2007年5月28日発行
第3巻第2号（通巻10号）
ISSN 1349-8622

SEC journal 10

Software Engineering Center

特集

SEC2006年度活動概要

IPA
独立行政法人 情報処理推進機構
http：//www.ipa.go.jp/

SEC journal
Software Engineering Center
No.10目次

1 巻頭言
大竹伸一（関西経済同友会　ソフト産業振興委員会委員長）

所長対談：Allen Brown　The Open Group 社長
2 企業情報システムにおける「境界のない情報の流れ」の構築に向けて

6 特集　SEC2006年度活動概要
8 エンタプライズ系活動概要
9 組込み系活動概要
◆エンタプライズ系
10 「共通フレーム2007」の概要
12 要求工学、設計・開発技術活動概要
14 見積り手法（改造型開発）
16 ITプロジェクトの「見える化」上流工程編
18 プロセス改善研究
20 定量データ分析
◆組込み系
22 組込みソフトウェアエンジニアリング領域の状況
24 組込みソフトウェア開発のプロセス面の支援
26 組込みソフトウェア開発のプロジェクトマネジメントの支援
28 ツール諸元表の策定
30 組込みスキル標準（ETSS）
◆共同研究
34 COSEにおけるデータ分析とフィードバック
35 プロジェクトの遂行及び生産性に影響を与える要因の分析
36 相関ルールマイニングを用いたソフトウェア生産性の決定要因抽出
37 異なるFP手法間におけるFP変換式の導出
38 推定・近似に基づいた機能規模計測法間での変換法
39 相互比較可能な形式へのFP値の変換方法に関する検討
40 先進ソフトウェア開発プロジェクトにおけるソースコード分析
41 組合せ・すり合わせ視点での組込みソフトウェア分析調査
42 形式検証技術の設計検証への実用化に向けて
43 ETSS向け教育研修コースを対象とした評価フレームワーク
44 品質モデル適応型テストプロセスの研究
45 組込みソフトウェア教育の実施効果に関する調査
46 コンポーネントベース高信頼性実時間組込みシステム構築技術の研究
47 性能指向組込みソフトウェアの研究:並列化、ハードウェア化を目的としたC言語記述に関する記述作法

技術解説
48 工期の厳しさに関連する要因の分析
門田暁人 ,馬嶋宏 ,増田浩 ,羽田野尚登 ,磯野聖 ,内海昭 ,菊地奈穂美 ,服部昇 ,細谷和伸 ,森和美
54 ハードコンシャス記述C:ハードウェア化を目的としたC言語記述作法
天野英晴

組織紹介
62 The Open Group
藤枝純教

64 BOOK REVIEW
65 ソフトウェア・エンジニアリング関連　イベントカレンダー
66 あとがき
67 お知らせ（論文募集／SEC journal バックナンバー）

巻　頭　言

大阪・関西を組込みソフト産業の一大集積地に！

関西経済同友会
ソフト産業振興委員会委員長
大竹　伸一
（西日本電信電話株式会社
代表取締役常務取締役）

高まるソフトウェアの重要性

近年の日本経済の発展を牽引している自動車、情報家電、携帯電話等の機能や性能は、搭載されるソフトウェア（組込みソフトウェア）の品質・性能に大きく依存し始めており、今後、その開発需要と重要性は、ますます拡大すると予測される。しかしながら、経済産業省の報告によると、日本の組込みソフトウェア技術者は９万人以上も不足しており、昨今の開発規模の巨大化・複雑化も相俟って、組込みソフトウェアに関するトラブルが急増する等、企業経営への影響も顕在化しつつある。

東京一極集中が進むソフトウェア産業

経済産業省の調査によると、ソフトウェア産業（情報サービス業）の地域別の年間売上高は、東京が9兆526億円と全体の6割強を占めており、大阪は9,209億円（全国比6.3%）と東京の売上高の10分の1しかない。事業所数では、東京が全体の約3割を、従業者数では東京が全体の5割以上を占めており、ソフトウェア産業の過度の東京一極集中が進展している。

大阪・関西圏の役割と活性化

今後、組込みソフトウェアはさらに色々な分野で利用され、製造業における金型と同様に産業を支える重要な基本技術となっていく。一方で大阪・関西には、優秀な大学、時代の先端をいく情報家電メーカ、情報系中小企業、専門学校が集積しており、ソフトウェア産業に対するポテンシャルが高い。さらには、ソフトウェア開発のグローバル化が進む中、アジアとの交流が深い大阪・関西には大きなアドバンテージがある。我々は、これらの強みを最大限に生かし、大阪・関西を組込みソフトウェア産業の一大集積地とすることで、関西地域の経済活性化はもちろん、日本の産業力強化にも貢献していきたいと考えている。

関西経済同友会の提言

こうした基本認識のもと、弊会のソフト産業振興委員会では、大阪・関西への組込みソフト産業集積に向けた提言を３月に発表しており、その内容をご紹介したい。

まず、この課題に取り組むためには、産官学の連携と強力な支援体制が必須であることから、そのためのしくみとして『大阪・関西を組込みソフト産業の一大集積地とするための推進エンジンとなる「組込みソフト産業推進協議会」（仮称）を設立する』ことを提言した。協議会では今後、組込みソフト技術者の人材育成や、メーカ各社からの発注の活発化に向けたソフト会社の技術力・開発品質の見える化等を実現するための資格認定・評価制度の導入について検討を進めていく予定である。

次に日本の労働力人口が減少する中で、不足する組込みソフト技術者を確保するためにも、アジアの親日国家との協業を積極的に進め、激化する国内外の開発競争に打ち勝たなければならないとの想いから『大阪・関西が組込みソフト産業におけるアジアの牽引役を担えるよう、産官学の連携により、組込みソフト技術者を志すベトナム等アジアの留学生や、アジアの組込みソフト会社のサテライト拠点を大阪・関西に誘致する』ことを提言した。

今後、これら提言の実現に向け及ばずながら力を尽くしていく所存である。

SECとの連携

以上の取り組みに加え、今後は、標準化の推進、各種セミナの開催や展示会の誘致等、多面的な取り組みが重要であり、これらの実現に向けてはSECからの支援・協力が必要不可欠であると考える。4月には、その第一弾としてSECの協力により、大阪でセミナを開催していただいたが、今後も更なる連携強化を図っていきたい。

所長対談 Allen Brown　The Open Group 社長

企業情報システムにおける「境界のない情報の流れ」の構築に向けて

企業活動の複雑化に伴って、企業情報システムには、企業内のみならず、パートナをも含めた「境界のない情報の流れ（Boundaryless Information Flow）」を実現することが求められている。その実現に欠かせないのがEA（Enterprise Architecture、エンタープライズアーキテクチャ）である。日本でも、政府がEAをベースとして各省庁の情報システム構築に着手するなど、EAに対する関心が高まっている。ITサプライヤとユーザからなる非営利団体オープン・グループは早くからこのEAに関する検討を開始し、その成果をTOGAF（The Open Group Architecture Framework、トガフ）として公表している。TOGAFはオープン・グループのWebサイトで閲覧することができ、そのダウンロード数は2万件に達しているという。企業や政府組織における情報システムの問題と解決手法についてオープン・グループのアーレン・ブラウン社長とIPA/SECの鶴保征城所長が語り合った。

鶴保　征城（つるほ　せいしろ）
1966年大阪大学大学院工学研究科電子工学専攻修士課程修了後、日本電信電話公社（現NTT）入社。NTTソフトウェア研究所長、NTTデータ通信株式会社取締役開発本部長、同社常務取締役技術開発本部長、NTTソフトウェア株式会社代表取締役社長を歴任し、2004年10月独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター所長に就任。

・社団法人 情報処理学会 会長（2001年～2002年）
・XMLコンソーシアム 会長（2001年～）
・高知工科大学工学部情報システム工学科 教授（2003年～）
・日本BPM協会 副会長（2006年～）
・実践的ソフトウェア教育コンソーシアム 会長（2006年～）
・社団法人 電気情報通信学会 フェロー
・社団法人 情報処理学会 フェロー

鶴保　エンタープライズ系分野においても組込み系分野においてもソフトウェアの重要性が増大しています。近年、開発すべきソフトウェアの巨大化が急速に進行し、しかも、そのソフトウェアは短期間にしかも高い品質・信頼性を確保して開発することが求められています。

本日は、ソフトウェア開発が抱えるこうした問題とその問題の解決策について議論を深めたいと考えています。初めに、オープン・グループの活動についてお話しいただきたいと思います。

ブラウン　オープン・グループは、ITの標準化活動を行っている非営利団体で、1996年2月に誕生しました。オープン・グループのメンバーは、ITのサプライヤとカスタマからなり、グローバルにわたっています。まず、オープン・グループのビジョンについてお話ししましょう。

1993年ごろに、オープン・グループの議論において、カスタマ側から「EAに関する新しい標準が必要である」という意見が提出されました。当時、その考え方は新しいものであり、EA標準に求められる内容がまだ具体的ではありませんでした。そこで、オープン・グループは様々なミーティングを行い、「PoC（Proof of Concept）」「IBM Open Blueprint」「TAFIM（Technical Architecture for Information Management）」の3つをスタートポイントとしてアーキテクチャのフレームワークを作成する活動を開始しました。

TAFIMは米国防総省によって作成されたアーキテクチャですが、オープン・グループのメンバーによるミーティングにおいて、EA標準のスタートポイントに最も適しているという合意がなされました。これを受けて、国防総省はTAFIMをオープン・グループに受け渡すことに合意し、オープン・グループがTAFIMをもとにしてEAの標準化の活動を進めてきました。その成果が、1996年に最初のバージョンとして公表されたTOGAFです。

TOGAFは、「フレームワーク」「アーキテクチャ開発メソッド」「リソースベース」を3本柱とし、2002年までにテクニカルエディションと呼ばれるバージョン7が作成されました。最新のバージョンはエンタープライズエディションと呼ばれるバージョン8であり、ビジネスモデルを取り入れた内容になっています。

企業は部門横断的、さらにパートナ企業とコラボレートして仕事をしているという状況にあるのに対して、情報システムのアプリケーションや情報は各部門内、あるいはパートナ企業に閉じています。その点を改め、必要な人材が必要な情報を取得できるようにアプリケーションや情報を統合することの必要性を、オープン・グループのカスタマグループが「インターオペラブルエンタープライズ」という白書にまとめました。そして、これを実現するためには、第一段階として企業内の情報の流れの相互運用性を図り、第二段階で企業間の相互運用性を確立する。そして、第三段階でグローバルな相互運用性を確立すると

いうステップを踏む必要があります。その実現を目標として、オープン・グループではEAのコンセプト作りに取り組み、その成果がTOGAFバージョン８に結実しました。

オープン・グループのミッションは、TOGAFに象徴されるように、グローバルな相互運用性を介して高いセキュリティと信頼性を確保し、同時にタイムリーに境界のない情報の流れを実現することです。

鶴保　日本でも国レベルでオープンシステムを重視しています。その典型は、EAを国のシステムに適用しようという動きです。EAは国自身がカスタマです。そして、カスタマ側のエンジニアの少なさを補うためにCIO補佐官（CIO：Chief Information Officer）を任命しています。CIO自身は官公庁のトップクラスが務め、CIOを補佐するCIO補佐官を民間から雇用するという体制で、EAにもとづいて政府機関の情報システムを構築する動きが始まろうとしている段階です。

企業向けアーキテクチャのカギを握るのは オペレーションのサポート

ブラウン　政府がEAに取り組む上で有効な方法の1つが先のTOGAFです。また、DoDAF（Department of Defence Architecture Framework、米国防総省が体系化したEAのためのフレームワーク、ドダフ）のドキュメントを検証し、それをTOGAFと関係付けて見ていくことが日本政府にとって非常に参考になるのではないかと思います。DoDAFは政府のための強力なアーキテクチャフレームワーク（FEA：Federal Enterprise Architecture）であり、DoDAFとTOGAFを一緒に使うことによって日本政府のEAは非常に強力なものになると考えられます。

鶴保　FEAとTAFIM、そしてDoDAFはそれぞれどのような関係があるのでしょうか。

ブラウン　まず、TAFIMからFEAが作成されました。その背景には、国防総省を含めて24の省庁においてFEA導入の大統領指示が出されたことにあります。それに対して国防総省は、「FEAは管理中心の仕組みであり、管理中心のFEAのままでは軍のオペレーションを遂行できない。オペレーション主体の仕組みが必要だ」と考えたのです。そのニーズに応えるべく、オープン・グループがDoDAFを作成することにしました。DoDAFはFEAの精神を受け継いでいますが、もっとオペレーションにフォーカスしているアーキテクチャフレームワークです。同様に、ビジネスモデルや企業の活動に対応するためのカギを握るのは、管理だけではなくオペレーションをサポートすることです。ですからDoDAFとTOGAFは近い関係にあります。

Allen Brown
1993年以来The Open Groupに在籍。X/Open Company Limitedのチーフ・フィナンシャル・オフィサー兼ビジネス開発担当副社長を経て、1994年には、チーフ・オペレーティング・オフィサーを務め、X/OpenとOSFとの統合化に積極的に参画。統合後、シニア・バイス・プレジデントとして統合活動を推進、1998年には執行役員社長兼CEO、同年The Open Group社長兼最高経営責任者に就任。

鶴保　日本の場合、EAの導入が必ずしも成功しているとはいえない状況です。私は、その要因について分析する必要があると考えています。まず、EAが使いやすいものになっているのかどうか。その点を分析すべきでしょう。さらに、CIO補佐官を省庁に1～2名採用するだけでEAの導入が果たしてうまくいくのかどうか。そのような組織上の問題、人間の問題があります。米国の場合、軍のほうはもともとエンジニアが非常に多かったと思いますが、各省庁はどうなのでしょうか。各省庁ではみずからエンジニアを雇用しているのですか。それとも、ITベンダの力を借りているのですか。

ブラウン　その2つの方法を組み合わせています。例えば、EAの領域でITベンダを使うケースもあります。そして、EAの導入を成功させるためにいくつか念頭に置いておくべきことがあります。EAを導入することは文化を変えるに等しい非常にチャレンジングなテーマであるということです。エンドユーザの側が、何が問題となっているかを明確に認識していれば、文化を変えるというテーマに対応することは難しくはありません。しかし、企業にはレガシーシステムを管理している人がいます。何かを変えていこうとすると、彼らが抵抗勢力になります。米国の省庁は、1996年に制定されたクリンガーコーエン法によって、EAが求められています。このクリンガーコーエン法が成功しているのかどうか、その点に関して米国では大きな議論が交わされているところです。EAの導入は時間がかかる大きなタスクです。それにもかかわらず、企業サイドからはすぐに結果を出すことが求められます。そのことが議論を呼び起こす背景となっています。

鶴保　エンタープライズシステムは、ビジネスの複雑化と迅速化の影響を受けています。そのために、複雑化、巨大化するソフトウェアをいかにスピーディに開発するかが大きな課題となっています。自動車のソフトウェアを例にソフトウェアのステップ数を見ると、SECが発足した2年半前にはトータルで500万ステップ弱でしたが、現在は700万～1,000万ステップほどになり、エンジンの制御を含めると2,000万ステップ近くになっています。エンタープライズ系システムも同様に拡大の傾向が

あります。高い信頼性が求められるソフトウェアを厳密に管理し、これからも大量に作り続けるためには、ソフトウェアの作り方自身を変える必要があると考えています。そのカギを握るのは、ソフトウェア開発プロセスの改善です。例えばボーイング社は、ボーイング777に搭載された700万ステップのソフトウェアを4年かけて開発したのに対して、後継機の787型機の4,000万ステップにのぼるソフトウェアの開発をわずか1年で終えているのです。その理由は、ボーイング社がソフトウェア開発プロセスの徹底した改善を進めたからです。

ブラウン　企業の情報システム構築・運用が抱える問題の解決にTOGAFは大きな効果を発揮します。TOGAFは、最初にアーキテクチャが持つべき原理や原則、ビジョン、網羅する範囲を特定します。次に、イニシエーションからビジネスアーキテクチャ、インフォメーションアーキテクチャから見ていき、マイグレーションと実装を行います。その際に、各フェーズにおいて前のフェーズに戻って確認するという反復的な方法をとっています。例えば、インフォメーションアーキテクチャの設計を行っているのであれば、インフォメーションアーキテクチャの設計が終了した段階でビジネスアーキテクチャに戻って相互の検証を行います。

また、TOGAFのサブプログラムのようなツールとして位置付けられるものにビジネスシナリオという考え方があり、これが非常に有用です。このビジネスシナリオは小規模なスコープのもので、ワークグループという環境の中ですべての課題を非常にスピーディに把握することが可能です。その利点の第一は多岐にわたる利害関係者を取り込むことができること、第二にクイックウィン、つまり早く成果を得ることができることです。

左より、SEC所長鶴保征城、The Open Group社長Allen Brown、オープン・グループ 日本代表藤枝純教。

藤枝純教（ふじえだ　じゅんきょう）：1961年京都大学文学部卒。日本アイ・ビー・エム本社マーケティング統括、CSK取締役を経て、現在信州大学大学院客員教授、グローバル情報社会研究所等を兼任。

鶴保　日本では、現場中心主義の考え方が強く、それによって企業のシステム化が成功してきました。しかし、その一方で、部分最適が進行しすぎたきらいがあります。部分最適を全体最適に持っていくことが組織の課題です。例えば、日産自動車のようなグローバル企業は全体最適になりつつあるようです。しかし、最も全体最適になることが求められるのは、部分最適なIT資産を膨大に持っている官公庁だと思います。

EAを導入するためには組織本来の目的から始める必要がありますが、私は、本格的なアーキテクチャを導入しなくても、政府組織が使いやすく「境界のない情報の流れ」を実現する手法が求められているのではないかと考えています。

エンジニアのプロ意識を高めることも大切

鶴保　オープン・グループはアーキテクチャの標準化に熱心に取り組んでいるという認識は以前から持っていたのですが、ソフトウェアの品質保証というテーマについてはどのような取り組みをされているのですか。

ブラウン　基本的にソフトウェアの品質確保の背景にあるのはテストです。UNIXアプリケーションやLinuxアプリケーションの認証に用いるテストソフトウェアをオープン・グループで持っています。そのテストソフトウェアをベンダのソフトウェアに適用するということからスタートしています。大切なことは適用の仕方です。オープン・グループとしては、UNIXやセキュリティ、アーキテクチャの標準化と同じ標準認証プロセスをソフトウェアの品質確保にも適用しています。

ただし、それだけがアプローチではありません。もう1つ中心にあるアプローチは、ITの実装にかかわるエンジニアたちのプロ意識を高めること、個人のレベルを高めることです。もう少し具体的にいうと、実際のITプロジェクトでは、プロジェクトがうまく進んでいない場合、原因を指摘しないケースが多々あります。そうした原因をキチッと指摘できるようエンジニアのプロ精神を高めることも重要なアプローチです。そのためには、行動規範を作成すること、また問題点を上部に上げていく方法の確立が求められます。

鶴保　SECでもソフトウェアエンジニアの強化に大きな関心をもっています。大学の講義とソフトウェア開発の現場が必要とする知識にギャップがあります。ソフトウェア開発の経験を豊富に持っている人材を大学に送り込んで、そのギャップを解消する取り組みが文部科学省を中心に推進されています。

ブラウン　オープン・グループはIT人材の認定制度や標準準拠に関するサービスも提供しています。IT人材の認定制度として

はTOGAFを使用する能力のある人を認定する制度を持っています。認定者数は全世界で2,000名を超えており、日本は150名です。TOGAFと同様に、ITAC（IT ARCHITECT Certification）という個人の技術者がもつスキルあるいは経験を評価して認定するプログラムもあります。

鶴保　TOGAFの認証とITACの認証との関係はどのようなものなのでしょうか。

ブラウン　TOGAFの認証アーキテクトは、TOGAFの中身や方法について4～5日学習したうえでケーススタディを学び、OKが出たらTOGAFのメソドロジーを使う能力があると認証されます。ITACは、過去にどれだけのプロジェクトを経験したかを踏まえてITアーキテクト個人のスキルを認証するものです。オープン・グループの会員となっているITベンダはそれぞれITアーキテクトを評価する認証プログラムを持っているのですが、その内容が非常に似ていること、そして中立の第三者機関がスキルに対して認証を与えたほうがユーザから信頼が得られるということで、オープン・グループがITACとして認証しています。TOGAFとITACの2つの認証を受けている技術者も存在しますが、TOGAFの認証を取得していなければ、ITACの認証を取得できないというわけではありません。

ソフトウェア品質管理の標準策定へ向けて

ブラウン　オープン・グループはEAの標準化活動に加え、高い堅牢性が求められる組込みソフトウェアの分野において「リアルタイムエンベデッドシステムフォーラム」という組織を設けてJavaの開発プロセスを作成する活動に取り組んでいます。その成果はJavaコミュニティプロセスにサブミッションとして受け入れられています。

鶴保　日本でも政府のR&D政策としてリアルタイム組込みOSとしてセキュアOSを開発するというプロジェクトが2つ進められています。1つは、経済産業省のプロジェクトで、バーチャルマシンを使ってよりセキュアなシステムを開発して攻撃に強いシステムを作ることを目指しています。もう1つは、文部科学省が進めているディペンダブルな組込みOSを作るプロジェクトです。両者はともに、国の政策としてリアルタイム組込みシステムを重視していくという流れの中で開発がスタートしています。開発にあたっては、産業界のみならず、大学など学の知見も結集する産学連携によって進められることになっています。

ブラウン　オープン・グループに設けられたリアルタイムOSグループは5年かけてOSのオープン標準を作成しました。そのベースにあるのは、UNIXのカーネルを土台としたPOSIXの標準化です。その標準化では、Javaの信頼性とパフォーマンスを修正するレコメンデーションをオープン・グループが提出し、Javaコミュニティで承認されました。

その次にオープン・グループがスタートさせたのが「ディペンダビリティ＆アシュアランスプログラム」です。OSの標準ができ、Javaも強力なものとなったのですが、全体をつないだときの品質管理や検証をだれが担うのかという問題が残されていました。ただし、その課題はリアルタイムOSに限ることではありません。そこで、フォーマルメソッドをベースにした要求仕様の手法とプラクティスに関する標準を策定しようと、リアルタイムグループが中心になりオープン・グループ全体で動いているところです。

鶴保　それは非常に興味深い話ですね。日本もいろいろな施策がスタートしています。日本の場合、総理大臣がトップを務める総合科学技術会議という組織があるのですが、その下に位置する実践組織の機能に少し弱いという懸念があります。産学官の力を結集して総合的なディペンダブルシステムを開発するという動きは少し弱いところがありますね。

最後に一言申し上げたいことがあります。それは、先ほども申し上げましたが、これからはソフトウェアの作り方を変えなければならないということです。ソフトウェア開発には、要求定義、設計、インプリメント（実装）という流れがありますが、その流れがうまくつながらずに切れていることが多いのが現状です。そこで、要求定義等の上流で書いたものがきちんと下流のプログラミングに流れていく一貫した仕組みを確立しなければなりません。そのためには、言語や方法論の問題をばらばらに解決しようとするのではなく、相互に知見を持ち寄ってコンセンサスを作っていくことがいいと考えています。我々は人類始まって以来の問題に直面しているといえます。IPA/SECとしても日本国内に閉じて活動しようとは思っていません。米国や日本という枠にとらわれず、グローバルな連携によって優れたソフトウェア開発の手法を確立していこうと考えています。

ブラウン　我々としても、SECや日本企業の皆さんにオープン・グループのメンバになっていただき、オープン・グループの活動に積極的にかかわっていただきたいと考えています。オープン・グループの活動に参加することによって学べることはたくさんあります。また、TOGAFは日本企業にとって必ず役に立つと確信しています。ぜひ、TOGAFのドキュメントに目を通していただければと思います。

文：小林秀雄　写真：越昭三郎

SEC2006年度活動概要

SEC副所長
牧内　勝哉

SECが2004年10月に発足してから2年半が経過した。組織の草創期、SECではミッション設定や役割分担、産業界・学界との関係構築などに力を割く必要があったが、2007年度に入った現在、SECの認知度も上がり、理解者・協力者も増え、活動への期待も高まってきていると認識している。ここでは、2006年度のSEC活動概要を報告する。

1　2006年度の活動領域

SECでは、「使えるソフトウェアエンジニアリング」の提案を目指し、経済産業省のエンタプライズ系ソフトウェア開発力強化推進タスクフォース、組込みソフトウェア開発力強化推進タスクフォース、先進ソフトウェア開発タスクフォース、及び、共同研究グループと共に研究に取り組んだ（図1）。各テーマの取り組み状況については、本特集をご覧頂きたい。なお、先進ソフトウェア開発タスクフォースは2007年3月に活動を終えたばかりであり、最終報告という形で次号に掲載させて頂くこととする。

従前から取り組みを継続しているテーマ（見積り手法、定量データ分析、見える化、コーディング作法、ETSS等）については、成果作成期から成果普及期へ移行する過渡期にあり、実証実験という形での導入支援等を行った。

また、2006年度に新たに取り組んだテーマには、機能安全、プロセス改善、共通フレーム、信頼性ベンチマーク等があった。

2　普及活動

2006年は、研究の結果をまとめた方法論をソフトウェア開発の現場に紹介、浸透を図る活動として、書籍の発行、SEC-Web

エンタプライズ系プロジェクト	組込み系プロジェクト	共同研究グループ
見積り手法 定量データ分析 開発プロセス共有化 要求・設計開発技術研究 プロジェクト見える化 プロセス改善研究	**組込みソフトウェアエンジニアリング領域** 設計品質技術 実装品質技術 プロジェクトマネジメント技術 開発プロセス技術 利用品質技術 機能安全 テスト技術 開発環境準備	**エンタプライズ系** 奈良先端科学技術大学院大学　松本研究室 大阪大学大学院　菊野研究室 東海大学大学院　古山研究室 愛媛大学大学院　阿萬裕久講師 鳥取環境大学　天嵜聡介助教 大阪大学大学院　楠本研究室 ドイツフラウンホーファIESE
先進ソフトウェア開発プロジェクト		
プローブ情報プラットフォーム開発	**組込みソフトウェア開発スキル領域** スキル基準 キャリア開発 教育	**組込み系** 東京大学COEものづくり経営研究センター 北陸先端科学技術大学院大学　片山研究室 国立情報学研究所　本位田研究室 宮崎大学　片山研究室 名古屋大学（NEXCESS） 早稲田大学理工学術院　中島研究室 慶應義塾大学　天野研究室

