付録１

情報システム学とソフトウェア工学のカリキュラムに関する調査報告書

情報システム学とソフトウェア工学のカリキュラムに関する調査研究報告書

情報処理学会

平成 12 年 3 月 31 日

目次

１．　まえがき

情報処理学会においては、平成 10 年度に情報処理教育委員会を設置し、情報処理教育カリキュラムのみならず、教育のための指針、方法、評価等を研究し学問、技術および関連事業の振興に寄与する活動を行ってきた。コンピュータサイエンスのカリキュラムに関しては、1997 年 11 月に「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」を公開し、さらにマイナーチェンジ版（第 1.1）を 1999 年 9 月に発行している。

しかしながら、社会基盤、市場構造、及び事業構造が、急速に情報技術によって改革されるにつれ、情報処理教育に対しても、日本技術者教育認定機構（以下、JABEE と略称）の求める「新しい技術者像」、すなわち、「専門における技術知識や信頼性に精通し、競争力のある独創的製品を創出する能力は勿論のこと、その技術の社会的意義、倫理性、他技術との関連、相乗効果、そして環境、エネルギー、資源、人口などの、人類が抱える重大な課題との関連を深く洞察することができる能力を有する技術者」の育成に貢献する必要

が生まれてきた。

情報処理および情報処理関連技術関連分野において、このような新しい技術者像をもった技術者を必要とする領域は多岐に亙っているが、当面必要とされている領域は、「情報システム学（以下、情報システムを IS と称することがある）」、及び「ソフトウェア工学（以下、ソフトウェアエンジニアリング、または SE と称する）」である。これらの領域においては、近い将来において、専門職業人（プロフェッショナル）、すなわち「社会が必要とする特定の業務に関して、高度な知的訓練と技能に基づいて独占的なサービスを提供するとともに、独自の倫理規定に基づいた自律機能を備えている職業人が必要とされている。このような専門職業人を育成するためには、生涯教育を含めて広く現行の教育を見直す必要があるが、まず、大学等の高等教育機関で実施されている学部レベルの技術者教育プログラムから見直す必要がある、とされるに至った。

以上に述べた必要から、情報処理学会・情報処理教育委員会、および同学会・アクレディテーション委員会は、我が国の大学学部等の情報処理教育における、情報システム学、及びソフトウェアエンジニアリングのカリキュラムのあり方についての第 1 段階の調査、及び研究を、平成 12 年 2 月 16 日から平成 12 年 3 月 31 日までの間、実施した。以下、この調査、研究結果を報告する。

２．米国大学学部における情報処理教育

2.1 コンピュータサイエンス教育

コンピュータサイエンス教育プログラム認定に関する基礎は、ACM(Association for Computing Machinery)、及び IEEE-CS(The Institute of Electrical and Electronics Engineers Inc./Computer Society)が合同で設置した、Joint Curriculum Task Force によって策定された「コンピューティング・カリキュラム 1991（CC1991）」によって確立されたと言えよう。それまでは、ACM、及び IEEE-CS は、別個にモデルカリキュラムを策定していた。ACM は、主としてサイエンス的視点から「カリキュラム 68」を策定し、それを 1978 年、1988 年に改訂している。一方、IEEE-CS が、サイエンス、及びエンジニアリングの視点からカリキュラムモデルを策定し、1977 年に"A Model Curriculum for Computer Science and Engineering"を策定し、それを 1983 年に改訂している。両者のカリキュラムは、初期にはプログラミング言語やコーディングに重点をおいていたが、その後の改訂で、「アルゴリズムとデータ構造、プログラミング言語、コンピュータアーキテクチャ、数値および記号計算、オペレーティングシステム、ソフトウェアエンジニアリング、データベースと情報検索、人工知能、ロボティクス、人とコンピュータのコミュニケーション（全 10 知識体）を、理論(theory)とその適用、抽象化、設計の 3 プロセスに展開し、教育する」、という思想に転換した。CC1991 では、ACM、IEEE-CS におい

て別個に行われてきたカリキュラムに関する視点が統合され、サイエンス、及びエンジニアリングの両視点からカリキュラム編成のための中核知識が定義され、これを情報処理関連分野の各種カリキュラムへ展開できるようにする方法が示された。ACM/IEEE-CSの Computer Science Accreditation Board(CSAB)/Computer Science Accreditation Commission(CSAC) 、 及 び Accreditation Board for Engineering and Technology(ABET)/Engineering Accreditation Commission(EAC)が合同で行う、米国での大学学部等に対する教育プログラム認定活動は、おおむね、この CC1991 を基にして実施されてきた。

CC1991 では、前に記したように、1984 年以前のカリキュラムモデルがプログラミング言語とコーディングに重点を置いていたのを改善し、前記の 10 知識体に基づく、理論及びその適用、抽象化、設計プロセスを重視する方向への転換を行った。すでにこの方針によって教育された卒業生が企業で実務についているが、CC1991 に対するアンケート（124 件の回答が寄せられた）を行った結果、回答のなかにこれら卒業生を観察した結果が報告されていた。報告によれば、かれらは、科学的知識や専門的技術能力(technical skill)に関しては申し分ないが、つぎのような点で問題があるとされている

・人やグループと効果的に交流する能力／チームメンバとして働いた体験と働くために必要な心構え／自分の仕事の効率と生産性を自己管理する能力／の不足
・ ビジネスを実践するために必要な組織構造への理解の不足

CC1991 は、現在、IEEE 及び ACM によって改訂作業が行われており、まもなく「コンピューティング・カリキュラム 2001(CC2001)」として発効するとみられる。CC2001 は、つぎの 10 原則に基づいて策定されている。

・ 研究者(researchers)、及び実践者(practitioners)の育成を目的とする。
・ 数学(mathematics)、サイエンス、及びエンジニアリングを統合したコンピューティング(computing)という領域を対象とする。
・ 知識単位(knowledge units)を定義し、これを基にしたカリキュラム設計を行わせるようにする。
・ 個々のコース（米国でいうコースは、ラーニングユニット、または科目内最小単位の教育目的別集合）の設計を行うためのガイダンスを提供する。
・ すべての学生が習得しなければならない中核（コア）思想を明示する。
・ 各コースに対して推奨される中核思想が示されている。
・ 国際規模に適用可能とする。
・ 企業(industries)の参加を組み込んでいる。
・ 専門職業人に対して求められる実践技術能力(practice skills)を考慮する。
・ 学部(undergraduate)の教育プログラムが求める必要に適合する。

