であった．このことは，予習・復習の導入の効果を示している．



(a)問題Aの正答率



(b)問題Bの正答率

図７　各問題における正答率の人数の分布

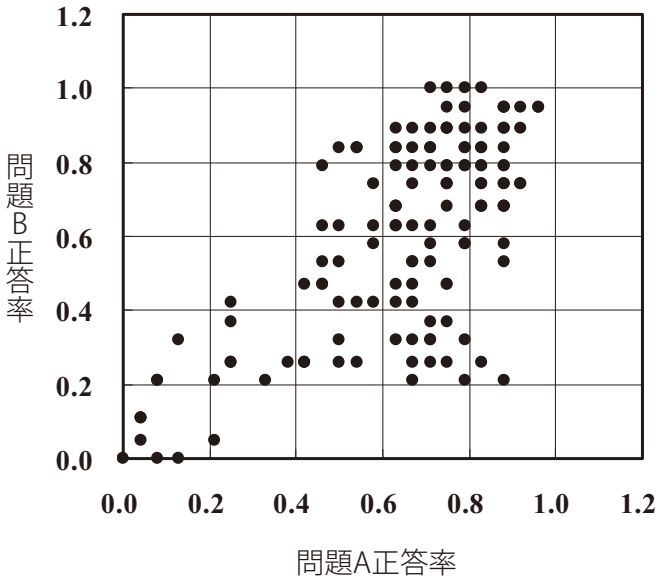


図８　各学習者の問題 A，B の正答率の相関

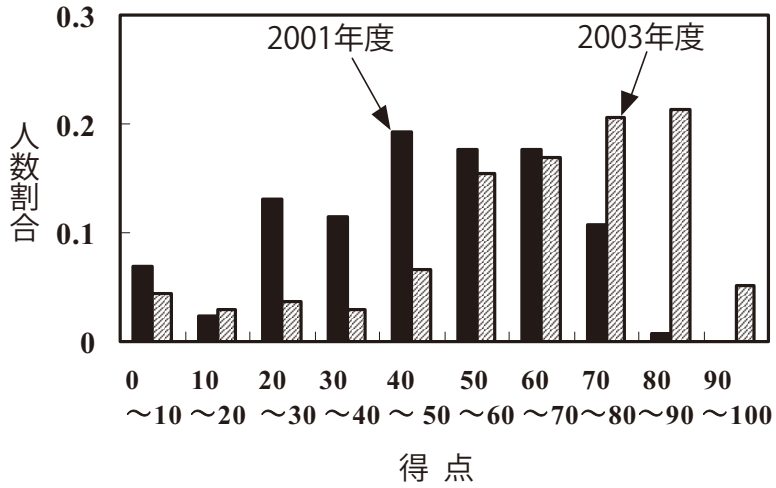
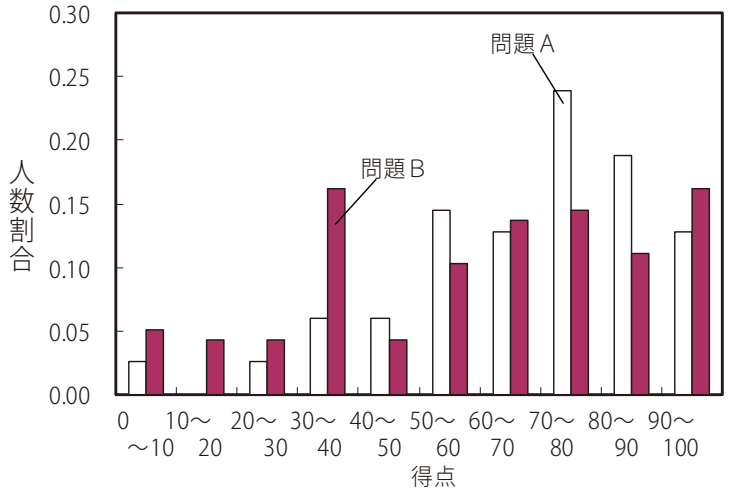


図 9　異なる年度の得点分布の比較



(a) 問題 A と問題 B の得点に対する人数割合分布

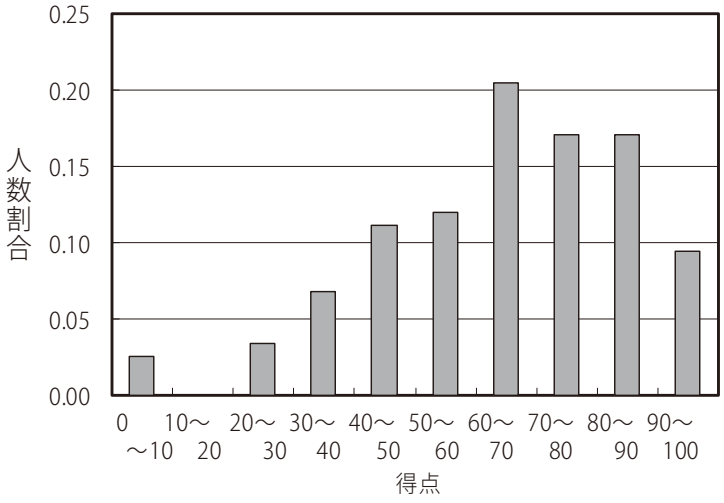


図10　2004年度の定期テストにおける人数割合分布

7．情報専門技術者教育の問題点

数年に渡るシステムマネジメント工学科での情報教育を通して，専門技術者教育に対する以下の特徴が見られた．

⑴予習・復習を必ず提出する学生の成績は優れており，予習・復習のサイクルによる学習効果は明らかである．

⑵しかしながら，部分的に予習・復習を提出しない学生が回数の経過とともに増加する．

⑶応用問題を提出しない学生は，基礎問題（予習）を提出していない学生に比べて多い．そのような学生は応用に対する理解の程度が低いことが考えられるが，むしろ，その中には講義を聞いていない学生が多く含まれているように思われる．