図1　SECの活動領域

サイトの運営、セミナーの主催、展示会への参加、講演、新しい方法論の導入支援活動、地域支援活動等を実施した。

【2006年度発行の書籍類】

2006年度は11種のSEC BOOKSと4種の『SEC journal』を発行した。

- 『ソフトウェア開発見積りガイドブック～ITユーザとベンダにおける定量的見積りの実現～』
- 『経営者が参画する要求品質の確保～超上流から攻めるIT化の勘どころ～第2版』
- 『組込みソフトウェア開発向け コーディング作法ガイド［C言語版］』（ESCR）
- 『組込みソフトウェア開発における 品質向上の勧め［ユーザビリティ編］』
- 『組込みスキル標準ETSS概説書［2006年度版］』
- 『ITプロジェクトの「見える化」～下流工程編～』
- 『ソフトウェア開発 データ白書2006～IT企業1400プロジェクトの定量データで示す開発の実態～』
- 『組込みソフトウェア開発における 品質向上の勧め［設計モデリング編］』
- 『組込みシステムの安全性向上の勧め（機能安全編）』
- 『組込みソフトウェア向け開発プロセスガイド』（ESPR）
- 『組込みソフトウェア向け プロジェクトマネジメントガイド［計画書編］』（ESMR）
- 『SEC journal』第6号～第9号

【SEC-Webサイトの充実】

SEC-Webサイトの1ヶ月当たりのアクセス数は平均約120,000件、利用登録者数も約9,700名に増加した。発行書籍のPDFや講演資料等のダウンロード数は合計約259,000件を超え、「SECメールマガジン」受信者数は約7,000名、DM「SECからのお知らせ」受信者数は約4,500名と、こちらも大幅に増加した。

【展示会・セミナーの主催、及び参加】

400名規模のイベント「SEC Forum 2006」（SEC成果発表会、図2)、「SEC コンファレンス2006」（SEC journal論文発表会）の主催を始め、50～100人規模の「SEC主催セミナー」を10回開催し、のべ約800名の方々にお越し頂いた。東京以外の地域（長野、大阪）でのSEC主催セミナー開催も、2006年の新たな取り組みであった。「ET West」では産業実態調査の速報が注目され、講演当日と翌日にはSEC-Webサイトへのアクセスと利用者の新規登録が集中した。「ESEC」と「SODEC」はブースに集客し過ぎてしまいアンケート用紙の配布を制限するほどの盛況であった。「ET2006」ではツール諸元表の実証実験としてスタンプラリーを行い、400名もの方にご協力頂くことができた。

図2　SEC Forum 2006における鶴保所長の講演（2006年6月12日）

図3　SIPAとの組込み技術者育成合意文書調印（2007年1月18日）

3　海外機関との連携

SECは、独国のフラウンホーファ協会実験的ソフトウェア・エンジニアリング研究所（IESE）との共同研究を引き続き行い、成果を国際学会で発表したり、ワークショップを共催するなどして関係を深めている。また、米国カーネギーメロン大学ソフトウェア・エンジニアリング研究所（SEI）のCMMI V1.2の策定作業には委員として参加し、韓国の韓国ソフトウェア振興院（KIPA）との交流も続いている。2006年度は、新たにタイ王国のソフトウェア推進機構（SIPA）との間にソフトウェアエンジニアリング分野と人材育成に関する協力協定を締結した（図3)。

4　まとめ

SECは、草創期を終え、活動期・浸透期ともいうことができる時期に入りつつある。産業、社会、生活の情報システムへの依存度もますます高まり、安全・安心、信頼性への要請も強くなってきている。SECは、こうした「動く標的」を的確に捉え、検討テーマや取り組みのアプローチを変貌させながら、問題を解決していく。

エンタプライズ系活動概要

SECエンタプライズ系プロジェクト　リーダ
長岡　満夫

SECは2004年10月に設立以来、品質と生産性向上を目的に、開発から保守・運用のプロセスでの定性的、定量的な視点からエンジニアリング要素の定着を目指し、安全・安心な社会インフラとソフトウェア産業の競争力強化に対する期待に応えるべく活動を行っている。

1　背景と目的

エンタプライズ系ソフトウェア開発力強化のためには、最適な開発の手法、開発体制、マネージメント手法を通じて人間の過ちを未然に防ぎ、甚大な影響を避けることができる技術、手法を各利害関係者が理解し、役割を分担し、技術と手法を活用して定性的かつ定量的にプロセス上でトレースできるマネージメントが必須である。

2　SECでの活動と活動サイクル

エンタプライズ系ソフトウェアではV字型開発プロセスでの上流系（企画設計、要件定義、設計のプロセス）の精度・品質が後工程の品質、コスト、納期に与える影響が大きいことから、SECは上流系での利害関係者の理解、役割分担を促進させ、下流系ではマネージメント力強化に注力することとしている。産学官の連携によりソフトウェア開発での基本的なテーマに実践的な手法で取り組み、調査・研究期（ノウハウの収集、分析、仮説と検証）、成果作成期（冊子出版、論文・寄稿、ガイドラインを成果とした啓発・啓蒙）、普及・実証期（体系的な成果の普及・定着、効果の検証）の活動サイクルで進めている。また、現在SECの各種成果（テーマごとの出版、定量データ白書、信頼性指標、ソフトウェア・ライフサイクル・プロセスの共通フレーム2007等）の各企業への本格的な適用・実施段階に入っている。個別テーマごとの具体的な内容については、それぞれの稿でご確認頂きたい。ノウハウの普及・定着を加速するために、実証実験、海外研究機関や大学との共同研究を実施し、開発ツール群、SEC-Webサイトの整備も進めてきており、2007年度からは試行適用を経て本格適用を目指している。具体的には、先進ソフトウェア開発プロジェクト（経済産業省）とEASE（Empirical Approach to Software Engineering）プロジェクト（文部科学省）の連携によるプロジェクト見える化ツール（EPM：Empirical Project Monitor）、2005年以降蓄積・分析してきた生産性、品質、工期等のベンチマーク可能な定量データに基づくプロジェクト診断システムである。

3　まとめ

2007年度はこれまでの成果の普及・実証期に移行し社会インフラの中に具体的に定着し、併行して次期3ヶ年活動計画の策定を行うフェーズに進んでいく。

普及・実証期に関する評価の視点は、①成果利用者、エンドユーザの視座に立ち、利用者がその恩恵を実感でき、わかりやすい論点や指標を抽出し、その「見える化」の実証を行う、②ITと非IT施策との一体的な施策の定着、評価のため、信頼性ガイドライン、取引・契約制度慣習等の施策も一体的に取り組む、③安全・安心な社会インフラの提供と産業競争力強化に向けた定量的な指標とベンチマークでの利用者プロジェクトの「相対評価」により、利用者側が取り組み状況に応じて分類し質・量・コストの観点で評価可能とする、ことである。

また、次期３ヶ年計画においては、安全・安心を支え絶えず変革し続ける社会インフラの実現に向け、高い信頼性を確保し、より一層の生産性の向上を目指して、開発プロセスの一貫性、開発モデルの多様性、企業競争力強化に向けた人材力と組織力に焦点を当て活動を推進していく。

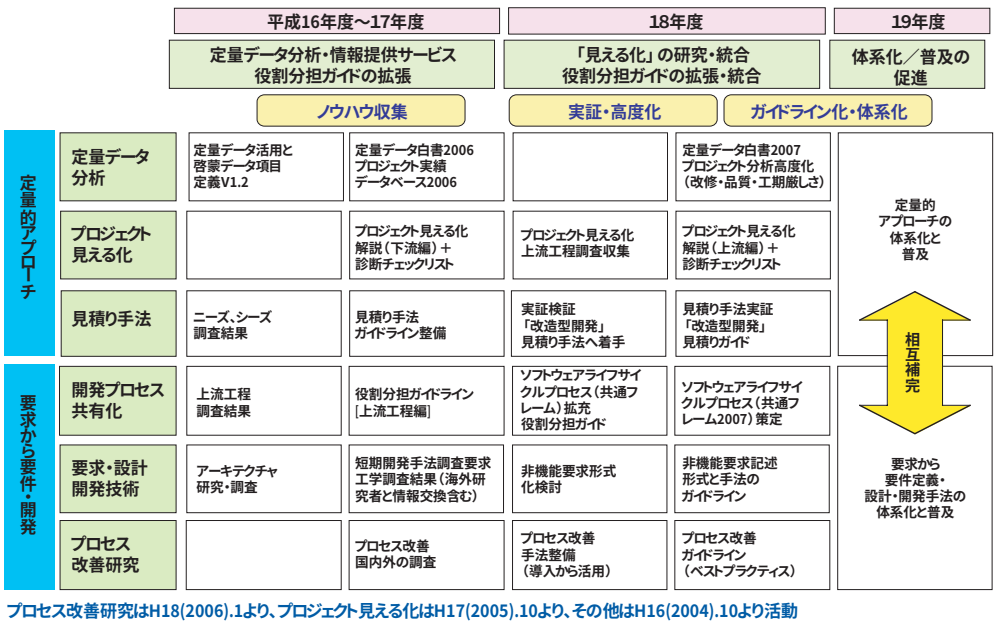


図１　SECでの取り組みと今後の課題について（３年間の歩み）

組込み系活動概要

SEC組込みプロジェクト　リーダ
門田　浩

SECは2004年10月に設立され、今年の9月末で満3年を迎える。一般に組織が新設されると、活動の年限を決めて成果を評価し、その後を決めることになる。まさにSECはそのタイミングを迎えている。

SECは、組込みに限らずソフトウェアが日本経済の大きな推進力の1つであるという認識の下、開発力を強化推進することを目的に設立された。我々が取り組む組込みソフトウェア産業は、設計された対象を機械化された工場で生産するという、従来の2次産業形態とは異なり、開発そのものが製品の製造工程のかなりを部分包含するというきわめて労働集約的な側面を持っている。

組込み系では、ソフトウェア開発のQCDを予測・改善することを目的とする工学的手法の導入と、もう1つの柱である人材面において評価・育成、ひいては調達、技術戦略にまでに影響を及ぼすスキルの可視化に取り組んだ。本稿では、3ヶ年のうち、特にこの1年を振り返る。

1　エンジニアリング領域

エンジニアリング領域では、3年間で成果を上げるために、開発工程中で、もっとも現場が苦しんでいる実装工程に焦点を当てて活動してきた。設計、コーディング、テスト、そしてそれぞれ関連するレビューあるいはインスペクション等のあるべき姿を、品質とは何かの原点に戻り再構築を試みた。もともとソフトウェアエンジニアリングには30年以上の歴史があり、我々が新たに何かを生み出すという性格の活動ではない。再構築には、現場で何が起きているか、どのような問題があり、実際にはどう対処しているか等の実状をフィードバックして戴きながらそれを整理するという手法を採った。

最初の成果はC言語を対象とした「コーディング作法」として世に問うことができた。これは規則のデータベースという性格以上に、品質とは何か、それに対応する実装とはどうあるべきかを学ぶ格好の教科書ともなり、C言語ツールベンダ数社でこの作法に基づくチェック機能を持つコンパイラも提供された。本成果の影響はこれだけに留まらず、今後、ISO/IEC 61508に代表される各種の国際標準規約の議論に、日本でのプラクティスとして活用することが期待されている。

次に挙げるべき大きな成果は2006年度末から2007年度の「開発プロセス標準」である。これは組込みに限らず、ソフトウェアを開発するに際し、何をするべきかを、標準的なV字プロセスモデルを用いて整理したもので、何をすべきかの項目に組込みとして必要な事項を付加した形式をとっている。本標準は開発現場で暗黙のうちに行われていたプロセスあるいはタスクを形式知に置き換えるための枠組みであり、道具である。現実の開発、あるいは開発におけるあるべき姿などを想定し、インスタンスを構築するのはユーザの仕事である。ここにも「銀の弾丸」は存在しないのである。

関連して2006年度の成果として、レビューガイドやテスト完了基準なども、まだ完成度は万全ではないが提供し、次期3ヵ年において早期にパッケージ化する予定である。

2　スキル標準

スキル標準活動は、ものづくりの観点からは、まったくの白紙からのスタートであったが、2005年度にはいち早く「組込みスキル標準ETSS」を公開した。これはもっぱら創始者であり提唱者である東海大学大原茂之教授の御尽力、及びタスクフォースメンバの努力の賜であり、ここに感謝の意を表したい。

ETSSは2006年度でほぼ完成の域に近づいており、ETSSを構成する3基準それぞれで活用が開始されている。スキル基準によるスキル診断は日本を代表する大手企業の採用、実証実験、キャリア基準を参照し社内評価制度を改版した企業等が相次いでいる。さらに教育研修基準に関しては教育だけでなく、組込みシステム技術協会（JASA）による技術者の評価を行うための試験制度ETECも開発され実用化に入った。

さらに大きなインパクトはこのETSSが海外にも高く評価されていることである。OMGにおいては国際標準に向けての共同作業が進んでおり、IPAとも友好関係を結んでいるタイでは政府機関SIPAでの導入が決まっている。

3　まとめ

以上、簡単ではあるが活動をレビューした。エンジニアリング領域では銀の弾丸無き世界へ楔をまずは打ち込んだところ、ETSSでは産業構造の変革に対応した人材育成への方法論を提供したばかりであり、先送りではなく解決すべき問題は未だ多岐にわたり、根が深いものと認識している。

次期3年の活動は、エンジニアリング領域ではものづくりでの手法整理と実証実験、上流工程への取り組みがメインとなる。他方スキル標準では普及活動を中心に、SECだけではなく国際化を含め、外部組織との連携でETSSの輪を広げることに注力する。

なお、経産省が推進し、自動車関係業界団体JasParが実行する「産学連携ソフトウェア工学実践事業（高信頼組込みソフトウェア開発）」プロジェクトではETSSによるスキル診断、開発プロセス標準、コーディング作法など各種のSEC成果を導入することが決まっている。

SEC 2006年度 活動概要 エンタプライズ系

「共通フレーム2007」の概要

SEC研究員
室谷　隆

昨今のコンピュータの活用方法は、初期の社内の業務効率化を目的としたものから大きく飛躍し、新しいビジネスの仕組みを構築、実現するものとなった。その結果、多くのステークホルダ（利害関係者）が登場してきた。システムを構築する際、この多くのステークホルダが関わり合い、作業していくことになるが、同じ作業を違う言葉で話していては、効率よくことが進まない。共通フレームは、ソフトウェア、システム、及びサービスに関係する人が“同じ言葉を話す”ことができるよう、共通の枠組みを提供するものである。今まで、「共通フレーム98」が存在したが、今般、この「共通フレーム98」の強化を行い、改訂版として「共通フレーム2007」を発行する。

1 「共通フレーム2007」とは

「共通フレーム2007」は、ソフトウェアライフサイクルプロセスISO/IEC 12207（JIS X 0160）をベースとし、更に日本独自の強化、拡張を行ったITシステム開発の作業規定である。

ソフトウェアライフサイクルプロセス（SLCP）は、ソフトウェアの構想から廃棄に至るまでの、開発・運用・保守に必要なプロセスを包括的に規定し、またプロセスの管理と改善についても規定している。さらに工程、時間、開発モデル、技法／ツールに無依存となっている。

2 これまでの経緯と課題

1994年に「共通フレーム94」が発行され、98年にはISO/IEC 12207（JIS X 0160）をベースに大幅な改訂を行った「共通フレーム98」が発行された。「共通フレーム98」は国際規格にはない、開発に入るまでの作業を企画プロセスとして設け、また発注側の業務担当者が行う作業を業務詳細設計として定義した。システム開発は、ビジネスを構築する観点からシステムを構築する、つまり、発注側と受注側双方の共同作業であることを明確に定義した画期的なものであった。しかしながら「共通フレーム98」が刊行されてから今日までに、国際規格ISO/IEC 12207に追補が発行され、オープン化、ネットワークの普及、Webシステム等ITシステムの利活用にも大きな変化があった。また経済産業省の「情報システムの信頼性向上に関するガイドライン（2006年6月）」、及び「情報システム・モデル取引・契約書（2007年4月）」の参照を受け、復刊、改訂が待ち望まれていた。

3 「共通フレーム2007」での改訂、強化点

「共通フレーム2007」では、「共通フレーム98」に比べ次のような改訂、強化を行っている。

(1) 規格レベルで強化した点

ISO/IEC 12207:1995は2002年に追補1、及び2004年に追補2が発行されている。ポイントはプロセス評価／改善の国際規格であるISO/IEC 15504の要求事項への対応である。「共通フレーム2007」はこの追補1及び追補2を産業界の課題解決と他のプロセスとの整合性の観点から一部取り込んだ。

①プロセスの追加

新規に以下のプロセスを追加した。

・使用性プロセス
・資産管理プロセス
・再利用プログラム管理プロセス
・ドメイン技術プロセス

②目的及び成果の追加

ISO/IEC 15504-2プロセス評価の要求事項に従った、プロセス参照モデルを確立するための結論（目的と成果）を追加した。

これにより、「共通フレーム2007」を参照した「評価モデル」が、ISO/IEC 15504プロセス評価のフレームワークとして使用できる。

③プロセスの変更及びアクティビティの追加

・人的資源プロセス（教育訓練を人的資源に改称し、内

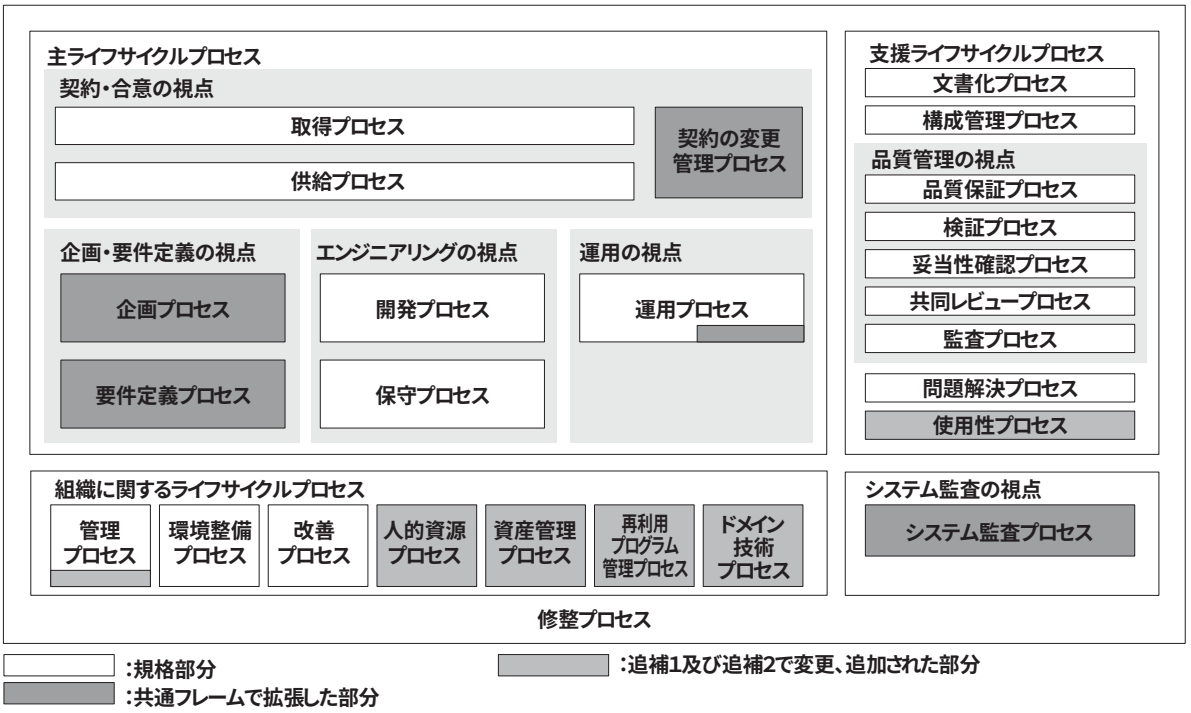


図１ 「共通フレーム2007」の全体構成

容を改訂）

・管理プロセスに測定アクティビティを追加

(2) 日本で独自に改訂及び拡張、強化した点

日本のソフトウェア産業界で必要とされるプロセスや作業項目を改訂し、拡張、強化した。主なポイントは業務の視点から、発注者が行うべき作業項目を改訂している。この業務の視点の作業は、発注側の企画者として新しい業務をどのように企画するか、企画された業務をどのように要件定義し、設計、運用していくか、運用テストや利用者教育はいつどのように計画すればよいか等である。このような業務の視点の作業は、発注者として必要な項目であるが、ソフトウェア開発を委託発注していると見落とされがちなものである。「共通フレーム98」にて追加されたこの発注者の業務視点の重要さを考慮し、作業項目の充実を図り、改訂、追加した。

①企画プロセスの改訂

企画者の視点から内容を見直し、タスク、アクティビティの再配置を実施。

②要件定義プロセスの追加

開発に入る前の要件定義の重要性に鑑み、要件定義を利害関係者の観点から整理し、必要なタスク、アクティビティを定義。

③契約の変更管理プロセスの追加

契約内容に影響を与える変更要求が生じた場合のタスク、アクティビティを定義。

④システム監査プロセスの改訂

2004年、「システム監査基準」、「システム管理基準」が改定、策定されたため、この内容に従って改訂。

⑤業務詳細設計アクティビティの分散化

「共通フレーム98」の業務詳細設計アクティビティの内容を、業務の視点から実施すべき適正なプロセスへの分散化（主に、要件定義プロセスと運用プロセスへ）。

⑥保守プロセスの強化

1999年にソフトウェア保守ISO/IEC 14764（JIS X 0161）が制定されたため、この規格の一部取り込み。

⑦タスク名称の付与（「共通フレーム98」での追加分）

ISO/IEC 12207では、タスク名がなく、すぐ文章で始まっている。タスクは項番で識別しているが、わかりやすくするためタスク名称を付加。

4 今後の展開

「共通フレーム2007」はそのベースをソフトウェアライフサイクルプロセスISO/IEC 12207（JIS X 0160）としているが、大規模システム開発で使用するシステムライフサイクルプロセスISO/IEC 15288（JIS X 0170）も存在している。ISO/IEC/JTC1/SC7/WG7ではこの2つのライフサイクルプロセスの整合化作業を実施中であり、双方の改訂版が2008年にも発行される予定である。

次期版には、ソフトウェアライフサイクルプロセスだけでなく、システムライフサイクルプロセスをも考慮した共通フレームの構築も視野に入れている。

SEC 2006年度 活動概要 エンタプライズ系

要求工学、設計・開発技術 活動概要

SEC研究員
吉田　尚志

情報システムの高度化に伴い、ソフトウェアの開発には「早く・安く・バグがない」という要素だけでなく、提供後のシステムを用いた高品質なサービスやシステム利用時の情報セキュリティなど広範な要求に対応し、実現していくことが望まれている。要求・設計開発技術研究部会では、昨年度までの自らの調査研究内容と上記の状況を踏まえた上で、設計以降のシステム開発へ多大な影響を与え、システムの稼動後においてもトラブルの原因になりやすいにも関わらず取り扱いが難しいとされてきた非機能要求に関して“非機能要求とアーキテクチャWG（以降WG）”を2006年4月に発足し、特に情報システムの設計に必要な非機能要求の記述方法に関する議論を始めた。2006年度は検討途中ではあるが、議論の中でわかってきた非機能要求の記述やその考え方について報告する。

1 非機能要求の性質

非機能要求とは、特定の機能として直接考慮されるわけではない要求を指し、実現したい機能に加えて望まれる特性や制限を意味して使われることが多い。SWEBOKでは、「非機能要求は、時として、制約条件（constraints）もしくは品質要求（quality requirements）として知られている。また、この要求はさらに以下のように分類できる。（例えば）性能要求、保守容易性要求、安全性要求、信頼性要求、電磁気互換性要求、その他の様々なタイプの要求である。」と記されており、ソフトウェア品質特性のモデルであるISO/IEC 9126はソフトウェアの重要な非機能要求を示している。

非機能要求は、なぜ扱いが難しいのか? それは、非機能要求が下記の性質を持つからである。

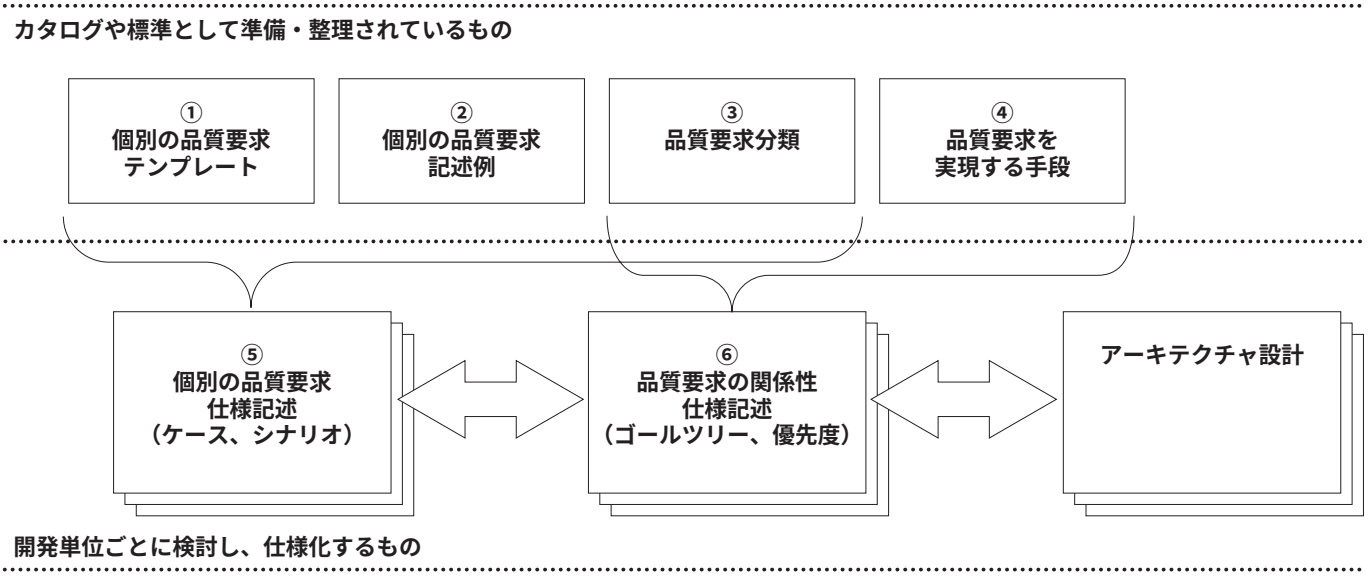
・主観性
・相対性
・相互干渉性

第一に、非機能要求は主観的に解釈され、評価される側面を持っている。様々な利害関係者がいる現在の情報システムではそれぞれの関係者の要求を把握することが望まれるが、一人ひとりの要求は同じ「性能」「保守」という言葉で表されたとしても、実際の要求は各々意味も、求めるレベルも異なることが普通である。次に、どれが重要な非機能要求か、どの程度の実現レベルが要求されるかといったことは、対象システムの特性や利用局面等に左右される。つまり、「信頼性」という言葉ひとつをとってみても、システムが異なれば求められる実現レベルは違うということである。例えば、一度問題があれば人命や社会全体へ影響を及ぼすシステムと、問題があった場合でも限られた利用者に少し不便なだけというシステムでは「信頼性」として求められるものは異なる。最後に、非機能要求には相互に干渉するという側面がある。例えば、性能を高めるチューニングは機能追加・修正等のための保守性を下げ、信頼性を高めるための冗長化や余裕はシステム資源の利用効率性を下げるかもしれない。