2.2 ソフトウェアエンジニアリング教育

ソフトウェアエンジニアリングは、CC1991 を基にして ABET/EAC が実施する認定活動のなかで、部分的に配慮されてきた。1997 年に入って、米国では、急速に進展する情報化産業に起因するソフトウェアエンジニアリングビジネスへのニーズの拡大、ソフトウェアエンジニアリング・プロフェッショナリズム確立への要求、などに刺激されて、IEEE 及び ACM が合同する、IEEE/Software Engineering Coordinating Committee(SWECC)を設立し、ソフトウェアエンジニアリングに対する、独自の倫理要綱、カリキュラムモデル、認定基準と評価方法の定義を策定する活動を開始した。すでに、倫理要綱、及び認定基準は策定ずみで公表されており、カリキュラムモデルの前提となる、Software Engineering Body of Knowledge(SEBOK と略称： カリキュラムで、教えるべき知識体系をまとめたもの)については、Straw Man 版、及び Stone Man 版が公開されており、さらに 2000 年度には Iron Man 版が完成する予定になっている。現在、カナダで教職についている、パーナス(D.L. Parnas、モジュール、及び情報隠蔽論で知られる）は、ソフトウェアエンジニアリングに対するコンピュータサイエンスの位置を、電気、電子、通信工学における電磁気論のそれと相似である、として、ソフトウェアエンジニアリング・教育プログラム認定に対する、コンピュータサイエンス・教育プログラム認定を関連付けている。

ソフトウェアにおいては、(1)感覚的、情緒的要素を含む利用者からの要求を完全に定義することが不可能である、(2)定義されたとしても、要求は不断に変更される、(3)プログラム変数のもつ状態値、及びその組み合わせが爆発的に大きい、などのことから、それが工業製品としての品質・機能をもつことを検証するための法則、原理、原則が、完全に形成されていない。したがって、管理、規律など、集団的人知に頼って品質・機能を確認するヒューマンファクタがとくに重視される。SWECC では、ソフトウェアエンジニアリング・カリキュラムモデルを編成する場合には、コンピュータエンジニアリング全般に比べて、より一層、学部等の教育における個人ソフトウェアプロセス(individual software process: PSP)教育、大学教官等の指導による小チームプロジェクトによるチームソフトウェアプロセス(team software process: TSP)教育、企業へのインターンシップなどを組み込む必要があるとしている。ここ数年、日本では、発想や PSP に秀でたソフトウェアサイエンティスト、またはソフトウェアエンジニアの育成に、官学を挙げての努力がなされているが、米国では、日本と違って、とくに現実の IT(information technology)からのアカデミアの遊離、ギャップの広がりが問題視されており、産学協同による実業界と学界のコラボレーションに努力が払われている。

(a) ABET および CSAB によるアクレディテーション

CSAB/CSAC は、先に述べた CC1991、及び ACM または IEEE-CS で作成された推奨カリキュ

ラムを認定基準に組み込んで、主として大学等の「Computer Science Department」の認定を行ってきた。一方、ABET/EAC は、主として「Computer Engineering Department」からの認定申請を受け入れ、認定を行っている。1991 年からは、CSAB/CSAC、ABET/EAC 双方から認定を受けられるような制度も取り入れられ、双方の認定を受ける大学等も出現した。1997 年 3 月現在で、

・ ABET/EAC の認定を受けた大学学部は、８０件
・ CSAB/CSAC の認定を受けた大学学部は、１２８件
・ 双方の認定を受けた大学学部は、９件

となっている。

(b)　ソフトウェアエンジニア資格

アメリカでは、master of business administration(MBA)という資格が社会的な貢献を認められているが、これに相当する master of software engineering(MSE)という degree を授ける組織が、1984 年 12 月に、カーネギーメロン大学 Software Engineering Institute(SEI)で始められた。学部で離散数学、プログラミング、データ構造、アルゴリズムを習得し、適当なコミュニケーションスキルをもった学生が、30-36 セメスタを学習することによって、MSE の degree を取得することができる。アメリカの 1 セメスタとは、1 週に 1 時間の授業と 2-3 時間の課外学習を行い、これを 15 週続けることを意味する。セメスタは、6 グループ（ソフトウェア仕様、ソフトウェア検証と確認、ソフトウェア生成と保守、ソフトウェア設計、ソフトウェアシステム、ソフトウェアプロジェクト管理）から構成されている。セメスタの 30%は、経験の獲得を目指す実践教育で占められている。倫理および社会問題に対応する教育も含まれている。

(c)　大学と企業とのコラボレーション

大学学部教育における大学と企業とのコラボレーションは、数多く発表されている。双方のベネフィット(benefit)が、ある一定値以上なければ成功しないものであるが、報告されているところでは、ベネフィットは双方にあるとされている。企業側のベネフィットは、社内教育を大学教員に負担してもらうことによる経費削減、大学への研究委託が進展しやすくなる、企業イメージの向上（免税問題も絡むらしい）、などが挙げられている。これに対して、大学側のベネフィットは、学生に実世界を展望する機会を与えることができる、実プロジェクトにメンバとして参加することによってエコノミー経験を与えることができる、社会人との対話を行う訓練ができる、実務と研究とのギャップを狭めるために役立つ、などがあるとされている。主な例をつぎに示す。

・カリフォルニア州立大学

Software Engineering Forum for Training(SEFT)という州政府が作った協会組織とコラボレートして、学生・企業人の教育を行っている。

・南カリフォルニア大学

University of Southern California の Center for Software Engineering は、29 の企業および政府が参加する共同組織を作って、学生・企業人の教育を行っている。

・アメリカ大学（American University）

American University, Dept. of Computer Science and Information Systems は、Center for System Management という企業と共同して、システムおよびプロジェクト管理に関する教育と検定を行っている。

・アンブリ・リドル航空大学（Embry-Riddle Aeronautical University）

毎年、大学の Industrial Advisory Board に入っている会員企業（モトローラ、マクダネル・ダグラス、ロッキード・マーチンなど）に対して、少なくとも 10 人の学生を派遣して、教育。訓練を受けている。

・フロリダ・アトランティック大学

Florida Atlantic University は、モトローラとの間で、教育プログラムおよび研究活動でのコラボレーションを行っていたが、新たに IBM などを含む 9 以上の企業との間で Industry Advisory Committee を作り、学部教育改善のための長期的な施策を立て、1990 年から実行に移している。

・メリランド大学

Computer Data Systems, Inc.は、1993 年以来、Capability Maturity Model をベースにした､ソフトウェアプロセス改善のための教育プログラムを一般に対して実施していたが、1996 年に University of Maryland と正式に契約を結び、共同でカリキュラムを開発し、学生も含めて教育・訓練を行うようになった。

・マンマス大学

US Army, Communications-Electronics Command, Software Engineering Center の協力によって、学部、修士課程、インターンに至るすべてのカリキュラムを開発し、ソフトウェアエンジニアを US Army に供給している。

2.3 情報システム学教育

情報システム学（以下、IS 学と称する）に関するモデルカリキュラム策定活動は、主として ACM によって 1970 年代の初めから開始された。表 1 に、我が国を含む、主な活動と成果の履歴を示している。現在、参照できる最新の成果物は、IS'97 と称する、学部用の情報システム学モデルカリキュラムであり、これは ACM、AIS（Association for Information Systems）、及び AITP（Association of Information Technology Professionals：旧ＤＰＭＡ）によって、1997 年に公示された。IS'97 は、学問領域として、