⑷実習や授業が理解できなくなった学生は，出席はするが，初めから授業や実習を行おうとせず他人のレポートや課題の答えを写すことを考えている．

⑸日本語による専門的に利用される言語が理解できない，また想像できないため，専門的な知識（知識Ａ）が理解できていない．（たとえば，継承，構造化，再起処理など．）

⑹講義や実習での説明を集中して聞いていることができる時間は30分～45分程度と短い．

これらの特徴は従来からの傾向として存在するが，上記に当てはまる学生数がこの数年の間に顕著に多くなってきた．とくに，⑵と⑸に該当する学生が増えてきたように感じられる．

現在の大学における環境は本学および他大学も含め，AO 入試や推薦入学の積極導入によって，学力の評価が行われずに多数の学生が入学している．彼らの学習に対するモチベーションは一般に低い．さらに，近年の学生においては，小中高校の頃，宿題などの自主学習が十分行われないようであり，入学時の学力が低いだけではなく，自主学習する癖もついていない．このため，予習・復習の課題の提出が相当な負荷と感じるようになり，課題が多すぎるとのクレームが多数発生してきた．これは，小中高校でのゆとり教育の影響であることや，さらに小中高校で絶対評価を取り入れたことによるインフラ成績（教員も学生も傷つかなくてよいという考えのもと成績が悪くても優秀な成績をつける傾向にあること）に慣れていることが原因であることは容易に想像できる．このため，現在はスパイラル教育法に限らず，予習・復習のサイクルを実現すること自体難しくなってきている．今後，現在の小中高校での教育システムが積極的に変わらないならば，対策として，大学では次の教育方法および体制の導入が必要と考えられる．

⑴（入学から卒業までの）カリキュラムのグランドデザインと（授業の）詳細設計，および設計を取りまとめるためのコーディネータの導入

入学時における学生の基礎学力を考慮しないならば，大学（具体的には学部，学科レベル）でのカリキュラムのグランドデザインと達成度による学生の差別化が急務と思われる．具体的には，専門コースの細分化と必修科目の増加のもと，科目間の関連付けを

明確に行い，科目間の連携を考慮した上で予習・復習の設定を適切な科目を決定する．（あらゆる大学，学部，学科でそうであると思われるが）現状ではカリキュラムを担当教員の責務と判断のもとで教育方法を導入しているが，全体のカリキュラムを構成するコーディネータを導入し，コーディネータの下で複数の科目を通して各カリキュラムの詳細を設計することが必要である．そのためには，必修科目を多数設定し，コアとなるカリキュラムを構築することが必要と考えられる．

⑵差別化カリキュラムの導入と評価の厳格化

本学科の学生が卒研生として配属された場合の学習状況をみると，プログラミングに拒絶反応を示す学生が存在する．このような学生に対しては，初歩的なレベルの課題においても拒絶反応を示すため，予定するカリキュラム構成を適用することは困難である．このため，予定するカリキュラムの内容を削減し，進度を遅くしたカリキュラム構成（差別化カリキュラム）を複数用意し，適用することが必要と思われる．このためには，同一科目に対して複数の詳細カリキュラムを用意することと学生の評価の厳格化が必要である．

⑶反復学習および学習用ソフトの導入

基礎的知識や応用に関する知識が十分理解できていない学生に対しては，応用問題を解かせるよりも基礎的な問題の反復学習を行わせる．類似問題（場合によっては同一問題）を複数用意するか，条件値のみを入れ替えて，「問題の提示→プログラムの作成→実行→評価→次問題の提示→…」を連続して繰り返して行える仕組みが必要である．問題の質よりも数を重要視し，自動的に問題作成と結果収集できる情報システムの開発が必要である．

8．まとめ

本論文では，工学部システムマネジメント工学科における情報教育例を示し，情報教育における実践例を示した．とくに，情報システムの設計・開発者に教育するための学科でのカリキュラム構成と，学習を進める方法として自発学習促進スパイラル教育法の導入事例を説明した．

システムマネジメント工学科は2007年度の学部改組により，分割かつ他学科へ統合された．ここで紹介したカリキュラムは2008年度に実質終了しており，カリキュラムおよび運用法の一部は改組後の新学科の教育として受け継がれている．なお，本原稿は過去の研究報告［4］を元にカリキュラムの特徴とその後の運用から得られた新しい考察を加えたものである．

参考文献

［1］中所武司：ソフトウェア工学（第 2 版），朝倉書店（2004）

［2］冬木正彦，辻昌之，植木泰博，荒川雅裕，北村裕：Web 型自発学習促進クラス授業支援システム CEAS の開発，教育システム情報学会誌，21，（4），343-354（2004）

［3］李正遠，関一也，松居辰則，岡本敏雄：学習履歴情報に基づいた学習過程のダイジェスト化．信学技報，ET　2003-8：43-48（2003）

［4］荒川雅裕，植木泰博，冬木正彦：授業支援型 e-Learning システム CEAS を活用した自発学習促進スパイラル教育法，日本教育工学会論文誌，Vol.28, No.4, 311-321（2004）