2 非機能要求記述の体系化への取り組み

2006年度は本WGにおいては非機能要求を「記述」という側面を中心に議論してきた。特に、品質特性とそれを中心とした要求からアーキテクチャ設計の決定に至るまでのプロセスの中で、どのような要求が扱われ記述されるかということを議論し、現状のプラクティスや利用可能な手法などを取り上げた。

記述を取り扱う意義は、先に述べたような性質を持ち、難易度の高い開発作業を、俗人的であいまいなまま行うのではなく、記述に示される有形な考え方やそれを用いる手段を用いて行えるようにすることである。ゆえに、WGでの目標は完璧な記述方法を作ることではなく、有用な記述とその考え方を示すことにより、現状では難しいと考えられている非機能要求をより体系的に扱うことができるようにすることである。図1は、2006年度に本WGで取り上げた記述を整理したものである。



図中の要素	WGで議論されたトピックを例示
①個別の品質要求テンプレート	Planguageパラメータ、SEIの6部位シナリオ記述
②個別の品質要求記述例	ISO/IEC 15408の仕様記述、SEI汎用シナリオ
③品質要求分類	ISO/IEC 9126-1、RADC、FURPS+、ほか多数
④品質要求を実現する手段	NFRフレームワークの手段カタログ、SEIのTactics/ABAS
⑤個別の品質要求使用記述	①や②をベースに記述される個々のケース、シナリオ
⑥品質要求の関係性仕様記述	NFRフレームワーク、SEIユーティリティツリー

図1　品質要求の各種記述

上記のように、体系的な考え方や記述方法を検討するとともに、成果物を現実の開発に応用できるように、WGでは各委員企業で実施されている方法や考え方を意見として取り入れるだけでなく、現実に開発された2つのシステムの要求仕様や基本設計書を入手し、それらをもとに非機能要求の記述実験を行うという実証的な方法を取り入れ活動してきた。

3 今後の展開

2006年度は、考え方や現状の整理を中心にしながら、実証を交えた検討と議論を進めてきた。2007年度は、これまで議論されてきたことの詳細化や整理を進め、具体的には、検討結果を反映した記述ガイドを作成し、その検証のためにより大規模な実証にも取り組む予定である。

また、信頼性、安全性といった情報システムが社会から求められている課題に対しても、SECでは要求・設計開発技術の活動として継続してそれらの課題解決に向けて有用な技術や手法を取り上げ、検討していきたいと考えている。

SEC 2006年度 活動概要 エンタプライズ系

見積り手法（改造型開発）

SECエンタプライズ系プロジェクト
サブリーダー
石谷　靖

見積り手法部会とSECでは、2006年4月に「ソフトウェア開発見積りガイドブック」を発行し、主に新規開発を念頭において、ソフトウェア開発の見積りでの基本的な考え方、具体的な方法及びノウハウを紹介した。残された課題として既存システムに機能の変更や新規機能の追加を行う開発の見積り方法を挙げた。2006年度の活動として、その開発を「改造型開発」*1と呼び、参加企業における実際の見積り方法（成功事例や失敗事例）を俎上にして、改造型開発の見積りでのベストプラクティスと留意点を議論した。なお、検討結果は2007年6月に「改造型開発見積りガイドブック（仮称）」として発行する予定である。

1 改造型開発見積り

改造型開発の特徴は、プロジェクトの前提としてすでに稼動しているシステムがあることである。開発は大きく3つのフェーズに分けられる。

それぞれにおける見積りの特徴は以下のとおりである。

(1) 調査分析

既存システムを調査して理解し、新たな要求を機能として実現するための影響分析を行う。影響分析を行う前には、既存システムを理解しておくことが必要である。調査分析に要するコストに影響を及ぼす要因には、既存システムの規模、理解容易性、保守性の高さ、既存部分への習熟度、変更箇所の分散度がある。

(2) 機能実現

新機能追加開発と機能変更の段階である。開発規模の見積りにおいて、既存の機能の削除及び変更を考慮に入れることが一般的であり変動要因には既存システムに起因するものがある。

(3) テスト

追加・変更した機能のテスト及び既存システムに対するテストであり、既存機能との関連部分に留意する必要がある。テストは機能の変更や追加等を契機に行うが、新たに追加した機能や変更した機能が既存システムの各所の機能と関係する度合いが高い場合は、テスト工数が増加する。新規開発との大きな違いは、機能量とテスト量の相関が新規開発に比べて低いことである。端的には、たったソースコード1行の変更でも、そのコードが含まれる機能が既存システムの多くの部分に関係すれば、その変更の妥当性を保証するために、関係する既存機能すべての再テストを実施することもある。図1にテスト工数に対する変動要因の例を示す。

コストは上記の3つのフェーズに対応して3つの要素から構成されるが、プロジェクトの特徴により全体に占める構成比のパターンは大きく異なる。図2に、コスト構造の特徴的なパターンを模式的に示した。パターンによって機能実現の量と、調査分析やテストにかかるコストとの間の関係が異なる。例えば、(a)パターンは機能実現に比較して既存システムの調査分析のために時間がかかり、また関係する既存機能が多くてテストにも時間がかかる場合である。一方、(b)パターンは、すでに経験・知識のあるシステムで、あまり既存部分の調査分析に時間をかける必要がない場合であり、(c)パターンは、調査分析にも機能実現にもあまり時間をかけずに済むが、影響範囲が大きくテストに時間がかかる場合である。

ここで重要なのは、様々なパターンがあるので、見積もろうとするプロジェクトがどのパターンに該当するのかをきちんと把握することである。条件に応じてどのようなパターンがあるかを把握しておけば、特定の組織である程度同じ開発条件（対象システム、顧客など）であ

＊1 「ソフトウェア開発データ白書2006」で定義する「開発プロジェクトの種別」のうち改修・保守と拡張に相当する。

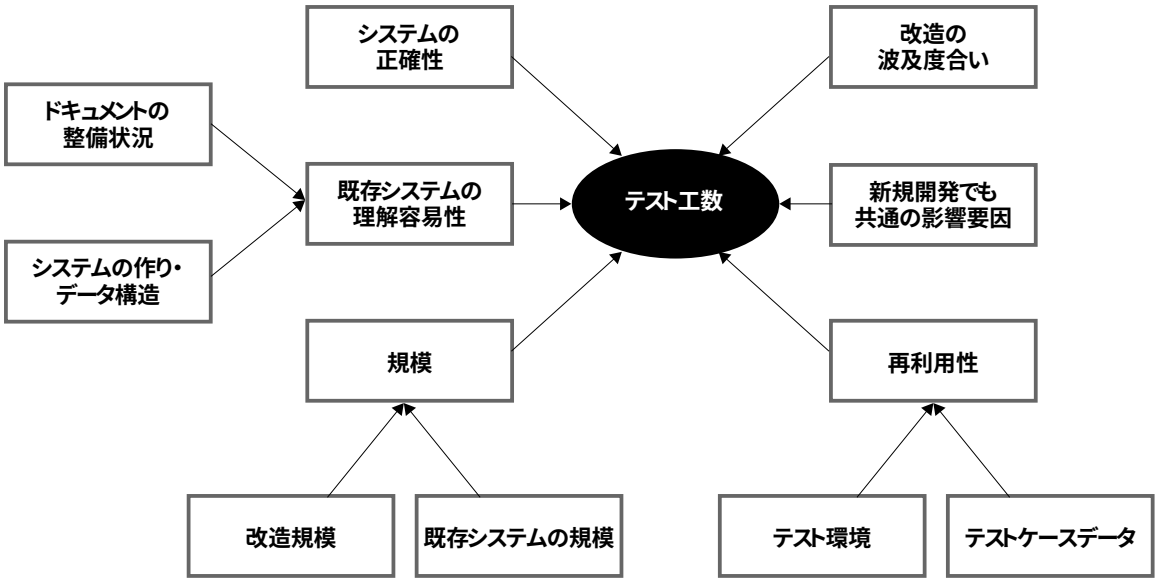


図1　テスト工数に対する変動要因（例）

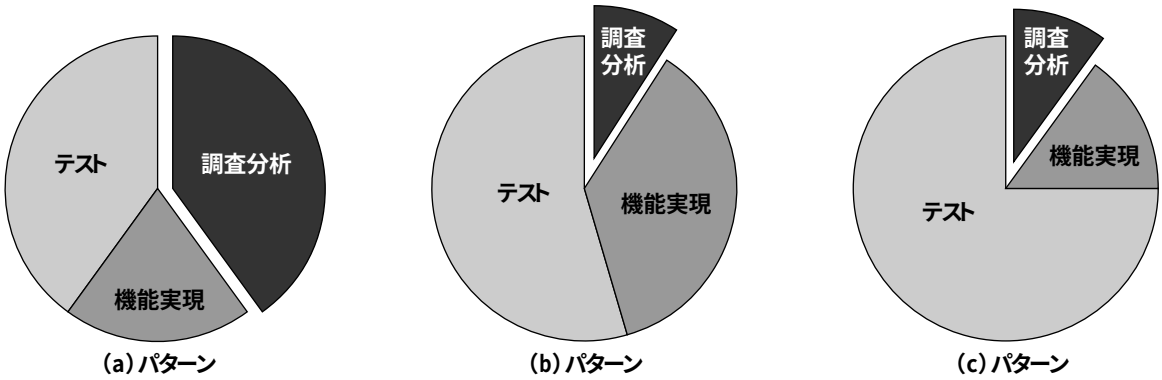


図2　改造型開発におけるコスト構造パターン例

れば、パターンに基づき予測できる。

「改造型開発見積りガイドブック（仮称）」では、ファンクションポイント法やCOCOMO法での考え方を紹介するとともに、改造型開発見積りでの成功例と失敗例に基づいて、以上に述べたフェーズごとのコストの変動要因をはじめ算出にあたっての特徴・留意事項の詳細を示している。また、改造型開発見積りに関して、3社の事例を紹介し、現場で見積りを行う際の参考にできるようにした。

2 今後の展開

2006年度発行の「ソフトウェア開発見積りガイドブック」と併せて、ソフトウェア開発に係る主要な見積りをカバーすることができたと考えている。

また、改造型開発の見積りの視点・考え方は、ERP等のパッケージの活用、モジュールなどの再利用を中心とした開発のように、何らかのシステムが既にある開発に適用可能である。そればかりではなく、改造型開発の見積り方法は、新規開発を含む開発見積り手法と位置づけることができる。

今後のテーマとして、パッケージ利用を含めたさまざまな開発形態に応じた見積り方法とともに、開発以外のシステムライフサイクルのコスト要素として、運用コストを始め、TCO（Total Cost of Ownership：総所有コスト）と捉えられるものの見積りがあり、これはユーザ企業にとってニーズが高い。また、見積りはプロジェクトの開始時に重要なだけではなく、プロジェクト全体の目標となるものである。定量的な分析や見える化等へのインプットとして、見積り値や見積り時の分析結果をプロジェクトのマネジメントで活用する方法を他の部会の成果と統合して示していきたいと考えている。

2007年度前半には、産業界の動向、課題・ニーズ等を念頭に、これらのテーマへの取り組み方を議論し、具体的なテーマを絞り込む予定である。

SEC 2006年度 活動概要 エンタプライズ系

ITプロジェクトの「見える化」上流工程編

SEC研究員
樋口　登

2006年6月に2005年度の部会活動成果として「ITプロジェクトの「見える化」～下流工程編～」を発行し、その後は上流工程を対象とした「見える化」に取り組んだ。下流工程では、プロジェクトが失敗する兆候を目に見える「現象」として捉えられることが多いが、上流工程においては、プロジェクトを失敗に導く要因は顕在化していない「リスク」として内在している。本部会では、要件定義が確定し、システムの設計に入った時点を想定して、上流工程におけるリスクを早期に認識するための手段と対応方法について検討した。

なお、2006年度の活動成果は、「ITプロジェクトの「見える化」～上流工程編～」として2007年5月に発行した。

1 ポイント

上流工程における「見える化」の方法として、3つのアプローチを検討した（図1）。すなわち、定性的見える化アプローチ、定量的見える化アプローチ、統合的アプローチである。

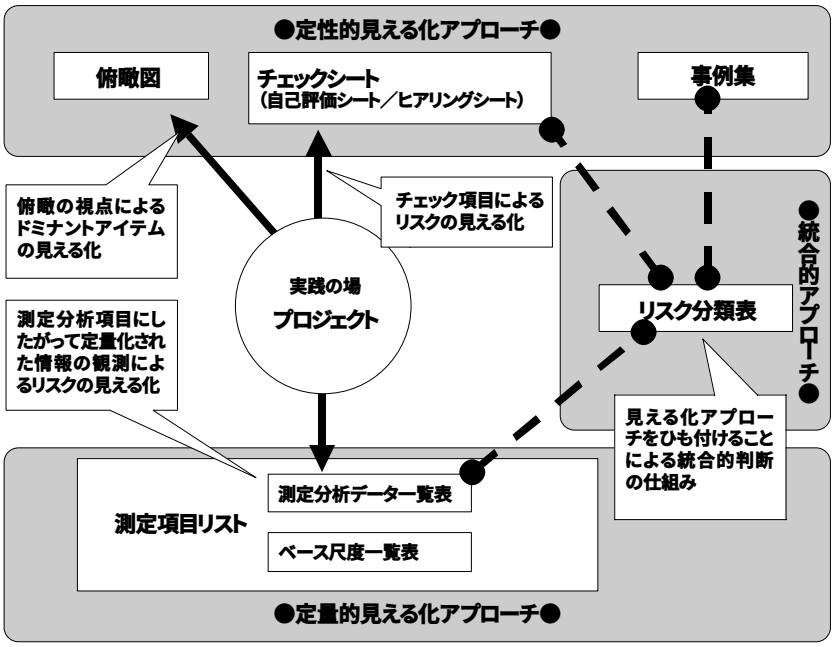


図1　上流工程における「見える化」の3つのアプローチ

定性的見える化アプローチは、プロジェクトの状態を定性的に把握するためのもので、プロジェクトの全体像を把握するための俯瞰図、プロジェクトのリスクを早期に発見するためのチェックシート、そして上流工程に起因する問題の事例集を利用する。

定量的見える化アプローチは、スコープの変化、プロジェクトの進捗状況、成果物の品質、コミュニケーションの状態などを定量的に見える化するためのもので、測定項目リストを利用する。

統合的アプローチは、定性的見える化アプローチと定量的見える化アプローチによって把握した事象からリスクを的確に洗い出すためのもので、リスク分類表を利用する。

これら3つのアプローチによって、計画との差異を把握しながらプロジェクトを進めることが重要となる。

2 各アプローチで用いる手法

上流工程における3つの見える化アプローチを実現させるための各手法について説明する。

(1) 俯瞰図

俯瞰図は、ドミナント・アイテム（プロジェクトの成否に大きな影響を及ぼす重点項目）を把握するために、プロジェクト計画書を作成する前やプロジェクト計画書と同時に作成し、問題を局所的に捉えすぎないようにするために利用する。要件や体制に変更が生じた場合には直ちに俯瞰図を修正する必要がある。

基本的な俯瞰図としては、ステークホルダ俯瞰図、プロジェクト

推進体制俯瞰図、周辺システム構成俯瞰図、システム構成俯瞰図、スケジュール俯瞰図、要員遷移俯瞰図がある。

(2) チェックシート

チェックシートには、自己評価シートとヒアリングシートがある。

自己評価シートは、プロジェクト・マネージャあるいはプロジェクト・リーダが当該プロジェクトを評価するために利用し、マネジメントの弱点を認識することができる。

ヒアリングシートは、専門家チームがプロジェクト・マネージャに対してヒアリングを行う場合に利用し、プロジェクトを客観的に評価して課題を把握し、対策を検討するための材料を提供できる。

本部会では、8つの実プロジェクトに対して自己評価シートとヒアリングシートを適用し、改良を重ねた。

(3) 事例集

上流工程における問題事例を58件収集し、それぞれについてプロジェクト・マネージャがどのような見切りを行ったのか、本来はどのように見切るべきだったのか、そのような問題が発生することを事前に捉えるためには何を把握しておくべきだったのか、及び問題への対応方法について検討しまとめた。

上流工程における問題の典型的なものは大きく5つに分類される。すなわち、プロジェクト・マネジメントの問題、要件定義などの開発範囲に関わる問題、システム設計・構築技術に関わる問題、ステークホルダに関わる問題、及びモチベーションにかかわる問題である。

(4) 測定項目リスト

プロジェクト状況を定量的に把握するための項目を検討し「測定項目リスト」としてまとめた。測定項目リストは「測定分析データ一覧表」と「ベース尺度一覧表」で構成される。

「測定分析データ一覧表」は、測定の目的、その目的を達成するための測定方法（導出尺度とベース尺度)、導出尺度の見方、導出尺度の変化からわかることをまとめたものである。なお、導出尺度はプロジェクトの状況を見るための項目で、ベース尺度は導出尺度を導くために実際に測定するデータ項目のことである。

「ベース尺度一覧表」は、ベース尺度の単位、収集する工程、時期、タイミング、収集者をまとめたものである。

定量データを測定するときには、あらかじめ測定分析データ一覧表を参照し、測定の目的などを考慮して測定する導出尺度を決め、その導出尺度を算出するためのベース尺度を求める。求めたベース尺度について、ベース尺度一覧表を用いてデータを収集する工程、時期、タイミング等を把握することができる。

(5) リスク分類表

リスク分類表は、ヒアリングシート、測定分析データ一覧表、事例集を連携させ、総合的な視点でリスクを洗い出すためのものである。

例えば、プロジェクト・マネージャへのヒアリング結果を基に、類似の過去事例を参照してリスクを検討したり、ヒアリングシートからリスク分類表を参照して測定項目を調べ、定量的な評価を行ったり、事例集から当該プロジェクトと類似するプロジェクトを見つけてチェックシートによってプロジェクトの状況を確認する、といった活用方法がある。

3 今後の展開

2005年度に下流工程、2006年度に上流工程におけるITプロジェクトの「見える化」を検討してきたが、2007年度では上流から下流までを通した見える化を検討し、今まで検討してきた「見える化」の手法やツールを実プロジェクトに適用する場合の課題を整理し、対応をまとめる予定である。

また、「見える化」手法の普及活動も推進する予定である。

謝辞

ソフトウェアエンジニアリング技術研究組合（COSE）及びチェックシートの実証にご協力いただいたプロジェクトに参加されていた方々やツールや手法等を提供していただいた関係各位のご協力に感謝いたします。

参考文献

[NAGAOKA]長岡良蔵：SEC journal No.6, p28,プロジェクト見える化とは―下流工程編―

[SEC2006] IPA SEC：ITプロジェクトの「見える化」～下流工程編～, 日経BP社, 2006

[SEC2007] IPA SEC：ITプロジェクトの「見える化」～上流工程編～, 日経BP社, 2007

SEC 2006年度 活動概要 エンタプライズ系

プロセス改善研究

SECリサーチフェロー
菊島　靖弘

来たるべきユビキタス時代に向けて、ソフトウェア規模は益々増大し複雑化している。さらにITと経営の融合、技術革新とグローバルな経済の進展に伴い、ITビジネスの質的な変化に対して抜本的な変革が望まれている。一方、ソフトウェアの開発現場を見ると、日々の作業に埋没し、改善の機会が体系的に組織に定着しているとは言いがたい。また、品質問題が企業の死命を制するに至ると共に、社会に与える影響も拡大化の傾向にある。これに対処する1つの有力な方法として、プロセス改善方法論がある。SECエンタプライズ系プロジェクトに、2006年1月、プロセス改善研究部会が発足した。

1 プロセス改善とは

プロセス改善とは、製品（プロダクト）が開発されるライフサイクルを種々の観点から分割し、制御可能な単位をプロセス（活動の集合）として認識し、各プロセスが成果物を作るために期待されていることは何で、それが今いかなる状況にあるかを診断し、診断結果に基づき改善計画を策定し、実施することをいう。

プロセス改善が目指すものは“QCD”の予測精度・制御精度を上げ、さらに改善された作業の有効性を高めることに他ならないのであるから、その目的を達成するためにそれぞれの企業が目的にかなった攻めのプロセス改善を進めることが原点である（図1）。

2 2006年度の活動内容と成果

プロセス改善研究部会では2006年度、研究部会を4つのワーキンググループ（以下WG）に分け活動を行った。それぞれのWGの活動内容及び成果は次のとおりである。

(1) ＜WG1＞モデルの作成法とモデル例の整備

プロセス改善の出発点となるプロセス診断のためのアセスメントモデルについては、それが継続的に行われ、維持され、加えて国際的にも通用するものでなければならない。WG1では、ISO/IEC 15504の要求事項を満足する、日本で開発されたSPEAK、SPINACH等を研究

◆守りのプロセス改善と攻めのプロセス改善

	守りのプロセス改善	攻めのプロセス改善
活　動	顕在化した課題、問題を解決し、良い方向に持っていこうとする問題管理型活動 損失が発生してから再発防止に専念する（させられる）	経営環境、技術動向などの時代の変化を先取し、あらかじめ問題が発生しないようにするリスク管理型活動 損失が発生しないように、あらかじめ予防保全をする
立　場	後手：問題に追われる立場 何もしなくてもやらされる羽目になり、振り回される →常に品質問題等に追われて仕事をするのではなく、逆に品質を追いかけたい!	先手問題を追いかける立場 自分たちがやっていこうと思わない限り実現しない →環境変化を敏感に捕え、先手をとって仕事のやり方を変えていく!

◆ビジネスゴールの達成(良いＱＣＤの実現）のためには

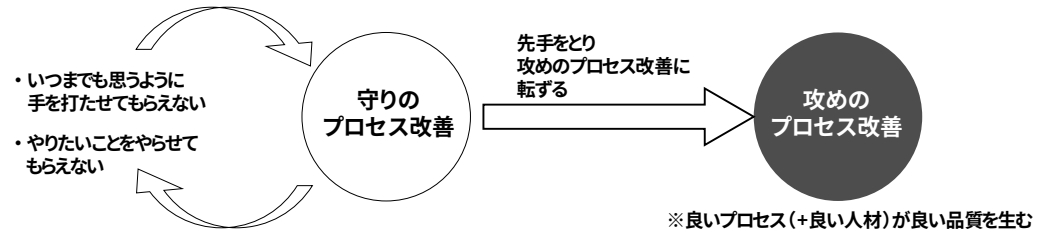


図1　守りのプロセス改善と、攻めのプロセス改善　出典：プロセス改善研究部会（足立久美委員）

し、「日本の状況を反映したプロセスアセスメントモデル」として、2007年度中に提供するべく着手した。

また、プロセス改善自体の理解が不十分な現状を踏まえ、プロセス改善を理解してもらい、やる気になってもらうための啓蒙書、加えて、啓蒙書を読んで、やる気になった人の実務要求に応えられるよう以下のガイドブックを作成・計画した。

- 「プロセス改善ナビゲーションガイド〜なぜなに編〜」（2007年4月発行）

  そもそもプロセス改善とは何を目指すのか、どのようなアプローチがあるのか、プロセス改善の基本的な概念を整理。
- 「プロセス改善ナビゲーションガイド〜プロセス診断活用編〜」（2007年4月発行）

  モデルベースのプロセス改善を推進しようとしたときに、道具として有効に利用できるプロセスアセスメントモデルの活用法について、国際規格ISO/IEC 15504の考え方をベースにまとめ、提供。
- 「プロセス改善ナビゲーションガイド〜虎の巻編〜」（2007年12月発行予定）

  プロセス改善を推進するうえでいだく実務的な疑問に答える形で、実務者向けのノウハウをガイドブックとして提供。

（2）＜WG2＞改善の具体例の収集・分析

WG2では、業界におけるプロセス改善の普及促進を図ることを目的として、プロセス改善の参考になる事例を調査し、まとめ、公開する活動を行った。

2006年度においては、7社を訪問、詳細調査を実施し、第1次発表分として以下の3社のベストプラクティス事例をSEC-Webサイト上で公開した。

- プロセス改善ベストプラクティス事例
  - ・富士通ソフトウェアテクノロジーズ
    一人ひとりを改善の主役にするためのKAIZEN塾
  - ・日立ハイテクノロジー
    設計からはじめる見逃しバグの防止
  - ・JAXA
    開発プロセスにおけるソフトウェアⅣ＆Ⅴ

さらにWG2では、現在、JISAやJUASをはじめとする各種団体等のシンポジウム・フォーラムで発表された多くのプロセス改善に係る発表、研究について分析を行っている。

これらの優れた発表を含め、プロセス改善を担うエンジニアにとって役に立つプロセス改善事例を検索、活用できる仕組みの検討を開始した。

（3）＜WG3＞プロセス改善への動機付け策の検討、啓蒙活動

WG3は、プロセス改善が正しく認識されていない現状に対して、以下の啓蒙活動を実施した。

- JISA SPES2006（2006年7月13日開催）講演
  「ITエクセレントカンパニー」
  副題「プロセス改善が実現する21世紀型のエクセレントカンパニー」
- JISA経営者セミナー（2006年9月20日開催）講演
  「マーケットニーズに応えるためのプロセス改善」
  副題「プロセス改善のアプローチ」
- JISA会報（2007年1月号　Bulletin84　寄稿）
  「プロセス改善への招待状」
  副題「ソフトウェア産業が日本を繁栄に導く」

（4）＜WG4＞国際化対応

WG4は、ソフトウェア開発の世界においてはグローバルな協業が進んでいるなかで、日本のプロセス改善の仕組みが世界で通用するものであることを検証するための調査研究を行った。さらに、日本のプロセス改善に関わる取り組みについて広く世界に発信、周知するため、海外シンポジウムでの発表、Springer社から6月に出版されるプロセス改善の研究書に寄稿した。

3 今後の展開

2007年度においては、以下について重点的に取り組む予定である。

- ıbアセスメントモデルの提供
- ıb新しいアセスメントモデルを用いたプロセス改善の実証実験
- ıbプロセス改善事例を検索、活用できる仕組みの開発
- ıbプロセス改善活動の啓蒙
- ıb国際化対応の推進