・情報技術（ＩＴ）とサービス（ＩＳ機能）に関する獲得、開発、マネジメント

・組織プロセス（システム開発）における技術基盤とシステムの開発・展開

の２領域をカバーしている。

一般的なソフトウェアパッケージの使用技能を前提としたカリキュラムであり、次の３レベルで構成されている。

レベル１；全学生対象（ＥＵＣの生産性向上、ＩＳの基礎、ＩＳの理論と実践）

レベル２；ＩＳ学の主専攻者・副専攻者対象（ハード的ソフト的なＩＴ、プログラミング、データ・ファイル・オブジェクトの構造、ネットワークと通信、ＩＳ開発の分析と論理設計）

レベル３；ＩＳ学の主専攻者対象（専門的なＩＳ開発プロジェクト、データベースの物理設計と実装、プログラミング環境の物理設計と実装、ＩＳの展開とプロジェクト管理の実践）

表１　IS 学カリキュラムの策定に関する主要年表

1972．5	ACM：大学院専門向けＩＳプログラム（Ashenhurst 1972）
1973．12	ACM：学部向けＩＳプログラム（Couger 1973）
1974	IFIP：情報分析者、システム設計者用カリキュラム
1981．3	ACM:学部・大学院ＩＳプログラム(Nunamaker,Couger and Davis 1982)
1981	DPMA：学部向けＩＳ教育カリキュラム（DPMA 1981）
1984．10	DPMA：ITおよびCISカリキュラム
1985．10	DPMA：ＣＩＳの大学院モデルカリキュラム
1985．10	DPMA：ＣＩＳの学部向けモデルカリキュラム
1987	IFIP，BCS：ＩＳカリキュラム（Buckinghan 1987）
1990．5	ACM/IEEE：学部向けＣＳカリキュラム
1991．6	DPMA：IS‘90（Longenecker and Feinstein 1991c）
1991．7	ACM：CSカリキュラム（Turner and Tucker　1991）
1992	科研費（代表浦昭二）：情報システムカリキュラム
1993	情報処理学会：情報システム学カリキュラム
1994．1	DPMA：IS‘94カリキュラム/ISの2年間プログラム（Longenecker, Feinstein et al. 1994）
1994	ACM，AIS，DPMA：IS’95
1996	私立大学情報教育協会：情報システム学モデルカリキュラム
1997	ACM，AIS，AITP：IS’97
1998	浦ほか：情報システム学カリキュラム（改定版）

３．　ソフトウェアエンジニアリング教育に対する要求

3.1　学部における教育の品質、及び効率の向上

昨今わが国の大学学部の一部に見られる、大学教員の教育理念についての自信喪失、学生における追求すべき理想の喪失、企業の学部卒業生に対する期待レベルのあいまい化、大学入学以前の初等教育方針変化に対する未熟対応、親として当然与えるべき家庭での躾の荒廃、わが国社会の個人責任回避風土、などに起因する、大学学部等教育の品質、及び効率低下が存在する。この問題に挑戦するための施策のひとつとして、つぎの定義を行い、その評価を実施する必要を考慮せざるを得ない。

(a)　教育プログラム品質の最低レベル（水準と評価尺度）、及び評価方法の定義

(b)　学部学生が達成すべき最低レベルの段階的定量化、及び評価方法の定義

以上の定義に沿った教育改善の結果として、JABEE が提唱する、教育プログラムの国際的共通評価水準（ワシントン協定が定める）を満たすことを目指す必要がある。そのために、わが国のソフトウェアエンジニアリングプログラム認定基準は、IEEE/ACM の SWECC で策定した、「Accreditation Criteria for Software Engineering (revised 9/25/98)」へ写像(mapping)可能でなければならない。わが国の基準を定義域にし、IEEE/ACM の SWECC 認定基準を値域に割り付けたとき、この定義域／値域間写像は部分・全射、または全域・全射でなければならない。

3.2　ソフトウェアの本質に基づいた教育

(a) 人工系工学を指向した教育

ソフトウェアエンジニアリング（ソフトウェア技術、またはソフトウェア技術業）とは、ABET/JABEE でいう「エンジニアリング」の定義に従う 1 分野であり、ソフトウェアの開発、運用、保守に関して、(a)学理に基づいて体系的に、(b)エコノミー及びヒューマンファクタを考慮しながら、(c)定量的に取り組む専門職業(profession)である。

自然系工学、すなわち電気工学や機械工学のように自然法則の利用を基軸とする工学に比べ、ソフトウェアエンジニアリングは、人が思考、実践、経験を通して得た、認知空間のなかの普遍的法則、原理、原則から構成(construct)される。たとえば、ダイクストラ(E.W. Dijkstra)の相互排除(mutual exclusion)、ホア(C.A.R. Hoare)のホア論理(Hoare logic)、リスコフ(B. Liskov)の抽象データ型のようなものである。これが、ソフトウェアエンジニアリングは人工系工学に属するといわれる所以であるが、これら過去に確立された人工的法則、原理、原則のほとんどは、数理科学などの人工的学理に基づいて実践された、ソフトウェアを自ら構築するときの経験から生まれてきた、といってよい。

ソフトウェアは、それを実装し、実行するコンピュータハードウェアに比して、実行時に生成される状態数がきわめて大きく、ハードウェアを含めての全正当性(total correctness)の検証(verification)はほとんど不可能とされ、部分正当性(partial

correctness)の検証すら現実には困難とされている。すなわち、現在得られている普遍的法則､原理､原則を極力用いて構成的(constructive)に作られたソフトウェアといえども、その正当性を実用的な水準で検証することは不可能である。このことから、ソフトウェアエンジニアリングは、まだ、工学とよばれるにふさわしい水準に到達していないのではないか、とする批判も受け入れざるを得ない状態である。しかし、すぐれたソフトウェアエンジニアは、直感、あるいは発想ともよばれるトレース不能な能力によって、正当性の高いソフトウェアを構築する。このようなエンジニアを仮に「スーパエキスパート、またはスーパプログラマ」とよぶとすれば、スーパエキスパートの育成に結びつくような教育とカリキュラムのあり方を検討しなければならない。

情報を基軸とする社会変革は、高品質・高機能ソフトウェアの広範囲における供給を促し、これによって社会基盤を構築し、もって経済の活性化を期待せんとする声が大きい。これに応えて､ソフトウェアを工学的生成物として社会に提供するためには､普遍的法則、原理、原則、知識、知見の教育と、規律(discipline)の訓練を組織化し、個人、およびグループの技術的水準を高度化、かつ平準化し、ソフトウェア開発、運用、保守を体系的実践し、管理することによって、検証不可能な部分を人知による確認(validation)をもって補うことにより、工業的価値を付与しなければならない。

ソフトウェアエンジニアリング教育プログラムにアクレディテーションを適用しようとする目的は、ソフトウェアエンジニアリング教育と訓練を、大学学部等の教育プログラムの段階から適正に行うことによって、上記のニーズに対応できるような､「最低限必要な」素養をもった学部等卒業生を育成しようとすることにある。

