第14回

ソフトウェア工学教育の確立へ向けて

松本 吉弘
（武蔵工業大学）
yhm@sft.cs.musashi-tech.ac.jp

早朝5時，2カ月に1度，アメリカの電話交換手がIEEE/ACM国際電話会議の開始を告げ，電話をつないでくれる．電話に出ると，すでにIEEE/ACM Computing Curricula 2001/Software Engineering Steering Committee（以下，CCSEと略す）メンバの何人かが電話機の向こうに集まっている．定数が集まると，司会者の誘導で，電話会議が始まる．このような国際電話会議がACMの費用で何回も繰り返された挙句，ようやくIEEE/ACM Software Engineering Curriculumの最終原案（これから正規の承認手続に入る）がまとまった（以下，CCSEドラフトと略す）．CCSEのサイト（http://sites.computer.org/ccse）に掲載されている．このなかには，情報処理学会アクレディテーション委員会でまとめた「Jpn1」と称する日本向けカリキュラムモデルも，カリキュラム例の1つとして記載されている．Jpn1は，CC2001（Computing Curricula 2001），JABEE☆1および，ABET☆2から与えられた制約要件を満たしたカリキュラム例の1つであり，Jpn1に対する日本語による説明は，http://www.ipsj.or.jp/12kyoiku/acre/Acc-SE/index.htmlにある．以下，本稿では，software engineeringをソフトウェア工学と訳す．

■IEEE/ACM Software Engineering Curriculum

CCSE会合では，ソフトウェア・プロフェッショナルに対する学部での計算機科学（CS）教育に対して，実業界から強い不満（たとえば，文献1））の烽火が上がり，D.L. Parnasなどからソフトウェア工学教育は計算機科学教育から独立させるべきである，という主張がされた[2]．計算機科学を代表して参加していたACM代表者から，これに抗する激しい議論が展開されたが，結局のところ，独立させることになった．しかし，この議論の結果，CCSEドラフトは，マネジメント色の強い（市販の各種図書によく見られる），いわゆるソフトウェアエンジニアリングとは違って，制約下での問題解決法，デザイン手法，大規模プログラミング能力，ソフトウェア共通基盤の構築能力，ユーザインタフェースの構築能力など，工学に重心を置いたものになった．

■ソフトウェア工学カリキュラムにおけるコア知識と最低時間数

CCSEでの作業は，おおよそ次のような順序で進んだ．はじめに，ソフトウェア工学教育知識（SEEK：software engineering education knowledge）を定義し，それを**表-1**に示す10のグループに分けた．次に，各グループが必要とするコンタクトアワー（授業／演習で教員が学生に接する時間）の最低水準を決めた．さらに，各グループを構成する各知識要素を，E: essential，D: desirable，O: optionalのなかのどれかに格付けした．Eに格付けされたコア知識要素は，ソフトウェア工学コア科目（必須科目）のなかに必ず含めなければならない，とした．ソフトウェア工学教育プログラム（日本のコースに相当）の授業／演習科目のなかに散りばめられている各知識要素に対するコンタクトアワー数の総和が，知識要素グループごとに，表-1に示した時間数を超えることが要求される．表-1にSEEKを構成する10の知識要素グループとそれぞれが必要とするコンタクトアワー数の最低水準を示す．

CCSEのなかで行われた主な議論を次にまとめる．

（1）定義されたアウトカムズを得ることを目標とした教育・訓練を行う（アウトカムズは，当該プログラムの利害関係者，および外部評価組織によって計量され，アウトカムズ目標を達成するように，自律的に教育プロセス改善に用いられなければならない）．
（2）設計を中心とした，工学教育・訓練に重点を置かねばならない．
（3）プロジェクトマネジメントなど，管理に必要な知識（たとえば，表-1のEVL，PRO，QUA，MGT）の教育は必要最低限にとどめ，あとは生涯教育にゆだねる．
（4）Capstoneプロジェクト（我が国の卒業研究に対応する）を充実し，これに対する企業の支援を積極的に要請する．
（5）工学の他の専門分野の学科に属する学生に対しても門戸を開放し，ソフトウェア工学教育プログラムでの履修，および修了認定取得が可能であるようにする．
（6）ソフトウェアは国際的流通財であるため，国情の

☆1 Japan Accreditation Board for Engineering Education（日本技術者教育認定機構）．
☆2 Accreditation Board for Engineering and Technology