SEC 2006年度 活動概要 エンタプライズ系

定量データ分析

SEC 研究員
菊地 奈穂美

SECでプロジェクトデータの収集と分析に取り組んで3年目に入った。2005年度まで国内企業からのプロジェクトデータ収集とそのデータを基とする統計情報の「データ白書」による公開を行ってきた。2006年度からはこれを一歩進めて、次の3つの柱で活動を展開している。

①重点項目を中心とするデータ収集の継続。改修・保守や拡張（改修開発と総称）*1プロジェクトへの分析対象拡充

②プロジェクトでの定量データ活用方法の整理。定量データ分析部会で「重点テーマ」に関する議論（プロジェクトの計画での定量データ適用ガイド、品質確保の定量的取組み、工数・工期・規模のトレードオフ関係、について）

③予測や詳細分析を目的としたSECと大学の共同研究

以下では、これらの概要を紹介する。

1 データ収集とデータ白書2007

(1) データ収集の重点方針

SECでのデータ収集は、企業間でもリファレンスとして役立つようなデータ分布状況の情報発信を目指し、主要なデータ項目を重点項目と決めて収集している。2006年度のデータ収集は昨年度の重点項目を基本としたうえで、プロジェクト種別「新規開発」に加え「改修・保守及び拡張」を、業種では「公務」を加えた。他の点は昨年と同様に、言語やFP計測手法名、アーキテクチャ、業種が明確、規模・工数・工期データが揃っていること、生産性等の比較ではScopeが同じであることが必要なため基本設計～総合テストの5工程をカバーしているプロジェクト、信頼性データとしてシステム稼動後、不具合数は継続して重点項目とした。データ収集で使用した項目と定義は、昨年とほぼ同じである。

(2) 収集データ及びデータ白書2007のポイント

2006年度はデータ提供協力企業20社の協力を得て、新たに約400件弱のプロジェクトデータ収集となり、昨年までの蓄積データに加えて合計約1,770件となった。これらを精査のうえ、白書2007の編纂で用いるデータとした。収集データの特徴は、昨年や今年に重点収集項目とした項目についてはデータ記入率が比較的高めとなっている。また、SEC収集データは、稼動後不具合数データの記入件数が400件強となり、比較的多く集まったといえる。

白書2007では、新たな切り口を増やし、主として次の観点の分析グラフや統計情報を追加している。

・新規開発プロジェクトの層別分析と同等の切り口で、改修・保守や拡張（改修開発）も対象として分析
・業種で公務（官公庁他）を層別分析
・信頼性（稼動後の不具合数）分析で、層別条件を加味して新切り口を拡充し、業種別、アーキテクチャ別、言語別等、属性別に層別分析

改修開発プロジェクトでの分析過程で、生産性（工数当たりの規模）を調べた際、提出データの規模Scopeで母体規模を含む/含まないが混在していたため、部会の議論を経て、白書での改修の規模は母体を含まない規模を求めたうえで、生産性データを算出した。改修は新規と違い、規模の計測Scopeが組織やプロジェクトによって異なることが多いため、計測ガイドの整備も今後は必要である。

2 重点テーマ

定量データや指標などをプロジェクト推進において効果的に適用する方法を整理するため、部会では重点テーマを整理し次の3つのWGにおいて検討を行っている。(1)は入門的な位置づけ、(2)は少し進んだ方法、(3)は実績データから有用な知見を見出す実証である。

*1 改修開発は、見積り手法の項では「改造型開発」と呼んでいるものと同じである。

(1) プロジェクトの計画での定量データ適用ガイド

定量的な取り組み導入の入門的な位置づけとして、プロジェクト運営における定量的な取り組みの重要性を示し、プロジェクトの見積りや計画策定で特にデータをどのように活用するとよいかをまとめている。ドラフトは2007年10月に公開する予定である。

(2) 品質・信頼性

システムの信頼性確保のための定量的な品質に関する意思決定支援を目的として、有用なデータ項目と定義を明確にし、品質指標や数量的モデルによって把握できるようにすることを狙っている。具体的には、プロジェクトの進行に応じて、ソフトウェアプロダクト及びプロジェクトの品質について先を予測し、プロジェクト活動を制御するための方法を体系的に整理している。実際の企業での取り組み事例の集約及び産学での研究事例も参考にしまとめている。ドラフトは2007年10月に公開する予定である。

(3) 工数・工期・規模のトレードオフ関係

プロジェクトの特徴を表す要素間（主に工数・工期・規模など）のトレードオフ関係を明確にし、現実的なプロジェクト運営の判断の参考や成功要因を明らかにすることを目的とした。似たような規模のシステムを開発するにも、工期の長短に差がある場合では、システム開発に必要になる工数が異なることが多い。そのため、工期に着目しその厳しさの程度によって、プロジェクトの特徴がどのように違いがあるかを、白書2006のSEC収集データ[SEC2006]を使って深掘分析を行い、関連する特性を明らかにした。詳細はレポートとして本号SEC journal No.10に技術解説として収録している。

3 予測や詳細分析に関する共同研究

ソフトウェア・プロジェクト・データの特徴を踏まえた分析手法や予測技術などについては、専門性を持つ大学等研究機関と表1のように、共同研究の形で先行技術の調査及び検討を進めている。いずれも実際のデータはSEC収集データを用いている。

表1(a)は、プロジェクトの予測、見積り、実行時の制御での定量データの効果的な活用という目標に向けた検討である。

表1(b)では、SEC収集データのように複数企業混在のデータセットを参照し活用する局面で、異なる規模計測によるデータ（例えば、複数のFP手法）が混在することがある。そのような異なる計測手法の規模を同じ単位で扱うための検討を行っている。例えば、FP計測手法でIFPUG法とSPR法で計測したFP値は厳密には一致していない可能性があるためである。これまでの検討状況から、今後、合計値としてのFP以外にFP詳細データが増えることにより、FP手法間の変換（換算）の検討が加速できると考えている。

表1 関連する大学共同研究

(a) 部会先行課題	
大阪大学	プロジェクト混乱予測に関する研究
奈良先端科学技術大学院大学	欠損を含むプロジェクトデータを基にした工数予測の実証、評価
東海大学	生産性等と関係するプロジェクトの要因の分析
(b) 特定課題：異なる規模計測手法の規模データ比較法	
奈良先端科学技術大学院大学	異なるFP計測手法間におけるFP変換式の導出
愛媛大学	推定・近似に基づいた機能機の計測法間での変換法
鳥取環境大学	相互比較可能な形式へのファンクションポイント値の変換方法に関する検討

4 今後の展開

SECでは、主要なデータ要素の関係を見ることを目的とし重点項目を優先して収集してきたため、重点項目はデータの充足率も比較的高く、データ間のマクロな関係を把握できつつある。一方、収集データでは、プロファイル属性となる多くの項目（例：要員経験やスキル、他）は、多数の企業で一律の収集が容易ではないことからこれまでオプショナル項目としており、欠損率が高い項目もある。そのような属性データは、共同研究で推進しているような詳細分析での、生産性モデルの作成やプロジェクト結果の要因分析を行う際には重要となってくる。

今後は、プロジェクトでの予測、見積り、実行時の制御などでの定量データの効果的な活用を目的として、分析目的ごとに収集項目セットを整理し、実際の現場で有用なデータの関係やモデルなどを実証例と供に示し、エンジニアリングアプローチの普及を行うことが重要と考えている。

参考文献

[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006

SEC 2006年度 活動概要 組込み系

組込みソフトウェア エンジニアリング領域の状況

SEC組込みソフトウェアエンジニアリング領域
幹事
平山　雅之

ここではSEC発足以来の組込みソフトウェアエンジニアリングに対する取り組みを俯瞰し、次の3年に向けた活動の考え方等を紹介する。

1 はじめに

SECが発足してから、早いもので、すでに第1フェーズの最終段階に入っている。SEC発足前年の準備会に国内の有識者の皆様にお集まりいただき、我が国の組込みソフトウェア・組込みシステム産業が抱える課題等を集中的に議論する中から、「日本の組込みソフトウェア産業を元気にしよう！」というスローガンが生まれ、SEC最初の3年間の方針と方向性が決められた。この3年間、SECでは「日本の組込みソフトウェア産業を元気にする」ために、この準備会で決定した方針にのっとり、様々な活動を展開してきた。中でも組込みソフトウェアの開発力強化に関して、SECでは、大規模・複雑化が進行する組込みソフトウェア開発の世界において、その品質を如何に確かなものとしていくかという課題を中心に取り組んできた。SECではこの課題を解決すべく、2004年10月、組込みソフトウェアエンジニアリング領域として品質向上技術部会、開発プロセス技術部会、プロジェクトマネジメント技術部会の3部会体制からスタートし、2006年度下期には図1に示すように7部会体制（準備部会を除く）に活動領域を広げ、それぞれの切り口からの課題やそれらを解決するための手法や考え方の検討を進めてきた。本稿では、これらの経緯を踏まえ、SEC発足後の最初の3年間の組込みソフトウェアエンジニアリング領域の活動成果や現状の課題認識、次の3年に向けた方向性等について紹介してみたい。

2 第1フェーズの主要成果

組込みソフトウェアエンジニアリング領域では、組込みソフトウェアの開発力の向上、特にその品質の向上を大きな目標として部会活動を中心に検討を進めてきた。テーマを大きく分類すると、プロダクト品質の向上とプロセス品質の向上に分けることができる。

(1) プロダクト品質の向上

製品としてのソフトウェアそのものの品質を向上するために要求、設計、実装、テストといった開発のそれぞれの作業における品質向上のための手法や技法、考え方等を整理し、下記に示すものを小冊子や書籍の形で公開した。

- 「組込みソフトウェア開発向けコーディング作法ガイド［C言語版］」（略称：ESCR）
- 「組込みソフトウェア開発における品質向上の勧め（ユーザビリティ設計編／モデリング編／コーディング編）」

例えば、コーディング作法は、実際に組込みソフトウェアをコーディングする際に守るべき基本的な考え方やルールを体系的に整理したものとなっており、既にいくつかのツールベンダからESCR対応のコード解析ツール等も提供され、広く利用されるようになってきている。

(2) プロセス品質の向上

組込みソフトウェアの開発を円滑に進めるためにはその開発作業自体の質を改善していくことが求められる。組込みソフトウェアエンジニアリング領域では、組込みソフトウェアの開発プロセスとプロジェクトマネジメントに着目し、それぞれを円滑に進めるためのガイドを整備した。

- 「組込みソフトウェア向け開発プロセスガイド」（略称：ESPR）
- 「組込みソフトウェア向けプロジェクトマネジメントガイド［計画書編］」（略称：ESMR）
- 「組込みソフトウェア開発におけるプロジェクトマネジメントガイド導入の勧め」

ESPR、ESMRについても組込みソフトウェアの開発を

始める際の開発プロセス検討や開発計画書作成の参考として利用いただいている企業も出始めている。

3 2006年度の活動

2006年度については、これまでの部会活動成果の取りまとめや次の3年に向けた準備を平行して進めてきた。成果の取りまとめについては既に前章で述べたとおりESCR、ESPR、ESMRの3点を正式版として公開したが、これ以外にも次のような活動も進めてきた。

①ツール諸元表：組込みソフトウェア開発へのツールの積極的導入を促進するためのツール諸元表テンプレートを整備

②高信頼／安全性向上技術：テストの十分性を確保するための考え方の基礎検討やIEC 61508等を参考にシステムとしての機能安全を実現するための考え方を整理

③設計資産再利用技術：より高次の品質と生産性を実現するために設計資産の再利用技術に関するヒアリング調査等を実施

これらのテーマのうち、特に②③については組込みソフトウェアエンジニアリング領域として次の3年に続く助走テーマとして位置付け関連部会等での議論を進めてきた。

4 第2フェーズにおける組込みソフトウェア

(1) 現状の課題認識

第1フェーズでのSEC発足当初、組込みソフトウェア開発については、規模爆発と品質低下への対処が最優先課題として取り上げられた。この3年間のSEC組込みソフトウェアエンジニアリング領域の活動はこうした課題への対応という点において、十分とはいえないものの、多少なりとも参考としていただける成果を提供できたのではないかと考えている。しかし、改めて組込みソフトウェア開発の世界を直視すると、3年前に比べ組込みソフトウェア開発にはさらに多くの課題が山積している。特にその中でも、組込みソフトウェア開発に関して、「①諸外国の急速な追い上げ」「②国際規格を武器にした技術競争の激化」の2点は我が国の組込みソフトウェア産業が国際的に埋没する危機を招く主要因になりかねないところまできている。SECではそうした状況に陥いらないように、我が国の組込みソフトウェア産業の国際競争力を維持していくための方策を考え提供していきたいと考えている。特に諸外国の追い上げの中で我が国の組込みソフトウェアの地位を向上するためには、日本が得意とする品質や安全性といった視点やそのための技術に一層の注力をしていくのが近道であると考えている。その点からも第1フェーズで手がけた高品質を目指すアプローチを更に洗練し普及していく必要があると考えている。これはまた、同時に国際規格等を凌駕するより洗練された開発技術の整備への道筋をつけることにもつながっていくと考えられる。

(2) 第2フェーズの方向性

現在、SECでは上述の課題認識のもと、組込みソフトウェアエンジニアリングの第2フェーズに向けての方向付けを進めている。この中で、特に①第1フェーズとして提供した手法や考え方の普及加速、②安全安心を実現する組込みソフトウェア開発手法整備の2点を中心課題にすえたアプローチを検討している。特に前者については、実際に組込みソフトウェア開発に携わっている企業や技術者の皆様の積極的な参加によって成り立つテーマであり、多くの方々の参画と協力を期待している。

本稿の最後として、SEC第1フェーズの組込みソフトウェアエンジニアリング領域の部会活動に参加協力いただいた皆様、成果活用等に協力いただいた皆様に深く感謝するとともに、SEC第2フェーズの活動についても引き続きのご協力をお願いしたいと思います。

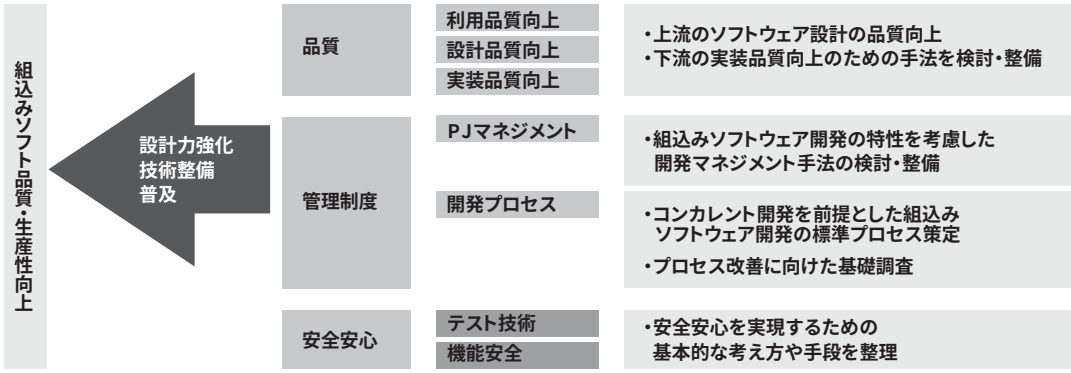


図1　エンジニアリング領域の部会体制（2006年度）

SEC 2006年度 活動概要 組込み系

組込みソフトウェア開発のプロセス面の支援

SEC組込みソフトウェアエンジニアリング領域
幹事
平山　雅之

ここでは組込みソフトウェア開発を対象とした開発プロセス整備のソリューションであるESPRについて、その狙いや概要を紹介する。

1 はじめに

近年の組込みソフトウェア開発では品質・納期・コスト等に関して目標未達となる開発プロジェクトが少なくないといわれている。こうした組込みソフトウェア開発に関する課題背景の1つには、開発プロセスの整備が遅れていることが大きく関わっている。このため、SECでは組込みソフトウェアの開発プロセスの問題を解決するための方策の1つとして、2006年秋に「組込みソフトウェア向け開発プロセスガイド」（ESPR：Embedded System development Process Reference）を策定・発行した。本稿ではこのESPRの概要を紹介するとともに、組込みソフトウェア開発における開発プロセスのあり方について紹介していきたい。

2 組込みソフトウェアの開発プロセス

(1) 開発プロセス標準化

①ソフトウェア開発プロセスとは

組込みソフトウェアを作る場合、その過程で様々な作業を順番に実施していく必要がある。ソフトウェア開発の世界での開発プロセスとは、こうした様々な作業をグルーピング化し、体系的に整理したものと考えることができる。例えば、「ソフトウェアアーキテクチャ設計（構造設計）」というプロセスの中には、ソフトウェアの動的構造に関する設計作業や静的構造に関する設計作業等が含まれる。こうした設計作業に関しては、入力として「ソフトウェアの要求仕様」が与えられ、この作業の結果として「ソフトウェアアーキテクチャ設計書」が作成されることになる。言い換えると、ソフトウェア開発プロセスとはこれらの入力に対して、ある作業を行い、ある出力を得るための一連の活動を体系的に整理したものと考えても差し支えない。

②ESPRの狙いと特徴

多人数からなるプロジェクトでは担当者によって作業の内容や認識が異なる場合が少なくなく、結果的に作業の抜け等が発生しやすくなるといわれている。これに対しESPRは組込みソフトウェアを作る上での必要不可欠な作業を整理することにより、作業者間の作業やプロセスのバラつきを押さえることを目的としており、以下のような特徴を持っている。

- 組込みソフトウェア開発に必要な作業を3階層（アクティビティ、タスク、サブタスク）のフレームで整理
- 各プロセスの結果として整理すべき各種ドキュメントの雛形をドキュメントテンプレートとして提供（テンプレートはSEC-Webサイトからダウンロード可能）
- 組込みソフトウェア開発において注意すべき事項、工夫すべき事項を掲載

③ESPRの想定利用者

ESPRは開発の現場でプロジェクトマネージャやリーダ、技術者の方々が担当するプロジェクトで実施すべき作業やプロセスを計画する際に参考にしていただくことを想定している。また開発作業の当事者以外でも個々の組織の作業標準や作業プロセスを整備する場合にも利用していただくことを想定している。ESPRはプロセスの専門家ではない技術者やマネージャの方々が読んでもわかるように、組込みソフトウェア開発の世界で一般的に利用されている用語を用いて開発プロセスを整理している点も特徴の1つである。

(2) ESPRの概略

ESPR Ver.1.0では、組込みソフトウェア開発に必要な作業のうち、主にソフトウェア開発に関する直接的な作業（SWP：Software Engineering Process）と間接的な支援業務（SUP：Support Process）の一部作業を階層的に整理している。SWPはSWP1からSWP6まで6つのアクティ

ビティに分けて整理してあり、図1（a）はこのうちの「SWP1：ソフトウェア要求定義」アクティビティに関して整理した部分である。図に見るように、アクティビティの入出力と主要な作業アイテム（タスク）、主な注意点を整理している。 また図1（b）は「SWP2：アーキテクチャ設計」アクティビティに含まれる主要作業タスクの1つである「SWP2.1.3 振る舞いの設計」タスクに関して整理した部分である。個々の作業プロセスやアクティビティで実施する作業詳細については、このようなタスクという概念で、その詳細を定義している。なお、ESPRで整理したこれらのプロセスや作業については、ソフトウェア開発プロセスに関するISO/IEC 12207他の国際規格等も参考に、組込みソフトウェア開発分野での専門家の意見を加えて整理している。

(3) 開発プロセスの評価・改善との関係

ソフトウェアの世界ではソフトウェア開発プロセスの評価／改善（SPA/SPI Software Process Assessment、Process Improvement）という考え方に関心が集まっている。その代表的な手法がCMMI（Capability Maturity Model Integration）やISO/IEC 15504の参照モデルである。これらのSPA/SPIの手法の多くはプロセス評価モデルを用いてプロセスが組織内で適切に決められ運用されているかどうかを評価し、問題点の是正措置を講じるという考え方に立脚しており、SECエンタプライズ領域では研究部会で検討している。一方、SEC組込みソフトウェアエンジニアリング領域では、開発プロセス技術部会において、ESPR策定と並行して組込みソフトウェア分野へのSPA/SPIの導入についても議論してきたが、現状以下のような課題があるとの指摘があった。

- SPA/SPIの手法としての投資対効果について、組込み分野での検証や手法評価が十分ではない
- 既存手法で提案されているプロセス評価モデルは組込みソフトウェア開発のプロセスに十分に対応しきれていない
- 極めてタイトなスケジュールで進む組込みソフトウェア開発の現場への適用については、既存手法ではあまりに時間と負荷がかかりすぎる
- 既存のSPA/SPI手法では自己改善を目的としつつも結局は組織のレベル付けに走ってしまうという課題も多く、活動が形骸化しがちである

こうした指摘を踏まえ、我が国の組込みソフトウェア分野においてはSPA/SPIの普及に重点をおくよりも、その前段階である組込みソフトウェア分野での開発プロセス整備の一環としてESPRの普及促進に力点をおいた活動を進めるほうが有効であると考えている。この点からも、より多くの皆様にESPRを手にとってご覧いただき、開発プロセス整備の参考にしていただければと考えている。

3 ESPR Ver.2に向けて

現在公開しているESPR Ver.1.0は、ソフトウェアエンジニアリングプロセス（SWP）、サポートプロセス（SUP）が整理されている。ESPRについては、2007年度中に更に上流のシステムエンジニアリングプロセス（SYP）を整備し、より充実したものにバージョンアップを予定している。既にいくつかの企業ではESPRをご活用いただいていると聞いているが、そうした企業には、実際にご利用していただいた際のご意見等もお寄せいただき、Ver.2発行時にそれらの意見を取り入れながら、よりよいものにしていきたいと考えている。

図1 ESPRの概要

SEC 2006年度 活動概要 組込み系

組込みソフトウェア開発のプロジェクトマネジメントの支援

SEC組込みソフトウェアエンジニアリング領域
幹事
平山　雅之

ここでは組込みソフトウェア開発を対象とした開発プロジェクトのマネジメントを円滑に進めるためのソリューションであるESMRについて、その狙いや概要を紹介する。

1 はじめに

近年の大規模な組込みソフトウェア開発では、多人数によるプロジェクトを組織して開発にあたる場合が多くなってきている。しかし、一方で、こうしたプロジェクトによる開発では如何にプロジェクト内の意思疎通を図り円滑に進めていくかが成否の分かれ目となる。プロジェクトマネジメントはこうしたプロジェクトを円滑に進めるための技術として近年注目を集めている。プロジェクトマネジメントの基本は計画に対して実際がどのようになっているかを常にチェックし、プロジェクトを適切な方向に導いていくことにある。このためSECでは組込みソフトウェア開発の現場にプロジェクトマネジメント技術の普及浸透を進めるため、まず、そのベースとなる実行可能な開発計画書を策定するためのガイドを「組込みソフトウェア向けプロジェクトマネジメントガイド」（ESMR：Embedded System development Management Reference）として整備し公開した。本稿ではこのESMRの概要を紹介する。

2 ESMRとは

(1) ESMRの狙いと特徴

ESMRは組込みソフトウェア開発を始めるにあたり、プロジェクトとして実行可能な全体計画を検討し作成することを狙っている。一般に開発計画書等は多くの企業やプロジェクトで作成されているが、そこにどのような情報を検討して盛り込んでいくかはまちまちであり、結果として計画の漏れや抜け等が発生することが少なくない。ESMRでは組込みソフトウェアの開発プロジェクト着手時点で検討し決めておくべきことを整理し、計画書として標準的なスタイルを利用して文書化することを狙っている。このため、ESMRは以下のような特徴を持っている。

- 品質・納期・コスト等の側面を意識したプロジェクトの全体像を整理するための枠組みを用意
- 組込みソフトウェア開発プロジェクトの計画を立てる場合に注意すべき点等も考情報として掲載
- 順次埋めていくことでプロジェクト計画書を作成できる標準的な計画書のテンプレートを用意（テンプレートはSEC-Webサイトからダウンロード可能）

(2) 想定する利用者

ESMRは実際の開発現場でプロジェクトを担当するマネージャやリーダの方々がプロジェクト計画書等を作成する際に活用していただくことを想定している。また、組織内でこうした計画書の標準形を整備する際にも活用いただけると考えている。

3 ESMRの構成と利用効果

(1) ESMRの構成

ESMRで推奨する開発計画書は全体が大きく2つに分かれている（図1）。第1部はプロジェクト計画のサマリ情報として様々なステークホルダへの説明資料として位置付けている。一方、第2部はプロジェクト計画の詳細であり、品質・納期・コストと開発のリスクといった4つの視点からそれぞれ詳細な計画を策定する枠組みを与えている。図2に示すように計画の詳細部についてはWBS（Work Breakdown Structure）等を活用し、開発規模見積りから開発工数配分まで詳細な見積りを行うための表等、図表形式を活用している。また、ESMRではこうした開発計画書の雛形と共に、それぞれの図表を利用する上での注意事項や組込みならではの留意事項、具体的に計画を立てる際に利用できる手法事例等も合わせて掲載している。

(2) ESMRの利用効果

開発プロジェクトを円滑に進める上でその開発計画書は出発点として位置付けることができる。一般的に実際の開発現場では、ある意味理想を掲げた目標としての計画書が少なくなく、現実には計画段階で無理のあるプロジェクトが多い。これに対しESMRはあくまでも「実行可能な開発計画書」の策定を目指している。ESMRでは品質、コスト、納期、リスクという4側面からプロジェクト全体を俯瞰することにより、添付の計画書テンプレートを利用して計画書を作成することで、それぞれの矛盾や不整合をチェックしやすくし、また、計画の抜け等を防ぐことができる。これらにより無理な計画を未然に防ぐ効果が期待できる。

4 既存のプロジェクトマネジメント手法との関連

(1) 見積り手法との関係

SECではエンタプライズ系ソフトウェア領域に関して見積り手法の検討を進めている。エンタプライズ系で検討している見積り手法はファンクションポイント（FP）法等ソフトウェアの価値を見積もる方法であり、主にユーザとベンダ間のソフトウェア価格決定の道具として位置付けられている。一方、ESMRでは実際の“ものづくり“に焦点を合わせ、その開発スケジュール等現場作業の見積りや計画等作業の段取りを中心としており、エンタプライズ系ソフトウェアの見積りとは狙いが異なっている。