(b)　分析性と構成性を意識した教育

従来の自然系科学は、自然法則を発見し、法則体系を知見する過程において、デカルト流の分析的プロセスをとる場合が多かったため、サイエンス教育は、一般に、分析的(analytical)に形成されている。これに対して､工学は､科学で得られた知見を活用して、人工物やシステムを研究・開発・製造・運用・維持することを目的とするため、エンジニアリング教育は、おのずから構成的(constructive)であることが要求される。

自然法則を基にする自然系科学と自然系工学の間では、教育過程において、分析的思考から構成的思考への移行に、さしたるギャップは見られなかった。たとえば、分析的に発見されたマックスウェルの電磁方程式は、変圧器や発電機を設計するための構成的基本法則としてそのまま役立ち､実践的な電気工学の教育においても自然に受け入れられてきた。ソフトウェアエンジニアリングでは、それが形成される前提としてソフトウェアサイエンスが存在し、そこで発見され、体系化される普遍的法則、原理、原則が、実用規模のソフトウェアを構成するために役立つ､というシナリオが成り立っていない｡このシナリオは、プロダクト、及びプロセスに関する、体系化されたアーキテクチャ、デザインパターン、コンポーネントなどを基礎にした、PSP、及び TSP 演習のなかで体験させる必要がある。

一方、スーパエキスパートが、自己の才能に目覚めるためのモラトリアムはできるだけ若い年齢層に移行させることが望ましい。このためには、1,2 年生で分析的リベラルアーツ科目を、構成的リベラル科目と並行して習得すること、および産学協同の TSP 演習にできるだけ早くから参加させること、などが役立つと考える。

(c) エコノミーファクタを重視する教育

ソフトウェアにおけるエコノミー(economy)とは、情報倫理（たとえば情報処理学会・倫理要綱を参照）に従い、顧客の利益および従業員の利益や、環境問題、住民問題を配慮しつつ、法律・制度の遵守して、できるだけ少ない投資からできるだけ大きな利益(profit)を得るために必要な知識の習得、訓練を意味する。社会情勢、経済情勢、リスク、不確定性を考慮し、ソフトウェアの出荷品質に対する投資の損益分岐点を明確にし、生産、品質、原価を管理するための知識習得と訓練を行う必要がある。現在、CSAB/CSAC や ABET/EAC によって行われている教育プログラム認定には、倫理問題の考慮はされていても、エコノミー視点からの配慮が不足しているとされる。

(d) ヒューマンファクタに目覚める教育

ソフトウェアは、人の知的資産のひとつであり、その内容のすべてが、そのソフトウェアに関わるすべての関係者の認知空間で形成された「モデル」によって充足(satisfy)されるべき「論理システム」である。ソフトウェアは、その全ライフサイクルにおいて、「モデル」を充足するように、開発され、運用され、保守されなければならない。したがって、ソフトウェアエンジニアは、絶えず、日本語、または英語を用いた対話または記述により、顧客、チームメンバ、使用者または操作員、保守員などと交流し、自己を自律的、かつ相対的に管理し、組織を活用して、人との交流のなかで新しい創造を行う訓練を心がけねばならない。

ソフトウェアエンジニアリングにおいて、欧米を越えるためには、どのような教育を学部で提供すべきかについて考え、かつ具体的なソフトウェアエンジニアリングカリキュラムモデルを策定する必要がある。わが国で実施されて久しい卒業研究制度は、わが国独特なものであり、日本の工業を世界のトップレベルに押し上げることに実質的に寄与した、貴重なカリキュラム要素のひとつである。また、1970 年代後半から 1980 年代にかけて成功した、わが国ソフトウェアファクトリは、ソフトウェア品質、及び生産性向上のための集団による生産方法論の実践として、世界的に認知された実績例である。これらの実績をベースにして、わが国の産業風土、国民性を正しく認識し、真に独創性のある新しい技術者を生み出す可能性を模索、研究し、この結果に基づく適切な教育方法のあり方を策定したい。

3.3　プロフェッショナル育成

IEEE/SWECC が求める技術者像は、JABEE の用語に従えば「専門職業人」、平易な用語でいえば、「プロフェッショナル」であると考えられる。我が国でも、司法試験、外交官試験、建築士試験、電気主任技術者試験、医師国家試験、技術士試験など、プロフェッショナルへの道が設けられている。　情報分野では、JIPDEC の情報処理試験があるが、これは、国の高度情報化人材育成のために設けられた国家試験であって、プロフェッショナル資格とは目的を異にしている。現在、大学学部生の多くは、在学中に第二種情報処理試験に合格しているが、入社試験は主として「人物本位」で行われ、必ずしも情報処理試験は、合格の条件にはなっていない。米国やカナダで認定された大学学部出身者は、州政府が公的に認める専門職業人（プロフェッショナル）の道に進むことができる場合がある（テキサス州、カナダのブリティッシュコロンビア州の例がある）。

米国では、日本ではなお企業内で処理すべきであると考えられている業務、たとえば、厚生、勤労、人事、財務ですら、社外にアウトソーシングし、それを受ける専門ベンチャー企業が数多く生まれ、アウトソーシングした企業もそれによって利益を増進できたことが、景気改善に大きく寄与したとされる。日本でも、ソフトウェアエンジニアリングは、大部分の大企業からは分離され、関連会社にアウトソーシングされているが、今後はもっと小さなベンチャーソフトウェアエンジニアリング企業の形成が期待されている。これを促進するためには、「ソフトウェアエンジニアリングを専門職業とするプロフェッショナルの存在を社会的に認め」、「プロフェッショナル認定制度を確立し」、「企業がこれらプロフェッショナルにソフトウェアエンジニアリング業務を積極的にアウトソーシングする」ようにしなければならない。

幸い、我が国でも、JABEE が認定する教育プログラムを卒業した学生には、技術士の第 1 次試験が免除されるようになったが、これに応えることのできる、ソフトウェアエンジニアリングでの専門職業人を育成するようなカリキュラムを検討する必要がある。

４．　情報システム学教育に対する要求

実社会における事業に寄与する情報システムの開発（企画・分析・設計・実装）・運用・維持・管理を実施するために必要とされる要件を備えた専門職としての技術者を育成することにある。そのためには、つぎの具体的要求を満たす必要がある。

（a）情報システム学を専攻する大学学部卒業生の人材像の明確化

（b）情報システムの専門家が組織や社会において期待される能力と知識の明確化と知識体系の確立

（c）情報システム学のコアとなる専門知識体系の確立

（d）情報システム学を専攻する大学学部卒業生を中心に（社会人の再教育を含めた大学

院修士課程修了者も考慮した)、備えるべき能力と知識、およびそれらの確認方法の明確化
（e）学部課程を中心にした情報システム学のモデルカリキュラムの原型策定

IS'97 に見られる情報システム学の範囲

IS'97 では、情報システム学がカバーする範囲に関して、つぎのような見解を与えている。

情報システム学は、学術的領域として２つの広い領域をカバーしている。それは、「情報技術の資源とサービスの調達と、開発、マネジメントに関する情報システム機能」と「組織プロセスで利用するための基盤とシステムの開発および展開」である。