記号	知識要素名	必要時間数	説明
CMP	Computing Essentials	172	CS，形式的方法論，コンパイラ，ミドルウェアなどの構築，構築ツール
FND	Mathematical & Engineering Fundamentals	89	数学基礎，設計論など工学基礎，エンジニアリング経済
PRF	Professional Practice	35	グループダイナミクス，グループ心理，コミュニケーション，情報倫理，知的所有権
MAA	Software Modeling & Analysis	53	モデル基礎，モデル化，システムおよびソフトウェア要求
DES	Software Design	45	ソフトウェア設計法，アーキテクチャ論，ヒューマンコンピュータインタフェース，設計評価
VAV	Software V&V	42	検証論と妥当性確認法，テスティング
EVL	Software Evolution	10	ソフトウェア進化論，新しい保守のあり方
PRO	Software Process	13	プロセス論，成熟度モデル
QUA	Software Quality	16	品質論，品質保証
MGT	Software Management	19	プロジェクト計画，制御，管理，構成管理

表 -1　SEEK で示されたコンタクトアワー数の最低水準

差があっても，すべての国がこのCCSEドラフトに準拠した学部教育を実施できるよう内容を配慮する．CCSEとしては，ABETによる教育認定やワシントン協定による国際的同等性相互承認には直接言及しないが，将来，ソフトウェア中心システムの国際入札などにおいて，担当技術者の修了した大学での教育プログラムの国際的同等性が参照される可能性もある．

数回にわたって公開したCCSE中間ドラフトに対する世界各国の教員，企業人からの意見をWeb上で受け付け，それぞれ対して回答を発し，これら意見を加味した国際電話会議（冒頭に紹介）を開き，ドラフトの改訂を10 回近く行った．この経過は，前記，CCSEのサイトに記録されている．

■ソフトウェア工学コミュニティ作りが大切

ソフトウェア工学教育・訓練を確立するためには，教育カリキュラムを細かく検討し，改善することも必要であろうが，これ以上に，ソフトウェア工学を魅力ある存在に育てることが必要である．ABETは，教育認定の対象とする領域を選定する際，その領域に対するしっかりしたプロフェッショナル・コミュニティが築かれていることを 1 つの条件としている．筆者が若いころ米国のソフトウェア工学コミュニティに参入できたのは，第 1 回のチューリング賞受賞者である，故Alan Perlis先生（当時Yale大学）に負うところが大きい．先生は，筆者が 1970 年代に公に発表した論文のいくつかを知らぬ間に読んでいて，1983 年 9 月に，30 名前後の著名な情報科学者だけを集めて有名リゾート地で開かれるワークショップに，突然招待してくれ，講演をさせてくれた[3]．以後，アメリカ各地の大学での特別講義を担当することができ，Software Reusabilityという領域では，弊論文が多く引用されるようになった．筆者の業績は，計算機科学とはほど遠いもの（ハードウェア生産技術をソフトウェア生産技術に発展させるための基本思想に関するもの）であった．若手技術者に対して，既成領域にとらわれることなく，幅の広い登竜門を提供することが，新しい領域に若手をひきつけ，その領域の興隆のために必要であることを知った．現在，アメリカなどでは，遺伝子工学，脳計算論，システム生物学（Systems Biology）などの持つディシプリンに基づいてソフトウェアを組み立てることを目指したソフトウェア工学研究が着手されている[4]．既成領域からかけ離れたディシプリンであっても，面白く，かつソフトウェア工学領域に対してブレークスルーをもたらす可能性がある研究を軽視してはならない．しっかりしたソフトウェア工学コミュニティを築き，若手研究・技術者に広く国際的な登竜門を提供するための努力を行うことが必要であろう．

参考文献
1）Leathbridge, T.C.: What Knowledge is Important to a Software Professional?, IEEE Computer, Vol.33. No.5, pp.44-50 (2000).
2）Parnas, D.L.: Software Engineering Programs Are Not Computer Science Programs, IEEE Software, November-December, pp.19-30 (1999).
3）Perlis, A. (ed.): Proc. Workshop on Reusability in Programming, ITT Programming (1983).
4）Poore, J.H.: A Tale of Three Disciplines and a Revolution, IEEE Computer, Vol.31, No.1, pp.30-36 (2004).

（平成 16 年 5 月 7 日受付）