(2) 既存手法等との関連

今回策定したESMRの開発計画書はIEEE 1058(1998)のソフトウェアの開発計画に関する規程を参考に策定している。このため、ESMRに準拠した開発計画書を策定することで国際的にも通用する計画書を作成することができる。一方でプロジェクトマネジメントに関しては、PMI（Project Management Institute）が編纂したPMBOK（Project Management Body of Knowledge）等が広く参照されており、その実践形態としてプロジェクトマネジメントの専門家集団からなるPMO（Project Management Office）といった考え方が模索されている。今回、ESMRの検討段階でもこうした考え方についても様々な議論を加えてきたが、組込みソフトウェア開発プロジェクトの場合、ソフトウェア開発やハードウェア開発等様々な要因が複雑に関係するため、それぞれの開発の特性等を熟知したプロジェクトマネージャを主体にマネジメント活動を進めたほうが良いのではないかといった意見があった。こうした点を考慮して、ESMRでは実際の開発プロジェクトの運営に直接の責任をもつプロジェクトマネージャやリーダの方々に参考にしていただき、マネジメント活動の最初の一歩である計画書作成を進めるための参考書として活用していただければと考えている。

5 ESMR Ver.2に向けて

現在公開中のESMR Ver.1.0は開発計画書の枠組みを規定したものである。実際にこうした枠組みを埋める際には、例えば、ソフトウェア規模の見積り方法等具体的な手法や考え方が必要となってくる。このため、SECでは2007年度以降、組込みソフトウェア開発の規模や工数見積りの具体的な手法整備を検討し、ESMR Ver.2としての公開の準備を進めていきたいと考えている。

プロジェクト概要情報	Chapter 1 プロジェクトの概要	
	1.1 プロジェクトの目的 1.2 プロジェクトの目標 1.3 目標達成のための方針・手段 1.4 プロジェクトの範囲	1.5 プロジェクトの前提条件 1.6 プロジェクトの成果物 1.7 スケジュールと予算 1.8 計画の更新
参照情報	Chapter 2 参照・定義	
	2.1 参照	2.2 定義
プロジェクトの体制	Chapter 3 体制	
	3.1 製品開発プロジェクトの体制 3.2 外部インタフェース	3.3 ソフトウェア開発プロジェクト内部体制 3.4 役割分担
プロジェクト詳細計画	Chapter 4 リソース計画	
	4.1 開発規模と工数の計画 4.2 人員計画 4.3 設備、機器等の調達計画	4.4 プロジェクトの人員研修計画 4.5 予算計画書
	Chapter 5 作業計画	
	5.1 開発作業の洗い出し 5.2 開発作業の順序付け	5.3 開発作業担当者の割付 5.4 作業計画
	Chapter 6 品質保証計画	
	6.1 品質目標 6.2 品質保証の体制と仕組み	6.3 品質保証に関する主要なイベント
	Chapter 7 リスクマネジメント	
	7.1 リスクマネジメントの方針と仕組み	7.2 リスク一覧表

図1 プロジェクト計画書全体目次

図2 ソフトウェア機能別の規模工数見積り表

SEC 2006年度 活動概要 組込み系

ツール諸元表の策定

SEC研究員
猪狩　秀夫

組込みソフトウェアの開発現場に、ますますの高品質と効率化を求める声が大きくなっている。SECでは、開発現場で活用できる手法・技法[※1]の提供を行ってきたが、効率のよい開発のためには、手法・技法の他に各種開発支援ツールの有効活用が必須である。そこで、各種ツールのスペック把握をしやすくする「ツール諸元表」を策定し、提供することにした。

「ツール諸元表」は、ツールの機能概要や特徴等を表形式で整理するもので、ツールユーザによるツール選定や導入を容易にすることを目的としており、将来的にはツール提供各社のパンフレット等に標準的に添付して頂くことを目指している。

1 ツール諸元表作成の目的と背景

一般ユーザが購入する家電製品や自動車等のパンフレットは、ユーザが知りたい情報が「スペック表」、「機能比較表」等にまとめられ、各社の記載項目がほぼ一致しているため、各製品の仕様を比較しやすくなっている。

ところが、組込みソフトウェアの開発現場で参考にする、開発支援ツールパンフレット類に記載されている情報はまちまちである。そのため、ユーザがツールを導入する際は、様々な手段により苦労してツールの情報を入手しているのが現状である。

そこで、SECでは、ツール選定時に必要な製品情報のポイントをまとめた「ツール諸元表」を策定することとし、経済産業省組込みソフトウェア開発力強化推進委員会開発環境準備部会にて、その項目を選定した。

2 ツール諸元表

ツール諸元表では、軸項目として15項目を選定し、それぞれ軸項目に対応する詳細項目を定義（表1）した。

詳細項目は、ツール導入時に確認が必要な項目に絞って選択を行った。

例えば、あるテストツールの諸元表を見たとき、軸項目「対応可能プロセス」の記載事項として「SWP4～SWP6」とあれば、そのツールは『組込みソフトウェア向け開発プロセスガイド（ESPR Ver 1.0）』の規定によるSWP4「実装及び単体テスト」、SPW5「ソフトウェア結合およびソフトウェア結合テスト」、SPW6「ソフトウェア結合テスト」に利用できることがわかる。また、軸項目「教育」の記載事項として「導入時教育コース（3日間）有り、無償」とあれば、導入サポートサービスを受けられることがわかり、ツール導入後の計画が立てやすくなる。

表1　ツール諸元表

軸項目	詳　細
製品名（紹介文）	ツール名
	ツールのロゴ
	社名、サイト
ツール概要	ツールの特徴、ツールコンセプト
	ツール概要の説明
対応可能プロセス	組込みソフトウェア向け開発プロセスガイドのアクティビティを記載
連携可能他社製品	データの互換性を記載
	連携可能な会社のツール名（自社も含む）
動作環境	必要メモリ
	OS
	CPU
	HDD
	通信環境
	…
対応ドメイン 規格に準拠しているかどうか等	どういうドメインに向いているのか? （例：家電系、車載系、通信系・・・） 規格にのっとっているか?　例：MISRA
教育（スキル標準とは別なもの）	教育コースがあるか?
	期間、コスト
事例	参照可能な事例 提示してあるサイト等を記載
対応手法、参考文献	ツールが対応している手法、参考文献等
	アルゴリズム（例：ROOMS、OCTOPUS）
想定する利用者	キャリア基準の職種を記載（例：プロジェクトマネージャ、ソフトウェアエンジニアなど）
前提条件 制限事項等を記載	ツールを扱うにあたって必要なスキル
	プログラム言語の習得
	ソフトウェアに関する基礎知識
	組込みに関する基礎知識
	制限事項を記載
カスタマイズ	カスタム可能かどうか?
マニュアル	オンライン対応可能
	言語対応（日本語、英語、ドイツ語など）
価格	オープン価格、定価
	ライセンス形態
サポート	技術サポート、導入サポートがあるか?
	コストがかかるか?

3 ツール諸元表の実証結果

SECでは、2006年の「ET2006（Embedded Technology 2006）」に出展し「ツール諸元表スタンプラリー（実証実験）」と題したイベントを実施した。そして、実際にツール販売を行っている出展企業13社の協力を得て、自社ツールの製品情報を記載したツール諸元表を会場で配布して頂いた（表２）。

また、SECブースにおいては、実証実験イベント参加者を対象にツール諸元表に関するアンケート調査を行い、407名から回答を得た。

アンケートの結果、ツール諸元表について「大変参考となる」、「参考とする」の合計が平均84％に達し、大変好評であった（図１）。

他にも、質問「ツール諸元表はツール選定や比較等の際に役に立つか」への回答では「大変役に立つ」、「あれば役に立つ」の合計が93％となり、組込みソフトウェア開発者のツール諸元表への期待が大きいと考えることのできる結果となった。

この結果を踏まえ、2007年3月、SEC-Webサイトにダウンロードファイルとしてツール諸元表の雛形を掲載した[※2]。

4 今後の展開

ツール諸元表のような企業の壁を超えた定型フォーマットが提供できるのは開発環境準備部会委員各位の協力の賜であり、感謝する。

ツール諸元表がツール提供企業各社に採用されれば、組込みソフトウェア開発者がツール選定時に余分な工数を割いて確認する作業が減り、また、ツール提供企業では、採用後に思わぬクレームが付くことを未然に防げることが実証実験の結果から予測できる。

是非ツール提供企業各位が早期に対応してくださることを期待する。

※１　手法・技法：SEC発行書籍「組込みソフトウェア開発向けコーディング作法ガイド［C言語版］」、「組込みソフトウェア向け開発プロセスガイド」、「組込みソフトウェア向けプロジェクトマネジメントガイド［計画書編］」

※2　ダウンロードファイルURL
https://sec.ipa.go.jp/download/200606eb.php
ファイル名「ツール諸元表雛形及び記載要領（Zip書庫）」

表２　ET2006でご協力頂いた13社

株式会社アドバンスドデータコントロールズ
株式会社ガイア・システム・ソリューション
ガイオ・テクノロジー株式会社
キャッツ株式会社
京都マイクロコンピュータ株式会社
株式会社CSKシステムズ
株式会社ソフィアシステムズ
株式会社東陽テクニカ
日本電気株式会社
日立ソフトウェアエンジニアリング株式会社
株式会社富士通ソフトウェアテクノロジーズ
マイクロソフト株式会社
横河ディジタルコンピュータ株式会社

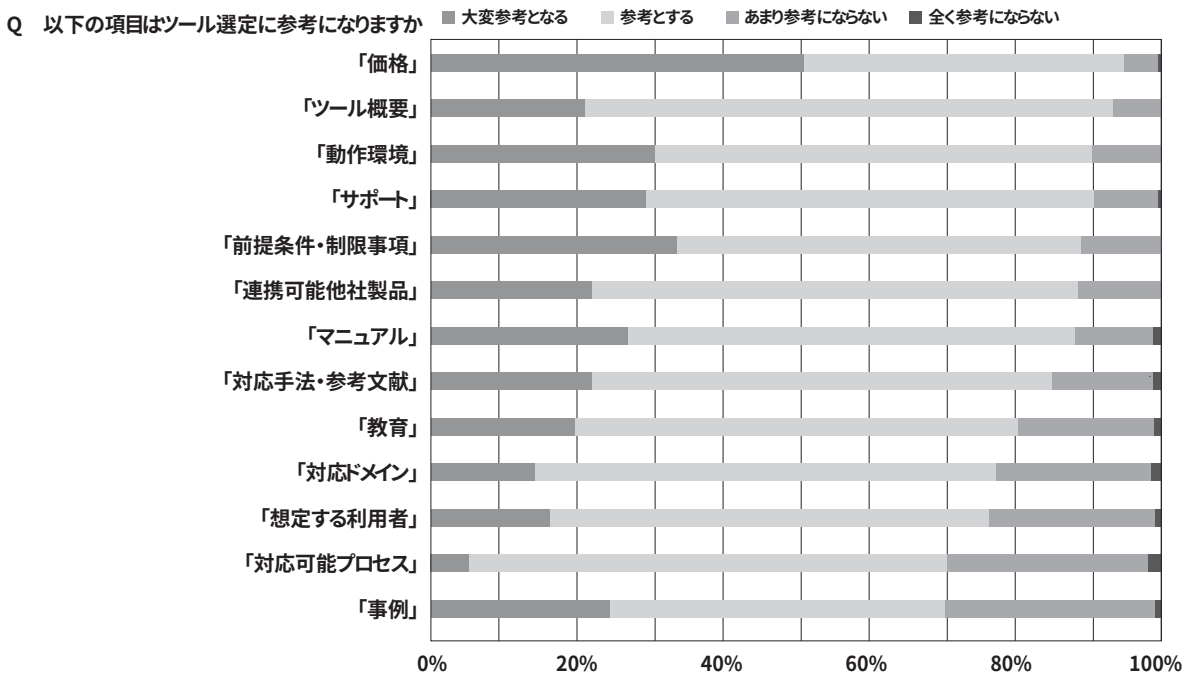


図１　ET2006におけるツール諸元表に関するアンケート調査結果（*n*=407名）

SEC 2006年度 活動概要 組込み系

組込みスキル標準（ETSS）

SECリサーチフェロー 大原 茂之
SEC研究員 佐藤 和夫
SEC研究員 関口 正
SEC研究員 渡辺 登

ETSS® Embedded Technology Skill Standards

組込みスキル標準（以下、ETSS）は、組込みソフトウェア開発力強化を目的とした、人材育成や人材活用を実現するための有効な指標となるよう策定されてきた。

ETSSを構成する要素は、組込みソフトウェア開発分野におけるスキルを体系的に分類し評価するための指標『スキル基準』、職種名称や職掌を定義する『キャリア基準』、教育カリキュラム作成のための構造や仕組み『教育研修基準』からなる。2006年6月に上記3つの各基準を正式バージョンとしたETSSを公開した。

1 2006年度活動テーマと総括

ETSS策定を担う組込みソフトウェア開発力強化推進委員会（スキル標準領域）では、「組込みスキル標準の適用拡大」を2006年度の活動テーマとして、ハードウェア担当者との協調、開発スキルや他のスキル標準との連携といった適用可能領域の拡大、これまでの成果物に対する改善等の審議を実施した。

当初、上記のテーマに関する審議を行うことによって、3つの基準の標準ドキュメントに大幅な改変が発生するものと予測したため、スキル基準とキャリア基準はVersion 2.0へのバージョンアップを予定していた。しかしながら、2006年度の審議結果から、これまで標準として定めた各基準のフレームワークや考え方を応用することで、これらの課題に対応可能であることが確認された。そのため、2006年度のETSSの改定内容は、限定的な細部調整事項が多いことからVersion 2.0とはせず、マイナーバージョンアップ（スキル基準Version 1.2、キャリア基準Version 1.1）に止めた。

以下に今年度の活動概要を説明する。

2 スキル基準部会（スキル基準）

スキル基準部会では、ETSSスキル基準の適用領域拡大として、組込みシステム開発におけるシステムLSIやSoC等のハードウェア開発技術への対応を検討した。

(1) ハードウェア開発技術スキルに関する検討

組込みソフトウェア開発において、ハードウェア（システムLSI等）開発担当者との協調開発やコンカレント開発に必要となるスキルについて、ETSSはどのように対応すべきかについて検討した。そのために、ハードウェア開発に関する有識者を新たにスキル基準部会に招聘した。

組込みシステム開発におけるソフトウェア開発とハードウェア開発とで使用する技術の比較を行い、特に開発技術スキルカテゴリに関してスキルの分類や評価が可能であるかを精査した。

その結果として、現行のスキル基準のフレームワーク等の基本部分は変更せずに、補足事項等を拡充する程度で対応可能であることが確認された。

(2) SEC成果物との連携検討

ETSSのスキル基準は、SEC成果物、例えば組込みエンジニアリング領域の成果物であるESPR（組込みソフトウェア向け開発プロセスガイド）、ESMR（組込みソフトウェア向けプロジェクトマネジメントガイド）、ESCR（組込みソフトウェア開発向けコーディング作法ガイド［C言語版］）等との連携について兼ねてから課題とされてきた。

検討を行った結果、これらのSEC成果物やETSSはまだ発展の途上であり、拙速に整合することは、メンテナンスや利用者側のバージョン管理を煩雑にする等のデメリット面が大きいとの判断がなされた。そのため、標準文書としては、これらのSEC成果物とはあえて整合しない方針とした。

今後、ETSSと共に、ESPRやESMRの導入企業が増えることが予測される。そのような企業等に向けた、ETSSスキル基準のESPRやESMR対応ガイド等の補足資料の提供を行っていく予定である。

(3) スキル基準部会2006年度成果

前述の検討結果により、2007年6月に、大幅改変を意味するVersion 2.0相当の変更とはせずに、表現方法等の改善を行ったスキル基準（Version 1.2）を公開する予定である。

3 キャリア開発部会(キャリア基準)

2006年度のキャリア開発部会では、ITスキル標準（以下、ITSS）等の既存スキル標準との整合や、プロフェッショナルコミュニティ検討等、今後の改善や普及に向けた検討を行った。

(1) キャリア基準の改善検討

ETSSに対して、4段階評価のスキルレベルと7段階評価のキャリアレベルの構造が難解であるとの意見がある。これらの意見は、ETSSの構造自体の問題ではなく、その構造を説明するドキュメント類のわかりにくさや説明不足に起因すると考えられる。これに応じて、ドキュメントの記載内容の追加や拡充等を実施することとした。

(2) ITスキル標準との連携検討

ITSSのキャリアフレームワークで定義された職種と、ETSSキャリア基準で定義している職種との間でシームレスな職種移動を実現するための方策について検討を行った。具体的にはITSSが公開している「スキルディクショナリ」の記述フォーマットを利用してETSSキャリア基準の職種が定義可能であるかについて試行した。

結果として、ITSSとETSSの基本概念とする部分の差分が大きく、ETSSとして重きを置く人材が持つスキル分布の度合いの表現が困難であること等が確認された。これは、表面的な体裁を整えることは可能ではあるが、実質的には別の職種を定義することになってしまう。

ここで明らかとなった差分は、ITSSやETSSの策定の前提として、どのような業界で、どのように利用されるかといった条件の違いによって生じたものと考えられる。

(3) 産業構造審議会人材育成WG

2006年の10月から経済産業省の産業構造審議会情報経済分科会情報サービス・ソフトウェア小委員会の中で、“高度IT人材育成メカニズムの構築に向けた課題”を検討するために有識者による人材育成WGが検討を開始した。この人材育成WGの検討課題の中に、ETSS、ITSS、UISS（ユーザスキル標準）といったIT人材に関係する各種スキル標準の整合や、情報処理技術者試験改革に対する方針が検討されている。

この人材育成WGで検討された方針に則り、ETSSと各スキル標準との整合を今後検討することとなる見通しである。

(4) プロフェッショナルコミュニティの検討

ETSSキャリア基準で定義した職種ごとの、人材育成のあり方や職種定義の具体化や改善・改良を活動目的としたプロフェッショナルコミュニティ設置に関する準備検討を行った。

ETSSキャリア基準の全職種について一斉にコミュニティを立ち上げるのではなく、ETSSの活動と既に協力関係があり、職種との関連性が強い既存の団体やコミュニティ等と協調しながら現実的かつ有益な運営を順次試行していく。

(5) キャリア開発部会2006年度成果

2007年6月に、前述の検討結果と表現方法の改善を行ったキャリア基準（Version 1.1）を公開する予定である。

4 教育部会（教育研修基準）

2006年度の教育部会では、組込みソフトウェア開発分野における人材育成に関する現状の課題を整理し、次年度以降に確実に施策化あるいは標準化すべき事項を明確にするための審議を活動の中心とした。

効率よく高密度の審議を円滑に進めるために検討テーマのグルーピングを行い、3つのグループ（「人材インフラ検討グループ」「エントリ教育検討グループ」「プロフェ

ッショナル教育検討グループ」）で検討を行った。

（1）人材インフラ検討グループ

人材インフラ検討グループでは、人材育成に関する活性策や意識啓発等、組込みソフトウェア開発人材育成の基幹や基盤形成に関する課題や方策について検討を行った。人材インフラ検討グループで検討された主要なテーマを以下に記す。

①若年層に対する施策検討

これまで教育部会では、組込みソフトウェア開発分野の人材や大学生等を対象とした分析や検討を中心に活動を行ってきた。ところが少子化の進行や若年層（高校生以下）の理工離れ等によって、組込みソフトウェア開発分野への人材の主要な供給源とされる、情報系学科や電子系学科の学生数の減少や学力低下が顕著となってきた。組込みソフトウェア開発分野の人材の質と量を確保するにも、大学や専門学校等の学生を対象とした施策だけでは対応しきれない状況となりつつある。

このような状況に応じるために、若年層に対して組込みソフトウェア開発をはじめとする、わが国の「ものづくり」に対する興味や学力を育む施策が必要である。

②人材育成インフラフレームワーク

これまでの開発現場や高等教育機関（大学・専門学校）の人材に向けた教育カリキュラムの供給だけではなく、若年層（小中高生）も包括した人材育成の基盤（インフラ）の枠組み（フレームワーク）が必要である。

人材インフラ検討グループでは、人材育成インフラフレームワーク案を作成した。このフレームワーク案は、人材育成の段階をStage（育成ステージ）として「誘導」「体験」「入門」「専門」と定義し、人材育成の施策をActivity（育成アクティビティ）として「プロモーション」「カリキュラム（コンテンツ）」「指導者・教員」「教育評価」と定義した。

育成ステージと育成アクティビティをマトリクス状に交差させ、施策の具体化と施策各々の位置関係を可視化する。

（2）エントリ教育検討グループ

エントリ教育検討グループでは、組込みソフトウェア開発分野のエントリ人材育成に関する課題と施策の検討を行った。エントリ教育検討グループで検討された主要テーマを以下に記す。

①未経験者向け教育カリキュラムの改善

ETSSの教育研修基準では、組込みソフトウェア開発分野へ参入してくる人材（新入社員、大学生、他分野のソフトウェア技術者等）を対象としたカリキュラムとして「組込みソフトウェア開発未経験者向け教育カリキュラム」を提示している。

2006年度は、当教育カリキュラムに対して、カリキュラムを提供する側と受講する側のそれぞれの利用局面から改善検討を行った。その結果、「構成する科目が大きく使いづらく、利用者負担が大きい」「科目設計をする際に必要な情報の不足」等の要改善要項が抽出された。

②組込みソフトウェア開発人材育成教材に関する調査

組込みソフトウェア開発分野の人材育成教材をいかに普及、流通させていくかについて検討する前の予備調査として、教材に関する実態調査の方法について審議を行った。また、審議結果に基づき、2007年3月にSEC-Webサイト上でアンケートを実施した。この調査の集計結果は、組込みソフトウェア開発人材育成のための教材に関する検討を行う際に今後活用する。

（3）プロフェッショナル教育検討グループ

プロフェッショナル教育検討グループでは、企業内で行われる技術者教育のあるべき姿と現状を対比することで、組込みソフトウェア開発分野ですでにプロフェッショナルとして活躍している人材の育成に関する課題を明らかにした。プロフェッショナル教育検討グループで検討された主要テーマを以下に記す。

①組込み開発に適合したOJTプログラムの必要性

経済産業省が実施している「組込みソフトウェア産業実態調査報告書」からも、組込みソフトウェア開発分野の人材育成の有効な手段の1つとしてOJTは認知されては

いるものの、OJT計画の作成・実施・評価を継続的に実施している組織は稀であることが改めて確認された。

組織の事業戦略と連携した育成方針に基づいた、組込み開発に適合したOJTを実現するためのガイド作成等の検討と施策化を今後進めていく。

②人材育成のPDCAサイクルの確立

組織が事業を遂行するために必要となる人材・スキルを戦略的に確保するためには、組織の事業戦略と連動した、人材育成のPDCAサイクルをまわすことが有効である。しかし、戦略的なPDCAサイクルを組織の中でまわすためには、経営者や事業責任者の理解と協力が不可欠である。

経営者や事業責任者への人材育成に対する啓発を目的としたガイド作成等について今後検討を行う。

(4) 教育部会2006年度成果

前述の検討結果と今後の指針について活動報告書として取りまとめたものを2007年6月に公開する予定である。

また、今年度の検討結果として明らかになった改善事項に対応した、教育研修基準（Version 1.1）を2007年秋以降に公開することを目指す。

5 ETSS実証実験の継続実施

2005年度から実施しているETSS実証実験を、2006年度も継続して実施した。2006年度の実証実験は、個別の企業だけではなく複数の団体に対しても実施した。これらの団体は、いわゆる組込みソフトウェア開発分野以外の領域を対象としているものもあり、他領域へのETSS適用事例として、実証結果を得られるものと期待している。

6 ETSSプロモーション活動

SEC研究員を中心に、ETSS普及や啓発を目的として次のような活動を行った。

①チュートリアルセミナーの開催

ETSSの運用及び導入に関するチュートリアルセミナーをSEC主催セミナーの枠内で2回実施した。

その他にも、展示会におけるセッション、協力企業や団体のイベント等で、ETSSに関するプレゼンテーションを実施した。

②組込みスキル標準（ETSS）概説書の発行

ETSS活用の入門書として、2006年度も概説書を発行した。本概説書は上記のチュートリアルセミナーや展示会等の関連イベント等を利用し配布を行った。

7 2007年度以降の課題

ETSSに関して今後対応すべき課題を以下に記す。

①組込みスキル標準（ETSS）の改善

実運用や実証実験等によって、ETSSに対して改善を必要とする課題が顕在化した場合、今後も改善を行っていく。

特に2006年度の検討で明らかになった教育研修基準の改善課題については2007年度以降に対応を実施する。

②人材育成WG検討結果方針の対応

産業構造審議会人材育成WGによって取りまとめられた指針に基づいた、他の各スキル標準との整合を検討する。

上記の課題のほかにも、国際標準化や認証・認定に関するスキームの確立に向けた検討及び活動を適時進めていく。

SEC 2006年度 活動概要 共同研究

奈良先端科学技術大学院大学（http://se.naist.jp/）

COSEにおけるデータ分析とフィードバック

奈良先端科学技術大学院大学
情報科学研究科 教授
松本　健一

奈良先端科学技術大学院大学
情報科学研究科 准教授
門田　暁人

同志社大学文化情報学部
准教授
宿久　洋

ソフトウェア機能と開発プロセスの細分化は、ソフトウェアの信頼性と生産性の向上をもたらしたが、同時に、開発作業の状況把握を難しくしている。本研究では、ソフトウェアエンジニアリング技術研究組合（COSE）が実施した「プローブ情報プラットフォームソフトウェア」の開発において、EASEプロジェクト[EASE]が開発したデータ自動収集分析ツール群を適用し、基本分析までの自動化を図り、運用を含めたリアルタイムフィードバックシステムについて検討を行った（図１）[PRO2007]。

1 成果のポイント

分析作業のほとんどについて日次での運用が可能となることが確かめられ、分析の粒度も「社別」から「コンポーネント別」へと細分化できた。さらに、「相関ルールおよび例外ルールに基づく分析」[RULE2007]、「GQMモデルに基づく上流工程分析」[GQM2007]、「障害分析に基づくプロセス改善指針の抽出」[ODC2007]、「再利用における仕様書・設計書レビュー分析」[REVIEW2007]といった、従来よりも多面的で詳細な分析を行うことができた。

2 成果の詳細

ここでは、リアルタイムフィードバックの状況とその結果の概要について述べる。

・WebDAVを用いることで、開発データはネットワーク経由で提出可能となり、機密性を保持しながら、データ提出コストが低減し、提出遅れのフォローも適宜行えるようになった。

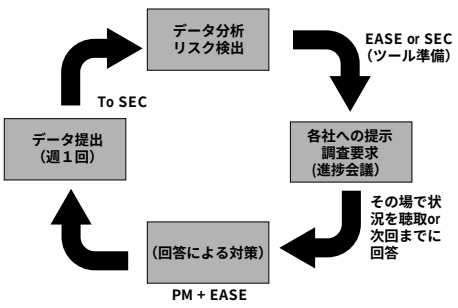


図１　COSEにおけるデータ分析サイクル

表１　調査要求（指摘）に対する回答数

調査要求（問題指摘）時期 ＼ 回答	プロジェクト管理上、対処が必要な問題	データ不整合、入力ミス	ツール運用誤り,データ提出漏れ	未回答/不明	合計
設計工程品質評価会議	34	7	3	1	45
週次フィードバック	35	11	3	10	59
社内試験工程品質評価会議	14	11	3	4	32
合計	83	29	9	15	136