情報システム機能には、情報技術基盤（コンピュータと通信)、内外のデータおよび組織規模のシステム開発および実装、さらにマネジメントまでを含めた広い範囲にまたがっている。また新しい情報技術を受け入れ、組織の戦略や、計画、実践に取り込むための支援も含んでいる。それらは、組織および個人の情報技術システムを支援するものである。

組織と組織間プロセスのためのシステム開発活動は、データ獲得、会話、調整、分析、意思決定支援における情報技術の創造的な利用を含んでいる。この活動のための方法、技法、技術、方法論がある。組織でシステムを創り出すことには、革新の問題、品質、人間－機械システム、マンマシンインタフェース、社会技術デザイン、変化のマネジメントなどが含まれている。

情報技術は全ての組織機能に行き渡っている。それは、会計や、財務、マーケティング、生産などで利用される。利用の普及はシステムマネジメントとシステム開発の専門知識を伴う情報システム専門家のニーズを増大させる。そのような知識を持った専門家が情報基盤や情報資源の調整に関する革新や、改革、マネジメントを支援する。IS スタッフによるシステム開発は、組織規模でのシステム統合を含むだけでなく、個人的および部門的アプリケーション開発を支援する。

情報システム学とコンピュータサイエンスとは近い関係にあるが、行われる仕事や、解かれる問題の種類、デザインおよびマネジメントされるシステムの種類、技術が採用される方法などの文脈においてコンピュータ科学との相違が導かれる。情報システム学の目標は、組織的使命と目的、情報技術の応用に集約される。このように情報システム学とコンピュータサイエンスは別の研究領域であるが、それらは共通の技術的知識を必要としている。

５．ソフトウェアエンジニアリング教育に対するカリキュラムのあり方

カリキュラムは、国際標準に従って、知識体(body of knowledge)、ラーニングユニット(learning unit)、知識レベル、コース、カリキュラム表示区画(curriculum presentation area)、カリキュラムモデルによって構成され、表示されなければならない。これらカリキュラム構成、表示要素の関係は、図１に示すとおりである。

ラーニングユニットは、我が国でいう科目を構成する単位であり、科目より少し粒度が

小さい。各ラーニングユニットには、知識体を構成する知識要素が、それぞれ知識レベルを付して写像される。コースは、ラーニングユニットを、技術課題ごとにまとめた集合である。カリキュラム表示区画を構成する要素は、（コース名、ラーニングユニット名）の組である。カリキュラムモデルは、カリキュラム表示区画を、時間軸に沿って、逐次並び、または並列に連結する方法で組み立てられる。

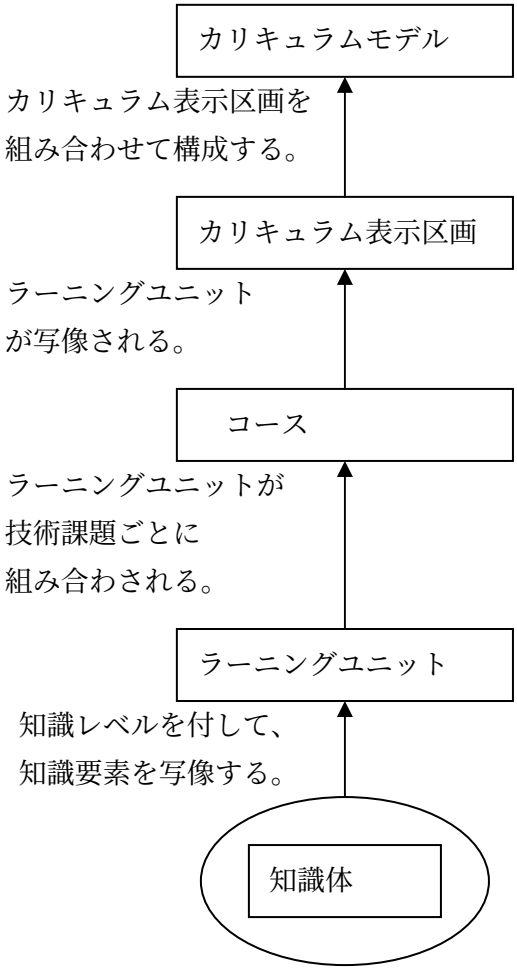

図１　カリキュラム設計に必要な要素とそれらの関係

現在、検討中のソフトウェアエンジニアリングカリキュラム策定のための基本的思想はつぎのとおりである。

(1)　知識体

知識体としては、SWECC で検討中のそれを、参考として取り入れる。

SWECC で検討中の知識体は、つぎの項目から構成されている。

- ソフトウェア構成管理（Software Configuration Management （SCM））
  - ソフトウェア構成管理プロセス
  - ソフトウェア構成の識別
  - ソフトウェア構成制御
  - ソフトウェア構成状態登録 (accounting)
  - ソフトウェア構成検査 (auditing)
  - ソフトウェア公開(release)管理及び提供(delivery)
- ソフトウェア構築（Software Construction）
  - 言語を用いたソフトウェア構築法
  - 数学を用いたソフトウェア構築法
  - 図式を用いたソフトウェア構築法
- ソフトウェア設計（Software Design）
  - 基礎となる思想、及び原理
  - 設計品質、及び計量法
  - ソフトウェアアーキテクチャ
  - 設計表記
  - 設計戦略、及び方法
- ソフトウェアエンジニアリング基盤（Software Engineering Infrastructure）
  - 開発方法論
  - ソフトウェアツール
  - 部品、及び統合
- ソフトウェアエンジニアリング管理（Software Engineering Management）
  - 管理プロセス
  - 計量
- ソフトウェアエンジニアリングプロセス（Software Engineering Process）
  - 基本思想、及び定義
  - プロセス定義
  - プロセス評価
  - プロセス実施、及び変更
- ソフトウェア改善、及び保守（Software Evolution and Maintenance）
  - 保守思想
  - 保守行為、及び役割
  - 保守プロセス
  - 保守体制
  - 保守に関わる課題
  - 保守費用、及びその見積もり

保守における計量
保守のためのツール、及び技法

- ソフトウェア品質（Software Quality）
  品質定義
  品質分析
- ソフトウェア要求技術（Software Requirements Engineering）
  要求技術におけるエンジニアリングプロセス
  要求抽出
  要求分析
  要求の正当性確認
  要求管理
- ソフトウェアテスト（Software Testing）
  基本思想、及び定義
  テストレベル
  テスト技法
  テストに関連する計測
  テストプロセスの組織化、及び制御
  テストの自動化

(2) 主専攻と副専攻

（大学院での専攻ではない。）

ソフトウェアエンジニアリング主専攻カリキュラムモデル、ソフトウェアエンジニアリング副専攻カリキュラムモデルを設けて、情報システムやコンピュータエンジニアリングを主専攻とする学生には、ソフトウェアエンジニアリング副専攻カリキュラムモデルを選べるようにする。

(3) カリキュラム表示区画

カリキュラム表示区画を設けて、カリキュラムモデルは、カリキュラム表示区画の逐次並び、及び並列並びの組み合わせから構成する。カリキュラム表示区画として、つぎのようなものを考える。

a. 総合人間系（人文及び社会科学、リベラルアーツ系科目、日本語、外国語、倫理を含む）
b. 総合理数系（数学、及び自然系工学の基礎）
c. コンピュータサイエンス（J97、および IEEE/ACM Computing Curriculum 2001 に対応）
d. ソフトウェアエンジニアリング序説
e. ソフトウェアエンジニアリングの理論と実践（開発・生産・保守・改善技術）
f. プロジェクト管理（組織論、ソフトウェアプロセス、コミュニケーション、ソフトウェ

アメトリクス（計量論)、ソフトウェア経済）
g．ソフトウェア基盤技術（開発環境、標準）
h．プロジェクト演習

これらのカリキュラム表示区画を、ひとつの案として、図2のように連結することによって、カリキュラムモデルを構成するものとする。この案については、今後、さらに研究を加える。

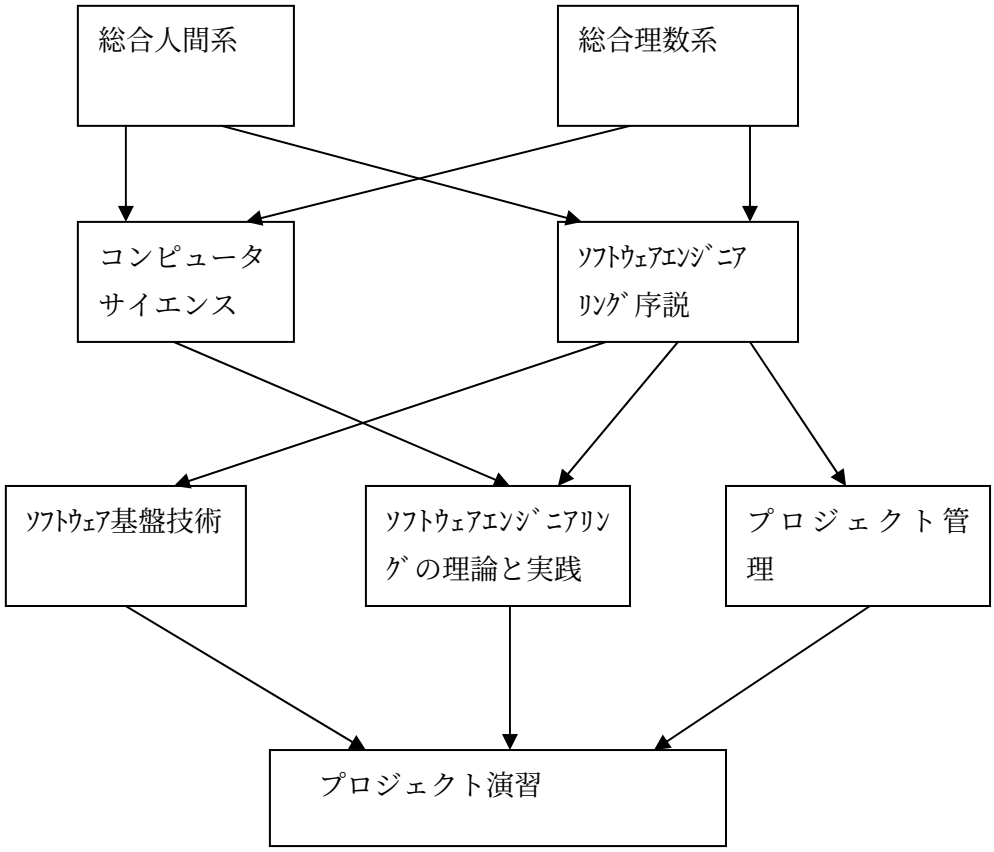

図2　ソフトウェアエンジニアリングモデルカリキュラムの一例

(3) ラーニングユニット

ラーニングユニットとしては、J97などの科目に含まれる、つぎの要素を設定する。

・J97/L科目

L-1 CS序説

L-2 プログラミング入門

・J97/M科目

情報代数
情報理論
M-2 離散数学
M-3 計算論
M-6 数理論理学
確率論

・J97/U 科目
U-1 論理回路
U-2 形式言語とオートマトン
U-3 データ構造とアルゴリズム
U-4 コンピュータアーキテクチャ
U-5 プログラミング言語論
U-7 オペレーティングシステム
U-8 コンパイラ
U-10 データベース
U-12 情報ネットワーク
U-19 ヒューマン／コンピュータ・インタフェース

・Development Process
ソフトウェアエンジニアリング概論（計量、エコノミー）
ソフトウェアプロセス（支援環境を含む）
システム／ソフトウェア要求分析、及び定義
形式的方法論
ソフトウェア設計論（手続き指向、及びオブジェクト指向）
ソフトウェアプログラミング
ソフトウェアテスト
・Supporting Life Cycle Process
品質保証
検証と確認
・Organizational Life Cycle Process
ソフトウェア構成管理
プロジェクト管理
・特別講義
マルチメディア技術
情報共通基盤

リアルタイムシステム／組み込みシステム
リエンジニアリング／ソフトウェア再利用
セキュリティ／安全性/ネットワーク犯罪

(4) 演習

・1 年生、及び 2 年生の間に、学内で、PSP(individual software process)演習を実施する。たとえば、1 年生では、コンピュータリテラシ、及びプログラミング演習（データ構造・アルゴリズム・ファイル操作など基本的なプログラムを作成）；2 年生ではハードウェアを含む、システム演習（デバイスドライバ・オペレーティングシステム・コンパイラなど）を対象として PSP を演習する。

・3 年生、または 4 年生では、「ソフトウェアラボ」、または「ソフトウェアハット」と称する、TSP 演習を実施する。これには企業の支援を得て、産学協同で実施することが望ましい。

・ソフトウェアラボ（またはソフトウェアハット）の構想

受講者を数名づつのグループに分け、各々をひとつの組織体（たとえば企業）に見立て、それぞれのチームは、同一要求仕様書を与えられて、同一目的の製品の計画から開発までを､同時に並行して､一定のソフトウェアエンジニアリングディジプリンに準拠して、スパイラル(spiral)プロセスモデルに準拠して実施する。この過程において、他チームの成果物を利用したり、提供したりすることによるエコノミー、及びチームとしてのコミュニケーションを体験する。さらに、できあがった結果を評価すること、評価するために必要な、原価、工程、品質、生産性など管理指標、及びそれらの計量に関する概念・方法についても学ぶことができる。

６． 情報システム学教育に対するカリキュラムのあり方

まだ緒についたばかりであるが、検討中の「カリキュラム策定のための基本構想」について述べる。

(1) 情報システム学カリキュラムの位置付け

情報システム学カリキュラムの位置付けを明確にするためには、コンピュータサイエンスやソフトウェアエンジニアリングのカリキュラムとの関係を明確にすることが必要であると考える。そのために、それぞれの学問分野の到達目標を明確にすることが望ましい。