・データ分析の結果に基づいて、開発ベンダに対して136件の調査依頼（問題指摘）を行った（表１）。そのうち約61％にあたる83件は、「プロジェクト管理上、対処が必要な問題」との回答を開発ベンダから得た。データの不整合や入力ミス、ツール運用の誤りやデータ提出漏れなどを除くと、問題につながらない調査依頼（指摘）は15件（約11%）にすぎなかった。

・分析作業のほとんどを日次で運用し、分析の粒度も「ベンダ別」から「コンポーネント別」へと細かくできたことは、これまで以上に多様な分析ときめ細かなフィードバックへの道を開いたといえる。

3 今後の展開

参加ベンダの開発体制や契約条件に応じて分析内容やフィードバック方法を変える等、より実用的な手法の開発が今後の課題である。

参考文献

[EASE] http://www.empirical.jp/
[GQM2007] 松村知子他：GQMモデルに基づく設計工程完成度計測手法の提案，奈良先端大テクニカルレポート，NAIST-IS-TR2007005，Mar. 2007
[ODC2007] 松村知子他，プロセス改善を目的とするODCを用いた欠陥修正工数分析，奈良先端大テクニカルレポート，NAIST-IS-TR2007006，Mar. 2007
[PRO2007] 松村知子他：自動データ収集・可視化ツールを用いたリアルタイムフィードバックシステムの構築と試行，奈良先端大テクニカルレポート，NAIST-TR-2007001，Feb. 2007
[REVIEW2007] 松村知子他：設計書の再利用を考慮したレビュー効率の比較方法の提案と事例紹介，奈良先端大テクニカルレポート，NAIST-IS-TR2007007，Mar. 2007
[RULE2007] 森崎修司他：相関ルール分析を用いた障害対応データの特徴分析，奈良先端大テクニカルレポート，NAIST-TR-2007003，Feb. 2007

SEC 2006年度 活動概要 共同研究

東海大学（http://www.u-tokai.ac.jp/）

プロジェクトの遂行及び生産性に影響を与える要因の分析

東海大学理学部
教授
古山　恒夫

ソフトウェアプロジェクトの実績データから、生産性・工期・信頼性に影響を与えるさまざまな要因を分析・抽出することは、プロジェクトの計画作成や運営上の指針を得る上で重要なことである。2006年度は、2005年度のSEC収集データ[SEC2006]から、ソフトウェアプロジェクトの遂行に影響を与える要因及びソフトウェアの生産性に影響を与える要因をそれぞれ明らかにすることを目的として調査研究を進めた。

1 成果のポイント

(1) プロジェクトの遂行に影響を与える要因の分析

ソフトウェアプロジェクトが、コスト・工期・品質それぞれについて計画どおりに遂行できなかった要因を明らかにした。

(2) ソフトウェアの生産性に影響を与える要因の分析

工数に影響を与える量的データ項目と、生産性に影響を与える質的データ項目を抽出した。

2 成果の詳細

(1) プロジェクトの遂行に影響を与える要因の分析

新規開発ソフトウェアプロジェクトが計画どおりに遂行できず、コスト超過・納期遅延・品質低下を起こしてしまう主な要因として次のものがある[FURUYAMA2006]。

- プロジェクト規模そのものが大きく、特に工期当たりの規模が大きいプロジェクトはコスト超過を起こす割合が高い。
- 要求仕様があいまいなプロジェクトは納期遅延や品質低下を起こす割合が高い。
- 事前に工期の妥当性を評価したプロジェクトでは納期遅延を起こす割合が低い。
- 業務分野の経験者を揃えたプロジェクトでは納期遅延や品質低下を起こす割合が低い。
- テスト体制を整備したプロジェクトでは、コスト超過・納期遅延・品質低下を起こす割合が低い。

(2) ソフトウェアの生産性に影響を与える要因の分析

新規開発ソフトウェアの生産性に影響を与える主な要因として次のものがある[FURUYAMA2007]。

- 工数は規模だけでなく開発時のバグ数にも大きな影響を受ける。
- セキュリティに対する要求レベルが高いプロジェクトやスキル・員数とも不十分なテスト体制をもつプロジェクトは生産性が低い傾向がある。
- スキルの高いPMがテスト工程で多くのバグを検出できるようにプロジェクトを運営している場合、生産性が低い傾向がある。
- ユーザ担当者の要求仕様への関与が高いプロジェクトは生産性が高い傾向がある。
- 作業スペースや騒音環境は単独では影響要因とならないが、他の要因の影響を強める効果をもっている。

3 今後の展開

今後は、さらにデータ分析を進めて、生産性に次いで信頼性に影響を与える要因の分析を行う。また、生産性に影響を与える要因の具体的な因果関係を明らかにしていく。

参考文献

[FURUYAMA2006] 古山恒夫，菊地奈穂美，安田守，鶴保征城：ソフトウェア開発プロジェクトの遂行に影響を与える要因の分析，ソフトウェアエンジニアリング最前線2006，pp. 43-50，近代科学社，2006
[FURUYAMA2007] 古山恒夫：エンタプライズ系ソフトウェアの生産性に影響を与える要因の分析，情報処理学会研究報告，Vol. 2007-SE-155，pp.73-80，2007
[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006

SEC 2006年度 活動概要 共同研究

大阪大学大学院（http://www-ise4.ist.osaka-u.ac.jp/）

相関ルールマイニングを用いたソフトウェア生産性の決定要因抽出

大阪大学大学院情報科学研究科 助教 水野 修
大阪大学大学院情報科学研究科 博士後期課程2年 浜野 康裕
大阪大学大学院情報科学研究科 教授 菊野 亨
SEC研究員 菊地 奈穂美
SEC研究員 平山 雅之

これまで、我々の研究室ではロジスティック回帰モデルやベイズ識別器を用いて、ソフトウェア開発に関する種々の要因がプロジェクトの混乱状態に与える影響について研究を行ってきた。しかし、欠損値が含まれるデータへの適用や、各要因が混乱状態に与える影響の強さの特定などが問題として残されてきた。そこで、本研究ではプロジェクト混乱の一要素である生産性について相関ルールマイニングによる特徴抽出を試みた。

1 成果のポイント

本研究では相関ルールマイニングを用いて、生産性の決定要因に関する分析を行った。相関ルールマイニングとはデータマイニング手法の一種であり、特に事象間の強い関係をルールとして抽出する手法である。本手法を用いることで、データの一部に欠損を含んでいるデータからも、生産性に関する有益な情報を抽出することができる。また、単に生産性に強い影響を与える要因を抽出できるだけではなく、要因の組み合わせに関する情報も得ることが可能となる。これにより生産性の良否を判断するための有益な材料を多く得ることができる。

2 研究成果

(1) 相関ルールマイニングとは

相関ルールマイニングは、相関ルール（以下、ルール）と呼ばれる、事象間の強い関係を知識として発見する分析手法である。データ集合全体を分析して、「ある事象Aが発生するならば別の事象Bも発生する」という事実を発見し、それをルールとして抽出するものである。

(2) 利用したデータ

今回利用したデータは、SEC収集データ[SEC2005]に記載されている1,009件である。

(3) 生産性の定義

本研究では，生産性Pを次のように定義した。

P＝log（ファンクションポイント／総工数）

ここで、総工数とは実績工数の5項目（基本設計、詳細設計、製作、結合テスト、総合テスト）の和である。また、生産性良否の基準としてあるプロジェクトの生産性Pの値が「Pの平均－Pの標準偏差」以下ならば「生産性が悪い」、「Pの平均＋Pの標準偏差」以上ならば「生産性が良い」とした。

(4) 抽出した生産性要因

相関ルールマイニングの結果、生産性に関する以下の4つのパターンが存在することがわかった。

① 生産性が良いパターン1：母体システムが非常に安定、かつチーム内での役割分担・責任所在が明確、かつ信頼性の要求レベルが低い
② 生産性が良いパターン2：チーム内での役割分担・責任所在が明確、かつユーザ担当者は受け入れ試験に関与しない、かつ信頼性要求レベルが低い
③ 生産性が良いパターン3：セキュリティ要求レベルが低い、かつプロジェクト全体の期間が長い
④ 生産性が悪いパターン：チーム内での役割分担・責任所在が不明確、かつ作業スペースが狭い、かつ処理形態が対話処理

なお、分析の詳細は[HAMANO2007]にある。

3 今後の展開

今後は、プロジェクトの混乱状態を回避するための方策をルールマイニングによって発見することなどが期待できる。

参考文献

[HAMANO2007] 浜野康裕，水野修，菊野亨，菊地奈穂美，平山雅之：相関ルールマイニングの適用によるソフトウェア生産性の決定要因の分析，情報処理学会第155回SIGSE，2007-SE-155，pp.65-72，2007
[SEC2005] IPA SEC：ソフトウェア開発データ白書2005，日経BP社，2005

SEC 2006年度 活動概要 共同研究

奈良先端科学技術大学院大学（http://se.naist.jp/）

異なるFP手法間における FP変換式の導出

奈良先端科学技術大学院大学
情報科学研究科 准教授
門田　暁人

奈良先端科学技術大学院大学
特任助教
角田　雅照

奈良先端科学技術大学院大学
情報科学研究科 博士前期課程2年
松本　真佑

ソフトウェアの規模を示すファンクションポイント（FP）を計測する手法には様々なものが存在し、規模が同じシステムであっても、FP計測手法によって値が異なる。また、FP計測手法が異なるシステム同士の規模（FP）を比較することができないという問題もある。本研究では、FP手法間のFP変換式の導出方法について検討し、適用の試行を行った。

1 成果のポイント

「FP手法の違いは、FP当たりの必要な開発工数、すなわち生産性（工数÷FP）の違いとしてデータに現れる」と仮定し、図１のようにFP手法間で生産性を一致させるように変換式を導出する方法を開発した。以下に手順を述べる（図２）。

手順１　生産性の変換

平均値と分散を考慮した変換式（Z-SCORE法）により生産性を変換する。

手順２　FPの変換

FP=生産性×工数より、工数と変換済み生産性を用いてFPを変換する。

手順３　FPの変換式導出

変換前のFPと変換後のFPを用いて回帰分析を行い、FPの変換式を求める。

生産性にはFP計測手法以外の要因も影響する[SEC2006][TSUNODA2006]。そのため、生産性に影響する要因が一致するプロジェクトのみを用いて変換式を求めることにより、それらの要因の影響を排除することができ、より正確な変換式を得ることができる。

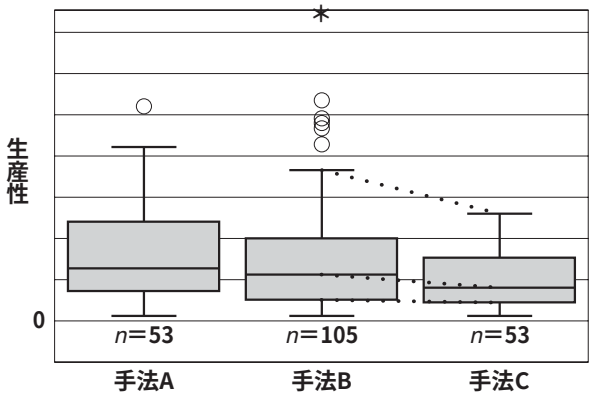
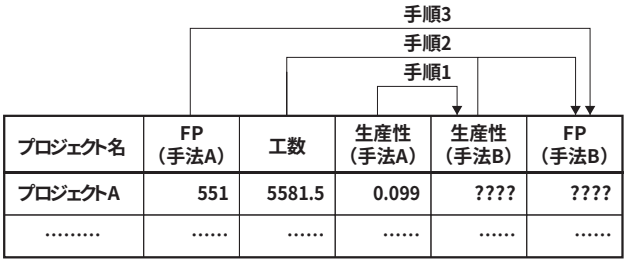


図１　FP手法間の生産性の分布の対応付けのイメージ



プロジェクト名	FP（手法A）	工数	生産性（手法A）	生産性（手法B）	FP（手法B）
プロジェクトA	551	5581.5	0.099	????	????
……	……	……	……	……	……

図２　変換式の導出手順

2 成果の詳細

SECにおいて収集されたSEC収集データ[SEC2006]から295件のデータを抽出し、ケーススタディを行った。平均要員数が中程度のプロジェクトのみを用いてFPの変換式を求めた結果、大まかな傾向としては、IFPUGと比較して、SPRとIFPUG改は値が大きめに出る傾向にあり、NESMA概算はほとんど差がないという結果となった。

3 今後の展開

本研究ではデータ件数が少ないことから、平均要員数以外の要因（業種、外部委託率、アーキテクチャ等）を一致させて変換式を導出することができなかった。そのため、本研究において見られた変換式の傾向は、現時点ではあくまでも参考程度に止めておき、今後データ件数を増やして、より正確な変換式を導出する必要がある。

参考文献

[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006
[TSUNODA2006] M. Tsunoda他：Productivity Analysis of Japanese Enterprise Software Development Projects，In Proc. Int'l Workshop on Mining Software Repositories，pp.14-17，Shanghai, China, 2006

SEC 2006年度 活動概要 共同研究

愛媛大学（http://www.ehime-u.ac.jp/）

推定・近似に基づいた機能規模計測法間での変換法

愛媛大学大学院理工学研究科
講師
阿萬　裕久

ソフトウェア品質に関する重要な属性としてファンクションポイント(FP)がある。FPは広く用いられているが、そこには複数の計測法が存在しており、業界で統一されているわけではない。それゆえ企業横断的に収集されたSECデータにおいても異なる計測法によるFP値が混在している。そこで本研究では、異なる計測法間での対応付けを目指し、SEC収集データ[SEC2006]におけるFP値及び各プロジェクトデータに対する統計解析を通じてFP計測法間での変換式の構築を試みた。結果として、一部のFP計測法間について信頼度の高い変換式を構築できた。

1 成果のポイント

本研究では、SEC収集データ[SEC2006]（1,419件）を用い、各FP計測法でのFP値を相互に変換するような変換式の構築を目的とした。今回は、データ件数の比較的多い３計測法（IFPUG法、SPR法、IFPUGカスタマイズ法）に注目した。これらに対し、以下に示す2つの分析手法を考え、FP値変換式の構築を検討した。

（1）分析手法1（重回帰分析によるFP値推定）

各FP計測法に対し、FP値を目的変数、プロジェクトデータを説明変数とした重回帰分析を行い、SEC収集データからFP値を推定する重回帰式を構築した。ただし、説明変数については表１の条件で絞り込みを行った。そして、各FP計測法におけるFP推定値を算出し、最小2乗法を使ってFP値間での変換式を構築した（図1)。

表１　データ項目の絞り込みにおける条件

	条　件
1	新規開発プロジェクトのデータである。
2	FP値（実績値）との間に有意な相関関係がある。 ※Spearmanの順位相関での無相関検定（有意水準5%）
3	欠損値・外れ値でないデータが30件以上ある。

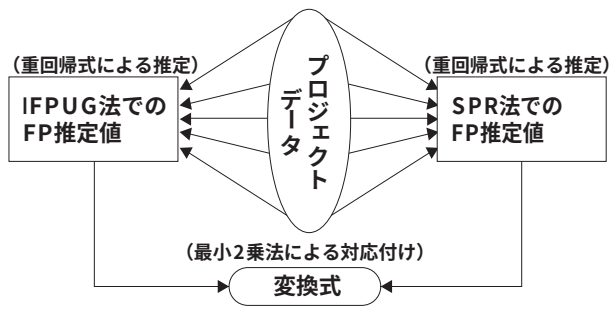


図1　変換式構築のイメージ（IFPUG - SPR間の場合）

その結果、SPR法－IFPUGカスタマイズ法間で信頼度の高い変換式を構築できた。しかし、他の組み合わせではFP値の推定がうまくいかず、信頼できる変換式は構築できなかった。主な原因として、欠損値の影響により説明変数に3～5個の項目しか使えなかったことが挙げられる。詳細は割愛するが、FP値推定に影響の大きいデータ項目として、トランザクションファンクション及びデータファンクションの実績値（特にILF実績値の影響が大きい)、実績工数、ピーク時の要員数等があった。これらを中心により多くのデータが収集されることでFP値の推定・変換の信頼性を高めることができると思われる。

（2）分析手法2（FP詳細データによる計算・近似）

トランザクションファンクション及びデータファンクションの実績値（FP詳細データ）を別の計測法でのFP値（あるいは近似値）算出へ流用し、それらに基づいて変換式を構築した。結果として、SPR法－IFPUGカスタマイズ法間については信頼できる変換式が得られたが、他の組み合わせについてはデータ件数が10件にも満たず結果の有意性を評価できなかった。FP詳細データがより多く得られれば変換式の信頼性も確保でき、分析手法１との比較・検討も可能になると考えられる。

2 今後の展開

複数のデータ項目を組み合わせると欠損値が多く現れ、利用できるデータ件数も限られる。今後、小標本を対象とした統計解析手法の適用も検討する必要がある。

参考文献

[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006

SEC 2006年度 活動概要 共同研究

鳥取環境大学（http://www.kankyo-u.ac.jp/）

相互比較可能な形式へのFP値の変換方法に関する検討

鳥取環境大学環境情報学部
助教
天嵜　聡介

現在SECでは企業横断的にプロジェクトデータを収集されている。特に重要な収集項目として規模を表すファンクションポイント（FP）がある。FPの計測手法は複数提案されており、企業により採用しているFP計測手法は異なっている。そのため、現状ではプロジェクトデータ中に複数のFP計測手法によるFP値が混在している。このデータをプロジェクト・ベンチマークとして有効に活用するためには、計測手法が異なるFP値の相互比較を可能にする必要がある。そこで本研究においては、異なるFP計測手法で計測されたFP値を、相互に他のFP計測手法によるFP値へと変換する方法の検討に取り組んだ。

1 アプローチ

本研究のアプローチを図1に示す。今回はSEC収集データ[SEC2006]に記載されているFP計測手法のうち、特に全体に占めるデータ数が多い4つのFP計測手法を対象とした。

まず、検討1では、測定対象から収集するデータ項目が完全に一致するFP計測手法（手法グループ1）の間での変換方法を検討する。次に、検討2では、手法グループ1に含まれなかった手法と、手法グループ1に含まれる手法のうち最もデータの件数が多いFP計測手法の2つの手法（手法グループ2）との間で変換方法を検討する。検討3では検討1、検討2の結果得られる変換手法の一群を統合する方法について検討する。そして、最後に検討4で検討3の結果の検証方法について検討を行う。

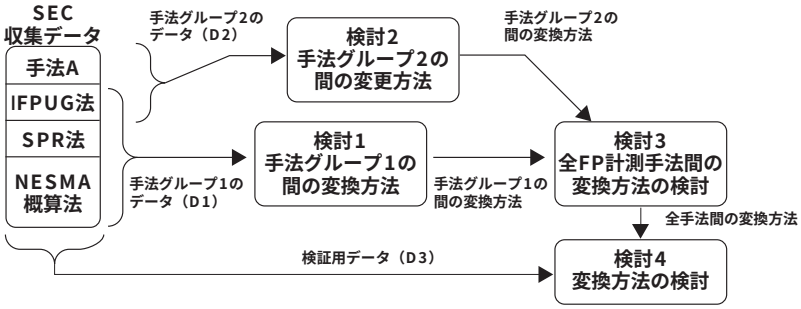


図1　研究のアプローチ

2 成果のポイント

分析の信頼性を確保するために入念なデータのチェック及び選別を行ったため、本研究で分析に利用したデータの件数はかなり少ない。そのため、変換方法の確立は一部のFP計測手法の間のみに止まった。

検討1の結果、今回検討した統計モデルの中では以下のモデルが変換精度に関して最も良好な結果を収めた（εは平均0の正規分布に従う誤差）。

$$(\text{FP計測手法Y}) = \beta_0 (\text{FP計測手法X})^{\beta_1} + \varepsilon$$

一方、検討2では変換方法の確立には至らなかった。対象のグループのデータD2の件数そのものが少なかったことに加えて、FP値の計測の基礎となるメトリクスのほとんどが欠測で十分な分析ができなかったためである。

以上の理由により検討3は未実施であるが、検討4は検討1の結果に対して部分的に実施することができた。その結果、検討1の結果は比較的条件の限られたプロジェクトのデータに対して精度の高い変換手法である一方で、一般性についてはより多くのデータによる検証が必要であることが確認された。

3 今後の展開

今後は、利用できるデータの量が増えることにより、今回検討を断念したFP計測手法間での変換方法や、検討1の結果の一般性の検証についての研究が進展すると考えられる。

参考文献

[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006

SEC 2006年度 活動概要 共同研究

大阪大学（http://sel.ist.osaka-u.ac.jp/）・大学院（http://www.ist.osaka-u.ac.jp/）

先進ソフトウェア開発プロジェクトにおけるソースコード分析

大阪大学・大学院情報科学研究科 教授 楠本 真二
大阪大学・大学院情報科学研究科 博士後期課程2年 吉田 則裕
大阪大学・大学院情報科学研究科 博士前期課程2年 馬場 慎太郎
大阪大学・大学院情報科学研究科 教授 井上 克郎

効果的なプロジェクト管理を実施するためにはプロジェクトの現状を正確に把握することが必要である。プロジェクトの現状把握には、いわゆる「見える化」が有用であり、そのための様々な見える化技術が提案され使用されている。

本研究では、特にソフトウェア開発の実装段階以降を対象として、ソースコードに対して適用可能な分析技術であるコードクローン分析技術[HIGO2007][HIGO]と複雑度メトリクスを用いた分析手法に関して検討を行った。また、2006年度と2005年度の先進ソフトウェア開発プロジェクトで開発されたソースコードに対して分析手法を適用し、ソースコード上に現れる特徴について評価した。

1 ソースコード分析の概要

分析対象のソースコードは５種類、約8,000行から35,000行の規模であった。主な分析としては、コードの重複度（ファイル全体のトークン数に対するいずれかのコードクローンに含まれるトークン数の割合）の算出、散布図上で目立ったコードクローン、メトリクス値として際だった値を持つコードクローンに対する詳細分析、各々の複雑度分析を実施した。

図１に１つのソースコード群に対する年度間のコードクローン散布図を示す。原点を左上隅として、水平・垂直方向にソースコード中のトークンを出現順に配置し、水平方向のトークンと垂直方向のトークンが等しい箇所に点をプロットした。その結果、図中ではクローンの対は点の連続した線分として出現した。

図中で、Xの部分は2005年度の、Yの部分は2006年度の、Zの部分は年度間のコードクローン散布図をそれぞれ表す。図１では、A～Fの部分をクローンとして抽出した。これらのコードクローンは、基本的には2005年度に開発されたコードが、2006年度にもある程度まとまって再利用されたものであった。

また、抽出したコードクローンに対して、ソースコードで内容を確認し、その特徴を調査した。インタビューの結果、ほとんどのコードクローンについて開発者・管理者が把握していることが確認できた。一方で、コードクローンに対する不具合修正が不十分であったデータもあり、ソースコード修正時にコードクローン分析を実施し、漏れなく該当箇所を検出するということの重要性が確認できた。

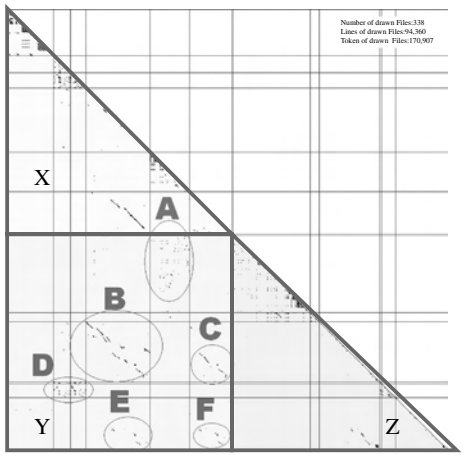


図１　年度間のクローン散布図

2 今後の展開

2006年度の先進ソフトウェア開発プロジェクトは2005年度のソフトウェアに対する保守プロジェクトと位置づけることができる。今後引き続き保守を実施するという立場からは、機能追加・変更時に考慮すべきコードクローンの情報を抽出できたと考えている。

参考文献

[HIGO] Y. Higo, T. Kamiya, S. Kusumoto, K. Inoue：Method and Implementation for Investigating Code Clones in a Software System, Information and Software Technology (in press)
[HIGO2007] 肥後，吉田，楠本，井上：産学連携に基づいたコードクローン可視化手法の改良と実装，情報処理学会論文誌，Vol.48，No.2，pp.811-822，2007

SEC 2006年度 活動概要 共同研究

東京大学COEものづくり経営研究センター（http://www.ut-mmrc.jp/）

組合せ・すり合わせ視点での組込みソフトウェア分析調査

東京大学COEものづくり経営研究センター センター長
東京大学大学院経済学研究科 教授
藤本 隆宏

東京大学COEものづくり経営研究センター
特任助教
立本 博文

組込みソフトウェア分野は、わが国の国際競争力維持のためにも強化が不可欠な分野である。しかしながら、現実の製品開発における組込みソフトウェア開発プロジェクトでは、必ずしも芳しい成果をあげているとはいえない。一方で、国際的に見て日本企業は「すり合わせ」開発は得意であり、組込みシステム／組込みソフトウェア開発に向いているという主張もある。また、ソフトウェア分野こそ、物理制約がなく、モジュール化がしやすいので「組合せ」開発が有効であるとの主張もある。東京大学COEものづくり経営研究センターとIPA/SECは共同して実態分析を行った。

1 実施内容及び成果

「平成18年度　組込みソフトウェア産業実態調査」のプロジェクト責任者向け調査データを分析対象の元データとした（n=548）。この母集団データから、プロジェクト責任者の主観評価（4段階尺度）により、QCDパフォーマンスが良好であるもの（評点=4）、そうでは無いもの（評点=1）を取り出し、11項目について対比を行った（抽出プロジェクトn=96、重複有り）。理論仮説では高成果集団の方が低成果集団よりも指標が高いと予想される。分析結果を表1に示す（紙面の関係上、詳細な結果の説明は別の機会に譲る）。 11指標中、3指標に関して予想と反対の結果となった。コストに関する指標に関して仮説と分析結果がすべて一致した。仮説と異なる結果となった仮説と不一致の項目を精査すると、すべて「すり合わせ」要素を示す項目であった。この点に注目すべきである。

表1　分析結果

	検定項目数	仮説と一致	仮説と不一致
品質	3	1	2
コスト	3	3	0
開発リードタイム	5	4	1
合計	11	8	3

2 インプリケーション

本分析をもってすぐに強い結論を得ることは難しいが、別途行ったヒアリング調査と併せて考えると、次のように考えることができる。現在の組込みシステムの製品開発プロセスにおいて、すり合わせメカニズムが有効に機能していない可能性がある。その直接的な理由として、過度に高い製品パフォーマンスを求める「すり合わせ過剰」や、手戻り工数と品質向上のためのすり合わせ工数を混同する「無駄なすり合わせ」の問題がある。「すり合わせをしているけれども、成果が上がらない」という錯覚が開発現場に現れている。「すり合わせ」領域があまりに広く、「すり合わせ」が機能不全に陥っている。全開発量に対して「すり合わせ」領域と「組合せ」領域をもっと明確にすることが必要であり、例えばプラットフォーム型の開発プロセスは、このような視点から強く求められている。ただし、プラットフォーム型の開発プロセスを実現するためには、現場のエンジニア視点だけでなく、事業戦略視点も入れ込んだロードマップを書く必要がある。この意味では、プラットフォーム型の開発は、開発組織と事業戦略組織が分離していては機能しない。極めて「組織的な問題」であるが、現状ではこの問題を解決している事例は少ない。

3 今後の展開

今回の調査分析の成果をもとに、さらに製品開発プロセスに入った調査が必要である。特にプラットフォーム型開発は、すり合わせ量の爆発の問題に対応する有効な手法であると考えられるが、成功事例は少なく、その要因解明をしていく必要がある。

SEC 2006年度 活動概要 共同研究

北陸先端科学技術大学院大学（http://kt-www.jaist.ac.jp/index-j.html）

形式検証技術の設計検証への実用化に向けて

北陸先端科学技術大学院大学
情報科学研究科　特任教授
岸　知二

北陸先端科学技術大学院大学
情報科学研究科　特任准教授
青木　利晃

北陸先端科学技術大学院大学
情報科学研究科　教授
片山　卓也

組込みソフトウェアの信頼性は、その規模と複雑度の増大、開発期間の短縮などに伴い、ますます重要な課題となっており、形式的手法に対する期待が高まっている。こうした背景の中、2005年度は文献調査やコンパクトな事例を用いた適用試行により、形式手法の組込みソフトウェア開発に対する適用性に関しフィージビリティスタディを行った。2006年度は、IPA/SECと北陸先端科学技術大学院大学（JAIST）双方のメンバが協調して事例に対する適用試行を行い、適用上の手法、効果、課題を明らかにした。

1 成果のポイント

今年度の活動において、一般的に認知度の高い機器であり、開発時の不具合が想定できるものとして「ＣＤプレーヤ」を取り上げ、実際の組込みソフトウェアの開発にて使用されると想定される次の３つのアプローチを実施することにした（図1）。

(1) 実施した3つのアプローチ

①ツール使用によるアプローチ

モデルチェッカであるSPIN を利用したUML設計に対するモデル検査支援ツールを使用したアプローチ。支援ツール上にて状態モデルを描画し、検証モデルへの変換を行う。

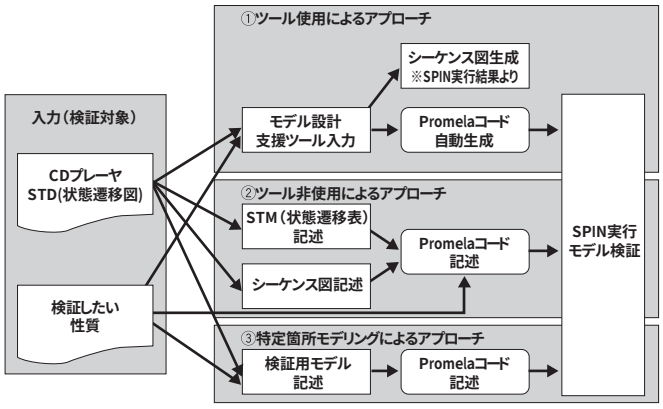


図１　例題に対する３つのアプローチ

②ツール非使用によるアプローチ

形式的検証を行う箇所についてはSPIN の入力コードであるPromela をテキストエディタにて直接記述し、モデル検証を実施する。

③特定箇所モデリングによるアプローチ

典型的なシステムアーキテクチャの実現方式と、その実装過程で発生頻度の高い不具合パターンに焦点を当てて単純化したものをモデル検証によって考察し、その結果から検証手法の提案を行う。

(2) 確認された形式検証の効果

この結果、形式検証の効果として、以下が確認された。

①形式的記述を行う効果

形式検証のためには形式的記述を行うことが前提であるが、そのことが通常の設計時に起こる「もれ」、「ぬけ」、「あいまいさ」を減らし、品質向上への効果をもたらす。

②形式的検証を行う効果

システムのふるまいの確認等に対しては、従来手法に比べて網羅的、確実な検証ができる。なおツールの利用や特定箇所モデリングなどは、それぞれ一長一短を持っており、目的に応じた使い分けが必要と考えられる。

③組込みソフトウェアの検証に対する有効性

複数の並行動作単位の動作タイミングにかかわる性質や、周期にかかわる性質等、組込みソフトウェアにおいて重要性の高い性質の確認に有効である。

2 今後の展開

実際に開発現場に形式検証を適用するには、典型的な性質に対する検証技法の整備、様々な検証技術の体系化、環境の整備、教育、開発プロセスの整備などが必要となる。今後はこうした側面から実適用を目指した検討が重要となろう。

SEC 2006年度 活動概要 共同研究

国立情報学研究所（http://www.nii.ac.jp/）

ETSS向け教育研修コースを対象とした評価フレームワーク

国立情報学研究所 アーキテクチャ科学研究系
教授・主幹
本位田　真一

株式会社三菱総合研究所　情報技術研究センター
粂野　文洋

本研究は、組込みソフトウェアに関する教育プログラムに対し、受講価値の判断やプログラム設計・改善を支援する評価手法（以下、評価フレームワーク）の策定を目的としている。本研究では、特にハイレベルの技術者向けの教育プログラムに焦点を当て、評価フレームワークの策定を行う。

1 調査研究の方法

国立情報学研究所のトップエスイープロジェクト（以下、トップエスイー）を題材に、教育目標となるトップレベルのソフトウェア技術者像を明確化することを本研究の出発点とする。さらに国内外の企業・大学に対して、トップレベルのソフトウェア技術者像に関するインタビュー調査、ソフトウェアエンジニアリング教育に関する海外の先進事例を調査する。以上の調査を通して、トップエスイーへの評価を試行する。

2 本年度の成果

(1) 教育プログラムのゴール体系

教育プログラムの受講価値を明確化するためのフレーム（以下、ゴール体系）を策定した。ゴール体系は教育プログラムの目標・内容を、目標とする人材像、人材育成アプローチ、スキルマップ、知識体系、知識体系対応マップ、カリキュラムマップの6項目で表現する。ゴール体系で表現することで、教育プログラムの受講価値を明確化することが策定の狙いである。トップエスイーのゴール体系の一部（目標とする人材像と人材育成アプローチ）を図1に示す。

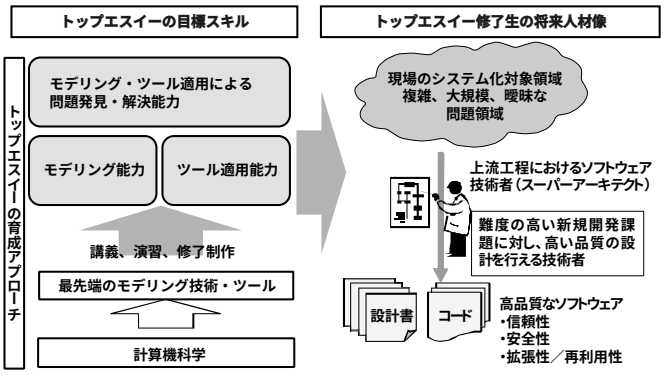


図1　トップエスイーのゴール体系

また、国内外の電気機器メーカ、ソフトウェア開発企業、大学（計10機関）に対し、トップレベルの技術者に求められる能力、トップエスイーに対する意見を伺う調査を実施した。求められる能力としては、基礎体力としての問題解決能力、上流工程における分析・モデリング能力、広範かつ専門的なドメイン知識、プロジェクト管理能力、コミュニケーション能力という声が多かった。トップエスイーに対しては、目標人材像には評価は高かったが、身に付けた知識や能力を企業に広める方策が必要との声も多く、課題も浮き彫りになった。

(2) 教育プログラムの実績の視覚化

教育プログラムの価値を明確化するには、ゴール体系だけでなく、その実績を明確にすることが必須と考えられる。そこで、トップエスイーの修了生12名に対し、ゴール体系の目標に対する達成度を評価した。評価は図1の3つの目標能力に関して、5人の講師で採点（0〜5点）し、その平均を算出することで行った。修了生12名の平均点は、ツール適用能力3.6、モデリング能力3.6、問題解決能力4.2という結果となった。

3 今後の展開

本年度に提示したゴール体系と実績評価が教育プログラムの価値判断に有効な情報となるかどうかの客観的な評価調査を行い、改善を進める予定である。また、目標能力が実地の現場で有効かどうかの追跡調査を行う必要がある。こうした調査研究を通して、トップエスイーの事例を一般化し、教育プログラムの価値判断を支援する評価手法の確立につなげていきたいと考えている。

SEC 2006年度 活動概要 共同研究

宮崎大学（http://www.miyazaki-u.ac.jp/）

品質モデル適応型テストプロセスの研究

宮崎大学工学部情報システム工学科
准教授
片山　徹郎

ソフトウェアは、IT産業のみならず製造業から金融業に至るまで、あらゆる産業の付加価値創出の源泉となっており、もはや現代経済社会の基盤として欠かせない存在となっている。とくに、日本が今後リードしていくべき組込みシステムの分野においては、ソフトウェアが製品の機能や競争力を左右する重要な役割を果たしており、その品質及び生産性の向上が不可欠である。その一方で、開発規模の増大と期間の短期化とが急速に進んでおり、体系的な対応策が強く求められている。

1 調査内容

このような背景のもと、テスト作業の重要性が日々増してきている。従来のテスト技法は、機能対応の、いわゆるバグ潰しの観点を主流としたものが多く、いかに効率的にプログラムの誤りを見つけるかについて、及びそれを実現するための個別手法の優劣について議論されてきた。しかしながら、近年の組込みソフトウェア量の増大と、それに伴う複雑化の結果、「バグゼロ」は実質無意味化しており、テストにおいては、新たな概念とその概念に基づいた技術や実用的な手法が求められている。すなわち、有限時間でその製品が持つ価値（安全、安心、便利、快適など）を損なわない、優先度を考慮した効率的なテストプロセス、さらにはソフトウェア特性に応じた手法のマッピングが必要とされている。

そこで本調査研究においては、機能安全の手法を品質モデルに適用することに資するため、現状のテスト関連技術の動向について調査し、テスト手法を分類した。また併せて、分類結果を適用可能なソフトウェア領域にマッピングをするための調査を実施した。

はじめに、テスト関連技術の調査を行った。この調査では、関連する研究論文や事例報告、及び記事について、国内／国際会議の予稿集や書籍、雑誌、Web等を調査の対象とし、調査結果より、テスト技術の技術整理を行った。テスト技術のまとめ方の方針は、基本的に、文献[MATSUMOTO2004]の「第5章ソフトウェアテスティング」の分類方針に従って分類した。ただし今回は、この章に記述されていない静的テスト技法やリスクベースドテスティングなどのテスト技法についても調査したうえで作成した。具体的には、ソフトウェアエンジニアリングの直感及び経験、仕様、コード、フォールト、利用、アプリケーション、のそれぞれに基づくテスト技法と、それらを組み合わせた技法、静的なテスト技法について、項目を設け分類した。

次に、技術整理の結果を踏まえて、テストレベル、及び、適用範囲のそれぞれの観点に基づき、テスト技法についてまとめなおした。ここで適用範囲の観点については、適用分野が多岐に渡りまとめることが困難となったため、今回は、組込みシステムのみを報告対象として取り上げ、まとめることとした。

最後に、報告されているテスト技術の実施テストレベルや実際の適用範囲を踏まえて、テスト技術の効果的な適用範囲について検討した。

2 今後の展開

今後の計画としては、今回実施したテスト技術の技術整理の結果を踏まえて、

① その技術を産業界に認知させ
② どのような対象に対して、どの技術を適用することが効果的であるか
③ またその際に、どのようなスキルが必要となるのか

を検討することが必要であり、検討した結果を踏まえて、実プロジェクトへの適用ガイドを作成することを考えている。

参考文献

[MATSUMOTO2004] 松本吉弘監訳：ソフトウェアエンジニアリング基礎知識体系-SWEBOK-，オーム社，2004

SEC 2006年度 活動概要 共同研究

名古屋大学組込みソフトウェア技術者人材養成プログラム NEXCESS
(http://www.nces.is.nagoya-u.ac.jp/NEXCESS/)

組込みソフトウェア教育の実施効果に関する調査

名古屋大学大学院情報科学研究科　研究員
山本　雅基

名古屋大学大学院情報科学研究科　教授
高田　広章

名古屋大学大学院情報科学研究科　教授
齋藤　洋典

名古屋大学情報連携基盤センター　教授
間瀬　健二

名古屋大学大学院工学研究科　准教授
河口　信夫

名古屋大学大学院情報科学研究科　准教授
冨山　宏之

名古屋大学大学院情報科学研究科　助教
本田　晋也

名古屋大学大学院情報科学研究科　研究員
金子　伸幸

経済産業省の組込みソフトウェア産業実態調査（2006年実施）によると、技術者の約70%は、自己のスキル向上に教育セミナの受講が有効な手段であると考えている。しかし、経営者の40%は技術者に対して年間1週間未満の教育時間しか与えておらず、技術者の約55%はスキル向上のための時間が取れないと報告している。この調査結果は、社会人の組込みソフトウェア技術者は教育に対して高い期待を有するが、十分な教育時間が与えられていない実態を示している。

このような背景の中で、我々は、教育の効果を可視化し組込みソフトウェア技術者教育の強化と普及へ貢献することを目的として、組込みソフトウェア教育の実施効果計測に関する研究をSECと共同で行っている。2006年度は、NEXCESSの初級技術者教育コースの受講者に対して教育効果の計測（研究1）と、企業で行われている教育効果計測の実態調査（研究2）を行った。

1 教育効果計測の成果

先行研究[KIRKPATRICKJ2006]では、目的別に以下の4段階に分けた受講者の評価の実施を推奨している。

・Reaction評価 ：研修時の受講者の反応評価
・Learning評価 ：獲得した知識と技能の評価
・Behavior評価 ：研修後の職場における評価
・Result評価　 ：会社事業に対する貢献の評価

社会人教育では、経営者は、教育の実施に要した投資と、教育の結果得られた利益に関する次の評価を要求する。

・ROI評価：教育に対する投資対効果の評価

研究1では、2006年度に開講した2回のNEXCESS初級コース「組込みソフトウェア開発技術の基礎」の受講者60名に対して、Reaction、Learning、Behavior評価を実施した。

調査の結果、Reaction評価が高い受講者は、低い受講者に比べてLearning評価が高くなる傾向を示すことが確認された。このことは、教育を通じて受講者の知識と技能を高めるためには、教育期間中の受講者の取り組み姿勢を高める指導方法が効果的であることを示す。一方、受講前後の筆記テストにおいて、受講前の点数が低くかつ受講後の点数が高い受講者は、業務に戻った後に職業性ストレスが増える傾向があることが確認された。このことは、教育受講後の職場における指導の必要性を示す。

表1　企業で行われている教育効果計測の実態

評価種別	実施率(%)
Reaction	100
Learning	77
Behavior	4
Result	4
ROI	0

研究２として、4社の計26種の教育コースに対して教育効果計測の実態を、筆記アンケートと面談を実施した。調査項目は、Reaction、Learning、Behavior、Result、ROI評価の実施状況であり、表1はその結果を示す。

調査結果は、評価種別の順に実施率が低くなることを示す。Reaction評価は100%実施されており、講師の教え方の改善や教材の改訂などに利用されている。Learning評価は、受講者の修了判定等を目的として、77%のコースで実施されている。Behavior及びResult評価は、受講者の上司が自ら教育講師を行った1つのコースだけで行われていた。ROI評価は、調査した4社26コースでは行われておらず、また、経営者からの明確な評価要求も出されていなかった。

2 今後の展開

社会人教育では、学校教育とは異なり、受講者の上司が存在することが特徴的である。今後、上司が受講者である部下の教育効果をいかに評価するかを含め、上司と部下の双方的な教育評価の仕組みづくりに取り組む。

参考文献

[KIRKPATRICJ2006] Kirkpatrick D.L.,& Kirkpatrick J.D.Evaluating Training Programs. CA:Berrett-Koehler Publishers, 2006

SEC 2006年度 活動概要 共同研究

早稲田大学（http://www.waseda.jp/）

コンポーネントベース高信頼性実時間組込みシステム構築技術の研究

早稲田大学理工学術院
教授
中島　達夫

早稲田大学理工学術院
助手
菅谷　みどり

情報アプライアンスは単体で機能するのではなく、組み合わせることにより様々な新しいサービスを作り出すことを可能とする。情報アプライアンスに用いられる組込みシステムは従来のエンタープライズシステムとは異なり、様々なリソースの制約（システム構築時に実時間動作実現等）を考慮する必要があるため、従来の手法とは異なる新しい技術の開発が必要となる。

我々の開発したアカウンティングシステムと仮想周期実行システムは、従来のLinuxでは、実装が困難であったリアルタイム処理のプログラムを容易にすることができた。

1 成果概要

(1) 組込みLinux上のアカウンティングシステム

アカウンティングシステムは、各アプリケーションが利用するCPUリソースの使用量を制限することを可能とする機能で、周期と実行時間の2つのパラメータを持つアカウンティングオブジェクトという抽象オブジェクトを提供する。アカウンティングオブジェクトをバインドされたプロセス群は、それらが周期ごとに決められた実行時間のみを消費することを保証する。

アカウンティングシステムの応用としては、ダウンロードしたプログラムが規定した以上のCPUキャパシティを用いないようにすること、リアルタイムクラスと非リアルタイムクラスのプログラムを柔軟に調停すること、システム全体のオーバロードを検出すること、リアルタイムプログラムのためのCPUリソースの予約等が考えられる。

アカウンティングシステムは、極力Linuxカーネルを変更しなくても使用できるようにデザインされている。そのため、Linuxカーネルは頻繁に変更されるにもかかわらず、アカウンティングシステムの維持を容易に行うことが可能となる。

(2) 仮想周期実行システム

仮想周期実行システムは、アカウンティングシステム上に実現され、長時間実行される実時間スレッドが高い応答性を必要とするユーザインタフェース等の非リアルタイムスレッドに与える影響を最小限とするために用いられる。現状の組込みシステムでは、既存のリアルタイムアプリケーションをLinux上に移植したものも多く、その場合には、非リアルタイムアプリケーションの応答性が大きく低下する可能性がある。仮想周期実行システムは、実時間スレッドの周期を強制的に短くすることにより、以上の問題点を解決する。つまり、長時間実行される実時間スレッドを強制的に短い周期でブロックすることで、非リアルタイムアプリケーションがスケジューリングされる可能性を増大させる。

2 まとめ

我々が開発したアカウンティングシステムは、従来のLinux上での高信頼性実時間動作が認められ、日本エンベデッドリナックスコンソーシアムのリソースマネジメントワーキンググループにおける標準化のリファレンス実装として用いられている。また、CE Linuxフォーラムでも実際の家電機器における使用に関して検討が進められている。

参考文献

[SUGAYA2006] Midori Sugaya, Shuichi Oikawa, Tatsuo Nakajima: Accounting System: A Fine-Grained CPU Resource Protection Mechanism for Embedded System: Ninth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2006), 72-84

SEC 2006年度 活動概要 共同研究

慶應義塾大学（http://www.keio.ac.jp/）

性能指向組込みソフトウェアの研究：並列化、ハードウェア化を目的としたC言語記述に関する記述作法

慶應義塾大学理工学部
教授
天野　英晴

組込み機器の多機能化により、組込みシステムには従来に比べて桁違いに大きな処理能力が要求されるようになった。このような処理能力を低電力、低コストで実現するため、単純なRISCプロセッサに、特定の機能のみ高速に実行するハードウェアのアクセラレータを接続して協調処理を行わせるシステムLSIあるいはSoCが一般化した。この設計を短時間で行い、生産性を向上させるため、近年、System-C、Spec-C等のシステム記述を指向したC言語が登場し、Handel-C、Bach-C、BDL等のハードウェア記述用C言語等が普及している。これらのハードウェア記述用C言語は、ANSI-Cに近い文法を持っていることから、組込み用のアプリケーションプログラムを簡単に移植することができ、複雑かつ高性能の組込み機器が簡単に設計可能となるはずだった。

ところが、ハードウェアや新しいアーキテクチャにオフロードされるべき組込み用アプリケーションの多くは、動的なメモリ割り付けや、ポインタを利用した関数、変数の共有等のソフトウェア的な技法を駆使して記述されており、これをハードウェア記述用C言語に変換することには非常な困難を伴った。「組込みソフトウェア開発向けコーディング作法ガイド［C言語版］」[ESCR]は、この絶望的な状況から抜け出すための大いなる希望である。この記述作法は「ハードウェア化しやすい記述」と重なることが多く、実用的な組込みプログラム技術者が、この作法を守ってアプリケーションを書いてくれれば、ハードウェア記述Cへの移植が従来に比べて格段に楽になる。本研究では、この考え方を一歩進め、ハードウェア化が容易なプログラミング作法、Hardware Concious Style（ハードコンシャス記述）を提案する。

ハードコンシャス記述はアプリケーションプログラマとハードウェア設計者のインタフェースとして利用されることが期待されている。図1にこの様子を示す。

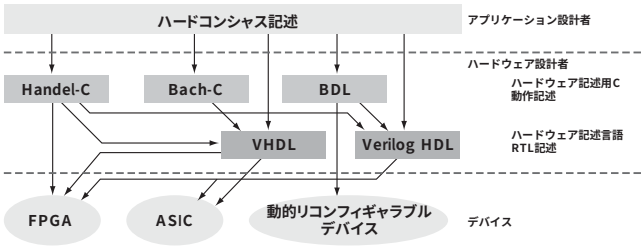


図1　ハードコンシャス記述の位置付け

1 ハードコンシャス記述の方法

ハードコンシャス記述では以下の3点について特に注意する。

①変数のサイズ、符号を常に意識する
②メモリに対する意識を変える
③すべての資源は静的に決っており、動的な生成、変更は極力行わない

本研究は、上記の方針で書き方をまとめ、さらに「組込みソフトウェア開発向けコーディング作法ガイド［C言語版］」[ESCR]に対する補足の形で詳細を示した。これに関して詳しくは、本誌「技術解説」を参照されたい。

2 有効性の確認と今後の課題

ハードコンシャス記述の有効性を確認するため、組込み用アプリケーションベンチマーク集MiBench[MIBENCH]から3つのプログラムを選んでハードコンシャス記述に書き直し、さらにその記述からHandel-C、Bach-Cに変換した。それぞれの主な修正点、修正時間の評価を行った結果、ハードコンシャス記述自体ができれば、それぞれ数時間、特定の部分のみ気をつけることで、ハードウェア記述Cへの変換ができることが明らかになった。

参考文献

[BACH2004] Bach-C言語マニュアル，シャープ株式会社，2004
[ESCR] IPA SEC：組込みソフトウェア開発向けコーディング作法ガイド［C言語版］，翔泳社，2006
[HANDEL2005] Handel-C Language Reference Manual，Celoxa，2005
[MIBENCH] http://www.eecs.umich.edu/mibench/
[WAKABAYASHI] 若林一敏：LSI 開発はこう変わるC言語設計の最前線（1），http://techon.nikkeibp.co.jp/NEWS/dac2003/030317 1.html

工期の厳しさに関連する要因の分析

奈良先端科学技術大学院大学
門田　暁人

株式会社日立製作所
馬嶋　宏

NECソフト株式会社
増田　浩

沖電気工業株式会社
羽田野　尚登

株式会社アルゴ21
磯野　聖

SEC研究員
内海　昭

SEC研究員
菊地　奈穂美

株式会社NTTデータ
服部　昇

三菱電機インフォメーションシステムズ株式会社
細谷　和伸

株式会社日立システムアンドサービス
森　和美

本稿では、「ソフトウェア開発データ白書2006」[SEC2006]記載のプロジェクトのうち大規模（1,000FP以上）のものを対象とし、開発期間（工期）の厳しさとプロジェクト特性の関係を統計的に分析した。その結果、開発計画時に決まる特性のうち、工期の厳しさに影響を与えるものは、開発規模、主開発言語、アーキテクチャであった。これらの要因については、(1) 規模が大きいほど工期が厳しい、(2) COBOLはVBより工期が厳しい、(3) 2層C／Sは他のアーキテクチャより工期が厳しくない、ことが確認できた。また、開発の進行に従って決まる特性のうち、工期の厳しさから影響を受けるものは、平均要員数、生産性、工数であった。これらの要因については、(1) 工期が厳しいプロジェクトほど平均要員数が多い、(2) 工期の厳しいプロジェクトでは高い生産性を達成したケースが存在しない、(3) 工期の厳しいプロジェクトでは著しく大きな工数がかかる場合がある、ことが確認できた。

1. はじめに

本稿は、2006年度IPA／SECエンタプライズ系タスクフォース定量データ分析部会WG3において、工期の厳しさとトレードオフの関係にある要因の調査を行うという課題に対し、著者らの委員が共同で分析と検討に取り組んだ成果をまとめたものである。

ソフトウェア開発において、開発すべき成果物の規模が大きいほど、開発に必要な期間（工期）も一般に長期化する。一方で、ビジネス上の都合など、開発を取り巻く環境によっては、短期間で大規模開発を行う、いわゆる「工期の厳しい」プロジェクトが立案される場合がある。しかし、従来、どういった要因が工期の厳しさに影響を与え、また影響を受けるかについて、定量的な分析はほとんど行われていない。

そこで、本稿では、工期の厳しさに関連する要因について、SEC収集データ[SEC2006]を用いて、統計的分析を行った結果を報告する。

2. 分析対象データ

分析対象となるプロジェクトは、「SEC収集データ」に記載の1,419件のうち、次の条件を満たす91件である。

- 新規開発プロジェクトである。
- 開発5工程の工数が計測されている（欠損値を含まない）。開発5工程とは、基本設計、詳細設計、製作、結合テスト、総合テスト（ベンダ確認）の各工程を指す。
- FP計測手法が明確である。
- 規模が1,000FP以上である。
- FP、工期、工数に欠損値を含まない。