この３つの学問分野の目標について、関係分野が相互に確認した現段階での表現は次の通りであるが、より適切な表現にするために、さらなる議論が必要であろう。

・コンピュータサイエンスが目指すもの：自動化、効率化

・ソフトウェアエンジニアリングが目指すもの：ソフトウェアにおける生産性向上、品質向上

・情報システム学が目指すもの：費用効果・便益（投資効果)、便益、顧客満足度

(2) 情報システム学必須カリキュラム案

情報システム学のプレゼンテーションエリアにサブエリアを設け、次のようなものをおく。

①情報人間術（コミュニケーション・チームワーク）
②システムの基礎
③情報システムの基礎
④問題把握とモデリング
⑤情報システムの計画と設計
⑥システム開発（実装）
⑦情報システム開発のプロジェクト管理
⑧情報システムの運営（維持、移行、評価）
⑨情報システムの専門性
⑩コンピュータサイエンス
⑪ネットワーク環境

以上のほかに、教養的基礎科目、あるいは参照学問領域における重要な関連科目があるが、これについては引き続き検討する。

(3) 情報システム学カリキュラムの構成

カリキュラムはコア領域と参照領域から構成する。さらに、参照領域は、人間・社会に関する部分と自然・技術に関する部分から構成する。カリキュラムには、各領域におけるサブエリア名、内容（レベル、時間数、単位数を含む）、養うべき能力、教授法の事例を明記する。

表 2　情報システムの必須カリキュラム案

	サブエリア	養うべき能力	教授法の事例
基礎	情報人間術（コミュニケーション＆チームワーク）	観察力、表現力、記述力、発表力、一般常識、一般倫理、マナー、協調性、行動力、責任感、計画性	現実の仕事を取り上げて、チームでインタビューして、人に読ませる図と文章としてのレポートにまとめる。学内発表と顧客レビューをしてもらう。計画を立てて役割分担、進捗を整理しながらグループ作業を実施。これを何回か繰り返す。
コア	システムの基礎	システム概念、システム事例、境界条件	自動販売機をシステム的に捉える。必要な情報および情報処理は何か！（マーケティング、対応、補給など）
	情報システムの基礎	IT による問題解決力、ツールの選択・活用力	（コンビニ、宅配などの）IS の事例研究。与えられた問題に対して具体的な IS を提案する（経営イノベーションとは何かを理解する）。インタビューの結果の中から一つの問題を抽出し、解決策を考えさせ、ツールを適用してみる。
	問題把握とモデリング	分析法、問題定義、モデリング	自動販売機のモデル化。モデリング手法の実習。ST、フローチャート、データモデル、DFD、UML、KJ、SSM、Work Design、スプレッドシート、図と文章など、ある問題に対して様々な手法を適用して対比、選択。RFP を書く。
	IS の計画と設計	仕様化、トレードオフ、品質設計、設計手法	設計化実装（分担開発など）→評価、フィードバック。サービス、環境の自主設計と運営。要件変更、改良／変更の提案。
	システム開発（実装）（インプリメンテーション）	開発手法、広義のプログラミング（コーディングだけではない）	
	IS 開発のプロジェクト管理	目標設定、資源管理, リーダシップ,計画立案	
	IS の運営（維持、移行、評価）	企画力、対応力、問題発見力、モラール	
	IS の専門性	倫理、組織、常識	契約書の作成と契約の実行。インターンシップ。

表 3　情報システムカリキュラムのコア領域と参照領域（2000/3/31 現在）

参照領域 （人間・社会）	コア領域	参照領域 （自然・技術）
	情報人間術（コミュニケーション&チームワーク）	コンピュータサイエンス ネットワーク環境
	システムの基礎 情報システムの基礎 問題把握とモデリング IS の計画と設計 システム開発（インプリメンテーション） IS 開発のプロジェクト管理 IS の運営 IS の専門性	

７．ＣＳ，ＳＥ，ＩＳの相関関係

ACM/IEEE-CS の Joint Curriculum Task Force は、CC2001 を策定するに当たって、それが基礎とする知識体が、ソフトウェアエンジニアリングを含む、各種の情報処理関連領域にも適用されることを予測して、その知識体のなかで、すべての関連領域に適用可能な中核的知識体として、「コンピューティングコア」という領域を設定した。CC2001 では、コンピューティングコアは、コンピュータサイエンス知識体(Computer Science Body of Knowledge: CSBOK)のなかに位置付けされている。

CSBOK の項目(topics)をつぎに示す。項目のなかで、*印を付した項目が、コンピューティングコア項目である。

DS. Discrete Structure

*DS1. Functions, relations, and sets

*DS2.Basic logic

*DS3.Proof techniques

*DS4.Basics of counting

*DS5.Graphs and trees

PF. Programming Fundamentals

*PF1.Algorithms and problem-solving

*PF2.Funcdamental programming constructs

*PF3.Basic data structures

*PF4.Recursions

*PF5.Abstruct data types

*PF6.Object-oriented programming

*PF7.Event-driven and concurrent programming

*PF8.Using modern APIs

AL. Algorithms and Complexity

*AL1.Basic algorithmic analysis

*AL2.Algorithmic strategies

*AL3.Fundamental computing algorithms

*AL4.Distributed algorithms

*AL5.Basic computability theory

AL6.The complexity classes P and NP

AL7.Automata theory

AL8.Advanced algorithmic analysis

AL9.Cryptographic algorithms

AL10.Geometirc algorithms

PL. Programming Languages

*PL1.History and overview of programming languages

*PL2.Virtual machines

*PL3.Introduction to language

PL4.Language translation systems

PL5.Type systems

PL6.Models of execution control

PL7.Declaration, modularity, and storage management

PL8.Programming language semantics

PL9.Functional programming paradigms

PL10.Object-orientd programming paradigms

PL11.Language-based constructs for parallelism

AR. Architecture

*AR1.Digital logic and digital systems

*AR2.Machine level representation of data

*AR3.Assembly level machine organization
*AR4.Memory system organization
*AR5.I/O and communication
*AR6.CPU implementation

OS. Operating Systems
*OS1.Operating system principles
*OS2.Concurrency
*OS3.Scheduling and dispatch
*OS4.Virtual memory
*OS5.Device management
*OS6.Security and protection
*OS7.File systems and naming
OS8.Real-time systems

HC. Human-Computer Interaction
*HC1.Principles of HCI
HC2.Modeling the user
HC3.Interaction
HC4.Window management system design
HC5.Help systems
HC6.Evaluation techniques
HC7.Computer-supported collaborative work

GR. Graphics
GR1.Graphic systems
GR2.Fundamental techniques in graphics
GR3.Basic rendering
GR4.Basic geometric modeling
GR5.Visualizaiton
GR6.Virtual reality
GR7.Computing animation
GR8.Advanced rendering
GR9.Advnced geometric modeling
GR10.Multimedia data technologies
GR11.Compression and decompression
GR12.Multimedia applications and content authoring
GR13.Multimedia servers and file systems
GR14.Networked and distributed multimedia systems

IS. Intelligent Systems

*IS1.Fundamental issues in intelligent systems

*IS2.Search and optimization methods

*IS3.Knowledge representation and reasoning

IS4.Learning

IS5.Agents

IS6.Computer vision

IS7.Natual language processing

IS8.Pattern recognition

IS9.Advanced machine learning

IS10.Robotics

IS11.Knowledge-based systems

IS12.Neural networks

IS13.Genetic algorithms

IM. Information Management

*IM1.Database systems

*IM2.Data modeling and the relational model

IM3.Database query languages

IM4.Relatinal database design

IM5.Transaction processing

IM6.Distributed databases

IM7.Advanced relational database design

IM8.Physical database design

NC. Net-Centric Computing

*NC1.Introduction to net-centric computing

*NC2.The web as an example of client-server computing

NC3.Building web application

NC4.Communication and networking

NC5.Distributed object systems

NC6.Collaboration technology and groupware

NC7.Distributed operating systems

NC8.Distributed systems

SE. Software Engineering

*SE1.Software processes and metrics

*SE2.Software requirements and specifications

*SE3.Software design and implementation

*SE4.Verification and validation
*SE5.Software tools and environments
*SE6.Software project methodologies
CN. Computational Science
CN1.Numerical analysis
CN2.Scientific visualization
CN3.Architecture for scientific computing
CN4.Programming for parallel architectures
CN5.Applications
SP. Social and Professional Issues
*SP1.History of computing
*SP2.Social context of computing
*SP3.Methods and tools of analysis
*SP4.Professional and ethical responsibilities
*SP5.Risks and liabilities of safety-critical systems
*SP6.Intellectual property
*SP7.Privacy and civil liberties
*SP8.Social implication of the Internet
SP9.Computer crime
SP10.Economic issues in computing
SP11.Phylosophical foundations of ethics

上に記載した CC2001 のなかの CSBOK になかで、ソフトウェアエンジニアリング、及び情報システム学に写像できる項目を選択し、表 4 に示した。SP は、CC2001 で新しく設けられた項目のひとつであり、１．で述べた、「新しい技術者像」に深く関わりをもつ重要な項目である。また、NC は、社会基盤の情報化という新しい動向に対応するために新しく登場した。しかし、IS(Intelligent systems)の一部、及び GR は、コンピューティングコアからは除かれている。これらのことは、わが国では、これから議論を必要とするところとなろう。数値解析を扱う CN がコンピューティングコアから除かれていることについては、さして議論にはならないであろう。

本調査・研究においては、わが国におけるソフトウェアエンジニアリング教育、及び情報システム学教育で考慮しなければならない知識体と、CC2001-CSBOK との関連を、表 4 にまとめて示した。

表 4　SE、CC2001、IS　のコア関係図

SE プログラム	CC2001		IS プログラム
	SE	IS	
総合理数系	DS	DS(1,2,5)	コンピュータサイエンス
コンピュータサイエンス	PF	PF	〃
〃	AL	AL(1-5)	情報システムの基礎
〃	PL	PL(1,3)	コンピュータサイエンス
〃	AR	AR	〃
〃	OS	OS	〃
SE の基礎知識	HC	HC(1)	情報システムの計画と設計
〃	GR	GR(10-13)	情報人間術
〃	IS*	IS*	――
〃	IM	IM(1-8)	問題把握とモデリング
SE の基礎知識 情報インフラ基盤技術	NC	NC(1-6,8)	ネットワーク環境
SE の理論と実践 大規模ソフトウェアの開発 ソフトウェア生産・保守技術	SE	SE(1-6)	情報システムの開発（実装）
――	CN	CN	――
総合人間系	SP(1,2,4)	SP(1-11)	情報システムの専門性
ソフトウェア組織論 コミュニケーション ソフトウェアメトリクス ソフトウェア経済	SP(3,5-8)		
――	――	――	システムの基礎 情報システム開発のプロジェクト管理 情報システムの運営

IS*：Intelligent Systems

8．むすび

この報告書に記載した内容は、情報処理学会・アクレディテーション委員会・ソフトウェアエンジニアリング分科会（平成 11 年 6 月発足)、及び情報処理学会・情報処理教育委員会・情報システム教育小委員会（平成 11 年 11 月発足）が、それぞれソフトウェアエンジニアリング、及び情報システム学におけるカリキュラムについて調査、研究した結果をまとめたものである。

図 1 で示したように、カリキュラム策定のベースになるものは、知識体と称する「学部学生に教えなければならない知識の内容」である。コンピュータサイエンス、ソフトウェアエンジニアリング、情報システム学の知識体を、それぞれ CSBOK(Computer Science Body of Knowledge), SEBOK(Software Engineering Body of Knowledge), ISBOK(Information Systems Body of Knowledge)とよぶとすれば、この 3 者は、互いに重なりをもっている。

すべての知識体（CSBOK、SEBOK、ISBOK）が共有する重なりは、中核（コア）知識とよばれ、CC2001 のなかで定義されている。しかし、2 者のみの重なり、すなわち SEBOK と CSBOK の重なり、ISBOK と CSBOK の重なり、及び SEBOK と ISBOK の重なりは、今後の研究課題として残されている。

IS 学においては、IS'97 によって、知識体、ラーニングユニット、知識体要素をラーニングユニットへ写像するために用いる知識レベル、コース、及びカリキュラム・プレゼンテーションエリアが公開されている。したがって、情報システム教育小委員会は、これを参照しながら、我が国での推奨カリキュラムモデルを作るための活動を継続して行うこととしている。

一方、ソフトウェアエンジニアリングにおいては、ACM/IEEE-SWECC から、ソフトウェアエンジニアリング認定基準、ソフトウェアエンジニアリング倫理要綱、及び中間的な SEBOK(Stone Man)が公開されたが、これ以外のものは策定中で、2001 年春から夏までは公開されない。アクレディテーション委員会・ソフトウェアエンジニアリング分科会としては、JABEE が 2000 年度に推進する、認定試行に備えて、ソフトウェアエンジニアリング教育プログラム認定基準、ソフトウェアエンジニアリング倫理要綱、知識体、カリキュラムモデルを策定する必要があり、SWECC が策定する各種の成果物すべての公開を待っていることが許されない。したがって、JABEE 試行の進展に歩調を合わせて、我が国独自のカリキュラムモデルを策定し、2001 年度に入ってから、SWECC の公開物を参照し、今一度見直す予定で、必要な策定を行うこととしている。

引用文献

- ソフトウェアエンジニアリング関連の引用文献の主なものは、情報処理学会・アクレディテーション委員会・ソフトウェアエンジニアリング分科会のホームページ（http://Acc-SE.kudpc.kyoto-u.ac.jp/）にリンクされている。
- CC2001 は、IEEE Computer Society のホームページ(http://www.computer.org/)にリンクされている。
- ACM/AIS/AITP が作成した、IS'97「情報システム学の学部用プログラムのためのモデルカリキュラムと指針」の和訳は、情報処理学会 HIS 研究会で、1998 年 12 月に作成され、提供されている。