分析対象を、規模の大きなプロジェクト（1,000FP以上）に限定した理由は、規模の小さなプロジェクトほど不確定要素が多く、生産性にも大きなばらつきが見られ[SEC2006]、一般性の高い結果を得ることが難しいと想定されたため、また、規模の大きなプロジェクトについての知見を得ることが、多くの企業にとってより重要であると考えたためである。

3. 開発速度に基づく予備分析

工期の厳しさについての最もシンプルな捉え方は1ヶ月

当たりの開発規模（FP）が大きいほど工期が厳しいとみなすことである。予備分析では、この考え方に基づいて次式のように定義する。

工期の厳しさ＝FP÷工期（月数）

この指標は、speed of deliveryとも呼ばれ、開発速度を示す尺度として従来用いられている[ISBSG]。値が大きいほど、開発スピードが速い反面、工期は短い（厳しい）といえる。

分析対象プロジェクト（91件）を開発規模（FP）の大きさに応じて4等分し、工期の厳しさ（開発速度）との関係を箱ひげ図により分析した（図1）。図中、箱の上端は（値が大きい方から数えて）上位25％に相当するデータの位置を示し、下端は下位25％の位置を示す。箱の中の横線は中央値である。また、箱から出ている「ひげ」の上端は最大値を、下端は最小値を表す。ただし、外れ値とみなせるデータは、ひげの外側の丸印や＊印として表される。詳しくは、文献[SEC2006]の157ページを参照していただきたい。

図1に示すように、規模と開発速度は強い関係があり、規模が大きくなるほど工期が厳しくなる。すなわち、システム全体の規模が大きくなるほど1ヶ月当たりの開発規模も大きくなる傾向が見られた※1。特に、規模の大きな上位25％のプロジェクト（2,885FP以上）では、著しく工期の厳しいケースが数多く見られた。

この結果は、開発規模が倍になっても工期は倍にはならず、短めに設定される傾向にあることを示している。また、顧客の要求する開発規模には上限がないが、工期には上限があることも示唆している。

現実的な観点からは、開発期間が規模に比例して大きくなるという前提で工期の厳しさについて議論するのではなく、開発期間は規模が大きくなるほど（規模に比べて）短めに設定されるという前提でさらなる分析を行うことが望ましいといえる。

4. 適正工期に基づく分析

4.1 工期の厳しさの定義

前章での分析からもわかるように、現実的な工期（適正工期と呼ぶ）は、開発規模に比例して決まるわけではない[PUTNUM2005]。Boehm[BOEHM1981] によると、適正工期は、開発工数に応じて決まり、開発工数の概ね3乗根に比例することが知られている。このことは、Putnumなど他の研究者によっても確認されている[PUTNUM1978]。

そこで、図2に示すように、Boehmによる適正工期の

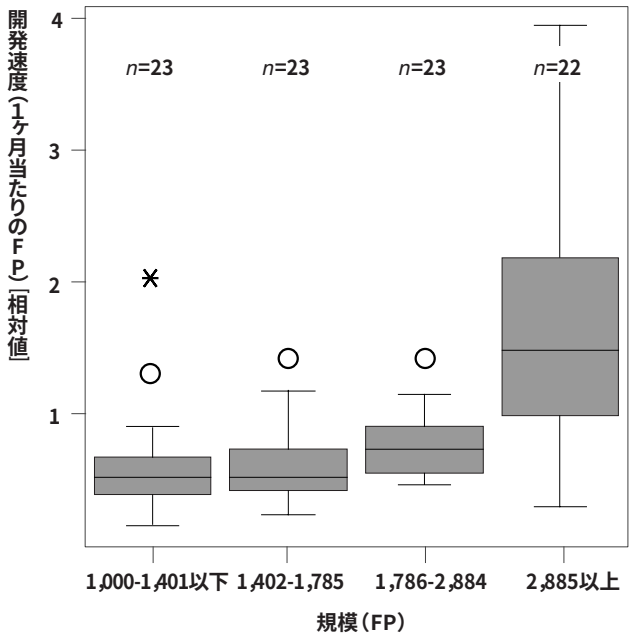


図1　開発速度（FP÷工期）と規模の関連

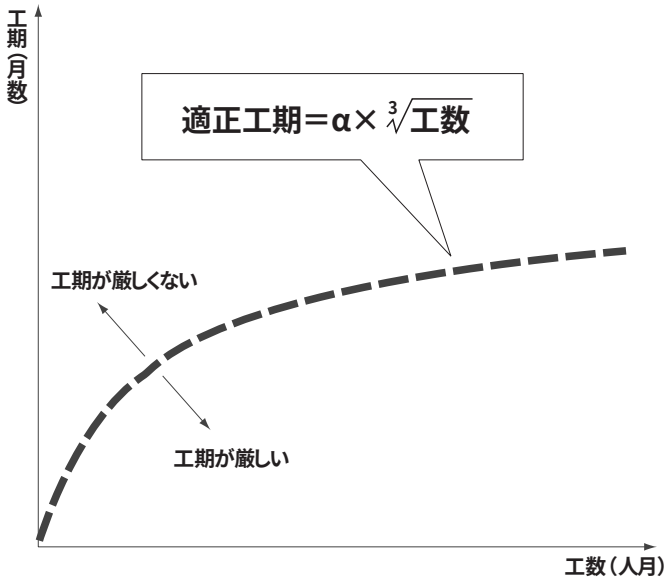


図2　適正工期による工期の厳しさの捉え方

※1 開発速度（FP÷工期）の定義の分子に規模（FP）が含まれるため、開発速度と規模との間に何らかの関係があることは一見自明であるが、規模が開発速度に対してプラスに働くのかマイナスに働くのか、また、線形に働くのか非線形に働くのかは、式だけからでは明らかでなく、定量的な分析が必要である。

式をもとにして想定される適正工期よりも短期の開発の場合に工期が厳しいとみなすことにする。この場合の工期の厳しさは、適正工期に対する実工期の比として表すことができ、次のように定義できる。

$$\text{適正工期} = \alpha \times \sqrt[3]{\text{工数}}$$

$$\text{工期の厳しさ} = \text{工期} \div \text{適正工期}$$

$$= \text{工期} \div (\alpha \times \sqrt[3]{\text{工数}})$$

$$= \text{工期} \div \sqrt[3]{\text{工数}} \quad \cdots \alpha = 1 \text{とした場合}$$

この定義では、値が小さいほど工期が厳しいことを意味する。αは未知数であるが、本稿では便宜上α＝1と定める。このように定めたとしても、工期の厳しさ（の大小）に影響する要因の分析においては問題とはならない。

4.2 分析方法

(1) プロジェクト特性の選定

分析対象としたプロジェクト特性を表1に示す。表1(a)は、開発計画時（もしくは、開発の初期段階）に決まるプロジェクト特性を示しており、工期の厳しさに影響を与える要因の候補である。一方、表1（b）は、開発が進むに従って決まる（もしくは、開発終了時に確定する）プロジェクト特性を示しており、工期の厳しさから影響を受ける要因の候補である。これら以外にもプロジェクト特性は多数あるが、本稿では、次の条件により分析対象とするプロジェクト特性及び名義尺度についてはカテゴリを選定した。

1a2つ以上のカテゴリで10件以上のデータのある特性

例：「業務パッケージの利用」という特性は、「有り」カテゴリが8件、「無し」カテゴリが70件なので、分析対象としなかった。

1bデータ数の少ない（5件以下）カテゴリは分析から除外する、もしくは、「その他」カテゴリに含める。

例：主開発言語＝Perl、Pro*Cなどは5件以下なので「その他の言語」に含めた。また、スタンドアロンとメインフレームは2件ずつしかないので分析から除外した。

1b平均要員数は、総開発工数÷工期により求めた値を用いた。

表1　分析対象のプロジェクト特性

(a) 開発計画時に決まるプロジェクト特性

プロジェクト特性	尺度	説　明
業種	名義	製造、卸売、金融、サービス、その他
主開発言語	名義	COBOL、VB、Java、PL／SQL、Developer 2000、その他
アーキテクチャ	名義	2層クライアントサーバ（C／S）、3層C／S、イントラネット／インターネット
要求仕様の明確さ	順序	かなり明確、やや曖昧、非常に曖昧
FP（見積値）	比例	ファンクションポイント

(b) 開発が進むに従って決まる特性

プロジェクト特性	尺度	説　明
工数	比例	開発総工数実績値（人月）
不具合密度	比例	1FP当たりの発生不具合密度
生産性	比例	FP÷工数
平均要員数	比例	工数÷工期

表2　分散分析の結果

(a) 開発計画時に決まるプロジェクト特性

プロジェクト特性	有意確率p	寄与率
主開発言語	0.004	15.5%
アーキテクチャ	0.022	6.6%
要求仕様の明確さ	0.238	2.3%
業種	0.218	2.2%
FP	0.889	0

(b) 開発が進むに従って決まる特性

プロジェクト特性	有意確率p	寄与率
平均要員数	0.000	29.2%
生産性	0.001	15.4%
工数	0.021	7.5%
不具合発生密度	0.401	0

②分析の手順

表1（a）の各特性が工期の厳しさに与える影響を分析するにあたって、一元配置分散分析を用いて、有意確率pと寄与率（調整済み寄与率ω^2）を導出した※2。ただし、表1（a）の特性のうちFPは比例尺度であり、そのままでは分散分析が行えないため、名義尺度に変換してから分析を行った。具体的には、値の大きさに応じて4つのカテゴリ（下位25%、下位25～50%、上位25～50%、上位25%）に変換した。この変換により、表1（a）の他の特性と同様に寄与率を求めることが可能となる。

一方、表1（b）の各特性が工期の厳しさから受ける影響を分析するにあたっては、工期の厳しさを名義尺度に変換（上記と同様に4つのカテゴリに分割）し、比例尺度である各特性との関係を分散分析により分析した。

4.3 分析結果

（1）工期の厳しさに影響を与える要因

分散分析の結果を表2に示す。表2（a）は開発計画時に決まるプロジェクト特性についての結果である。工期の厳しさと有意な関係（$p<0.05$）の見られた特性は、主開発言語（寄与率15.5%）とアーキテクチャ（寄与率6.6%）の2つである。残りの3つ（要求仕様の明確さ、業種、FP）については、有意な関係は見られなかった。ただし、FPについては、第3章で示したように開発速度とは強い関係があり、FPが大きいほど開発速度が速くなる（1ヶ月当たりの開発規模が増える）点に留意されたい。

これらの特性のうち有意な関係が見られたものについて、箱ひげ図を用いた詳細な分析を次に示す。

①主開発言語

図3に示されるように、主開発言語がCOBOLの場合に、工期が厳しい傾向にあった。特に、VBと比べて明確な差が見られた。1つの解釈として、COBOLとVBではシステムの性質が異なり、COBOLのシステムのほうが品質に対する要求がより厳しいために工期が厳しくなったのではないかと推測される。

②アーキテクチャ

図4に示されるように、アーキテクチャが2層C／Sの場合に、工期が厳しくない傾向にあった。さらなる分析の結果、2層C／Sのプロジェクトは、古い年代（1998～2000年）に開発が終了したケースが多く含まれていたのに対し、3層C／Sは2001年以降、イントラネット／インターネットは2002年以降のケースしか含まれていなかった（ただし、開発終了年が欠損値のものも存在する）。1つの解釈として、古い年代ほど、業界全体として工期の厳しくないプロジェクトが多かった可能性がある。今後、開発年と工期の厳しさの関係について、より詳しい調査が望まれる。

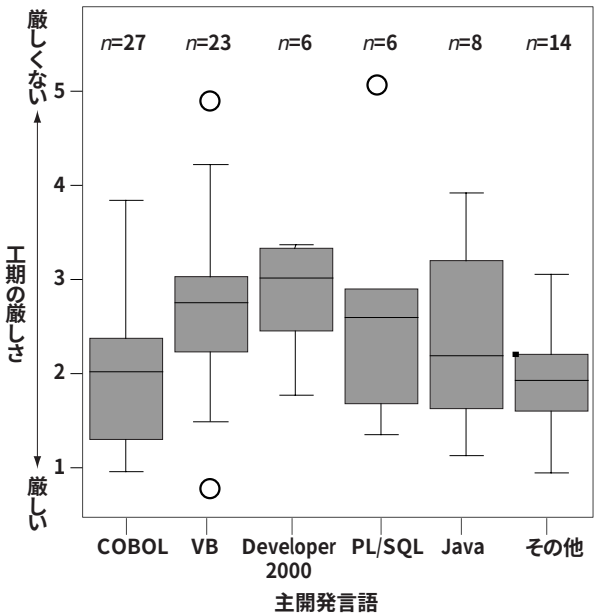


図3　主開発言語と工期の厳しさの関係

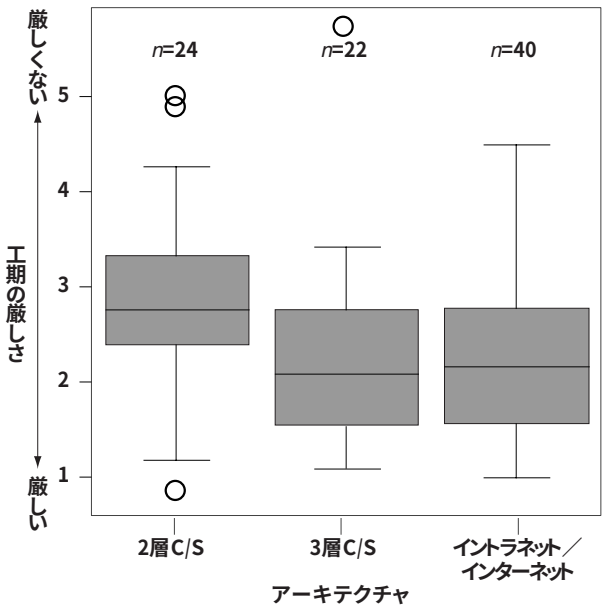


図4　アーキテクチャと工期の厳しさの関係

※2 分散分析の適用は、目的変数が正規分布に従うことが前提となるが、本稿では、簡便のため、変数の正規性を考慮せずに分散分析を行っている。そのため、得られた有意確率pは誤差を含む点に留意願いたい。ただし、本稿では1,000FP以上のプロジェクトを分析対象としているため、値の分布が極端に小さな方に偏っていることはなく、結論が変わるほどの誤差は含まないと考える。

(2) 工期の厳しさから影響を受ける要因

開発が進むに従って決まるプロジェクト特性についての分散分析の結果を表2（b）に示す。工期の厳しさと有意な関係（$p<0.05$）の見られた特性は、平均要員数（寄与率29.2％）、生産性（寄与率15.4％）、工数（寄与率7.5％）の3つである。不具合発生密度については、欠損値が多いこともあり、有意な関係が見られなかった。有意な関係が見られたものについて、箱ひげ図を用いた詳細な分析を次に示す。

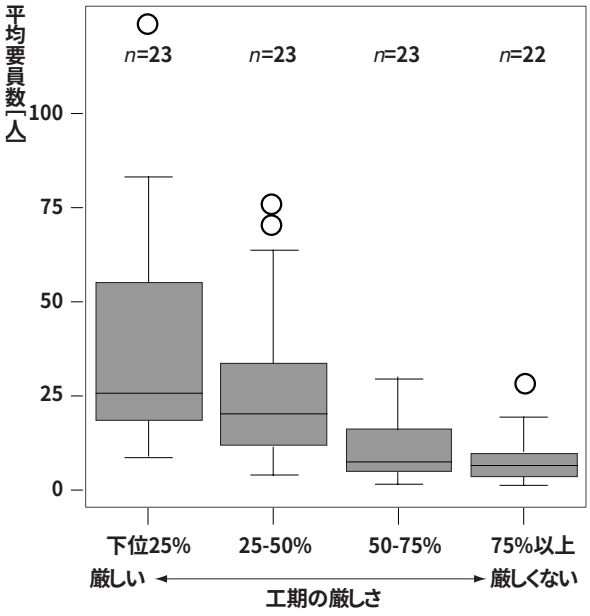


図5　平均要員数と工期の厳しさの関係

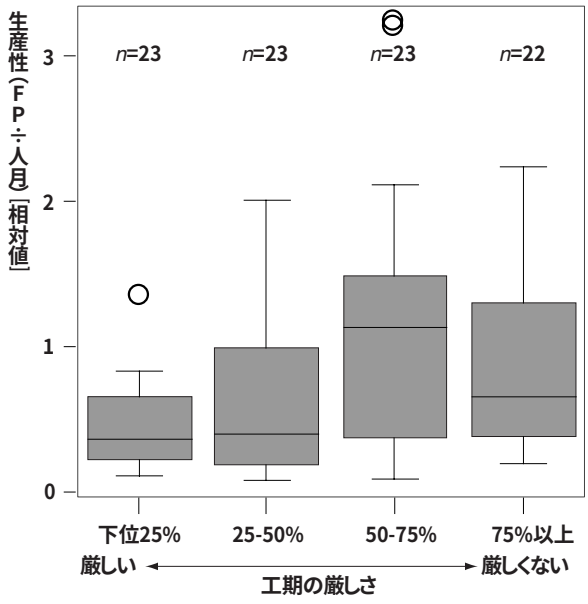


図6　生産性と工期の厳しさの関係

①平均要員数

平均要員数は、工期の厳しさと最も関連の強いプロジェクト特性であり、その寄与率は29.2％と高かった。図5に示されるように、工期の厳しいプロジェクトほど、平均要員数のばらつき、中央値ともに大きく、50人を超えるようなケースも多くみられた[*3]。

工期が厳しいプロジェクトほど、より多くの開発要員を割り当てて納期を守ろうとしている様子がうかがえる。一般に、平均要員数が著しく多いと、プロジェクト失敗のリスクも高まるため、極端な工期の厳しさはプロジェクト失敗のリスクに直結するといえる。

②生産性

図6に示されるように、工期の厳しいプロジェクト群（左端の箱ひげ図）の中には、高い生産性を達成したケースは含まれていなかった。高い生産性を達成するためには、工期に無理がないことが前提となるといえる。

③工数

図7に示されるように、工期の厳しくないプロジェクト群（右端の箱ひげ図）では、大きな工数を要したプロジェクトがほとんど見られなかった。工期の厳しいプロジェクトでは、著しく大きな工数がかかる場合があるといえる。

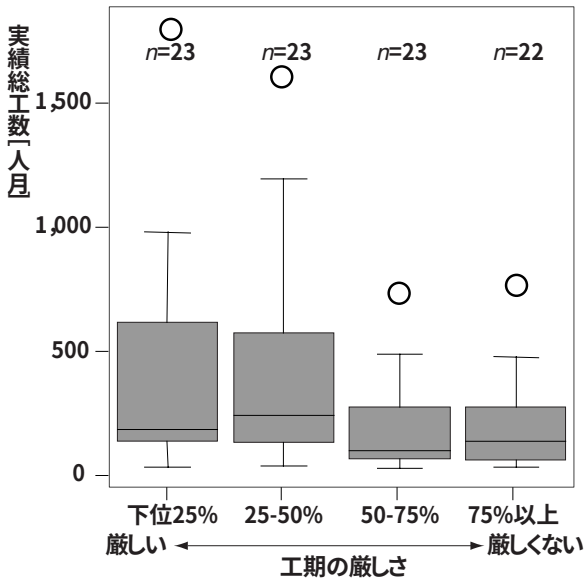


図7　工数と工期の厳しさの関係

※3 工期の厳しさ（工期÷工数の3 乗根）と平均要員数（工数÷工期）との関係は、式の定義から一見自明であるが、複数のプロジェクトを比較する場合においては、それぞれ開発規模が異なるため、平均要員数が大きいプロジェクトほど必ず工期が厳しい等の関係は自明ではない（紙面の都合上説明は省略するが、図5 の4 つの箱に重なりがあることからも分かる）。そのため、本稿のような定量的分析には意義がある。

4.4 考　察

プロジェクトの計画立案時に工期を決定する場合、前節で得られた結果について、ユーザ企業とベンダ企業の双方が共通認識を持つことが重要であると考える。例えば、計画として工期を標準（適正工期）よりも短縮する場合には、開発要員一人当たりの生産性が低く抑えられる（図6）。そのため、1 FP当たりの開発要員をより多く投入する必要があり、標準的な工期で開発した場合よりも、トータルではより多くの工数を要することとなる。計画立案時には、このトレードオフを理解した上で、短工期を要件に含めるかどうかを検討する必要がある。

一方、工期を極度に長くし、非常に緩やかな工期で開発することで工数の削減が見込めるようにも思われるが、図6の結果から、工期が緩やか過ぎても生産性向上の効果は見られない。そのため、短工期を要件としない開発においても、適正工期を見極めることが重要であるといえる。

5. まとめ

本稿では、「ソフトウェア開発データ白書2006」［SEC2006］記載のプロジェクトのうち大規模（1,000FP以上）のものを対象とし、工期の厳しさに影響を与える要因、及び工期の厳しさから影響を受ける要因のそれぞれを統計的に分析した。

1ヶ月当たりの開発規模（FP）が大きいほど工期が厳しいとみなす予備分析では、規模が大きくなるほど工期が指数的に厳しくなる傾向が見られた。特に、規模の大きな上位25％のプロジェクト（2,885FP以上）では、著しく工期の厳しいケースが数多く見られた。

次に、現実的な工期（適正工期）が開発工数の概ね3乗根に比例することを前提とし、想定される適正工期よりも短期の開発の場合に工期が厳しいとみなして、より詳細な要因分析を行った。分析の結果、工期の厳しさに影響を与える要因として、主開発言語とアーキテクチャが有意であり、COBOLはVBより工期が厳しいこと、2層C／Sは他のアーキテクチャより工期が厳しくないことなどが示された。また、工期の厳しさから影響を受ける要因として、平均要員数、生産性、工数が有意であり、工期が厳しいプロジェクトほど平均要員数が多い、工期の厳しいプロジェクトでは高い生産性を達成したケースが存在しない、工期の厳しいプロジェクトでは、著しく大きな工数がかかる場合があることなどが示された。

なお、本稿の分析対象プロジェクトは91件とやや少ないため、結果の一般性を高めるためには、今後、より多くのプロジェクトを用いた分析を行う必要がある。また、いくつかの要因（例えば、平均要員数と生産性）は独立ではないため、複数の要因間の関係を考慮した分析を行うことが今後の課題となる。

参考文献

[BOEHM1981] W. Boehm： Software engineering economics, Prentice-Hall, 1981

[ISBSG] ISBSG Information：Speed of Delivery, http://www.isbsg.org/isbsg.nsf/weben/Speed％20of％20Delivery

[PUTNUM1978] L.H. Putnum：A general empirical solution to the macro software sizing and estimation problem, IEEE Transactions on Software Engineering, Vol. SE-4, pp. 345-361, July 1978

[PUTNUM2005] L.H. Putnum and Ware Myers：初めて学ぶソフトウエアメトリクス，日経BP社，2005

[SEC2006] IPA SEC：ソフトウェア開発データ白書2006，日経BP社，2006

ハードコンシャス記述C：ハードウェア化を目的としたC言語記述作法

慶應義塾大学理工学部
教授
天野　英晴

音声、画像等のメディア処理、ネットワーク処理、暗号化、複号化等、高速処理を要求されるアプリケーションを、ハードウェアにオフロードすることで、優れた性能、コスト、消費電力を実現するシステムLSIが普及している。しかし、多くのアプリケーションプログラムは、動的メモリ割り付けやポインタ等、高度なソフトウェアの技法を駆使して記述されており、ハードウェア化することに多大な労力を要する。ハードコンシャス記述Cは、ハードウェア化を念頭においたアルゴリズムの記述作法であり、ANSI-Cの記述に一定の制限を設けると共にプログラマの意識を変えてもらうことで、ハードウェア化しやすい記述をしてもらうことが目的である。本稿では、このハードコンシャス記述Cの概観を紹介する。

1. はじめに

従来、組込みシステムで用いられるプロセッサは、消費電力の小さい単純なRISC型のマイクロプロセッサであった。しかし、組込み機器の多機能化により、音声、画像等のマルチメディア処理機能、ネットワーク接続機能、暗号化・復号化機能等、従来に比べて桁違いに大きな処理能力が要求されるようになった。このような処理能力を満足する高性能マイクロプロセッサは、膨大な電力を消費するため、通常は組込み機器に利用することはできない。このため、単純なRISCプロセッサに特定の機能のみ高速に実行するハードウェアのアクセラレータを接続して協調処理を行わせるシステムLSIあるいはSoC（System-on-a-Chip）が一般化した。さらにハードウェアに対するオフロードだけではなく、マルチプロセッサ化、コンフィギャラブル、リコンフィギャラブル、動的リコンフィギャラブル等、特定分野での高い性能を柔軟かつ低コストで実現するための新しいアーキテクチャが次々に登場している。

このような新しい高速化用アーキテクチャを用いた高速化設計を短時間で行い、生産性を向上させるため、System-C、Spec-C等のシステム記述を指向したC言語が登場し、Handel-C[HANDEL2005]、Bach-C[BACH2004]、BDL[WAKABAYASHI]等のハードウェア記述用のC言語が普及している。これらのハードウェア記述用C言語は、ANSI-Cに近い文法を持っていることから、組込み用のアプリケーションプログラムを簡単に移植することができ、高い機能合成能力と優れたデザイン環境を利用して、複雑かつ高性能の組込み機器が簡単に設計可能となるはずだった。

ところが、ハードウェアや新しいアーキテクチャにオフロードされるべき組込み用アプリケーションの多くは、動的なメモリ割り付けや、ポインタを利用した関数、変数の共有等のソフトウェア的な技法を駆使して記述されている。さらに、最近のC++やJAVAの利用はこれに拍車を掛け、プログラムを実行しないと変数の型やサイズが決まらない抽象化された記述が美しいプログラムとしてもてはやされる傾向にある。このため、ハードウェア設計者は、シンタックスの共有部分がかなりあるにも関わらず、これらの美しく記述されたアプリケーションをハードウェア記述用C 言語にそのまま移植することをあきらめ、七転八倒の末、元々のアルゴリズムとデータ構造、必要なメモリのサイズやレジスタのビット数等を理解して、一から書き直さざるを得ない場合がほとんどである。これにより、高性能組込み機器の生産性はそのうたい文句からは掛け離れた状態にある。

「組込みソフトウェア開発向けコーディング作法ガイド［C言語版］」[ESCR]は、この絶望的な状況から抜け出すための大いなる希望である。この記述作法は、「ハードウェア化しやすい記述」と重なることが多く、実用的な組込みプログラム技術者がこの作法を守ってアプリケーションを書けば、ハードウェア記述Cへの移植が従来に比べて格段に楽になる。この考え方を一歩進め、ハード

ウェア化が容易なプログラミング作法をまとめたのが、Hardware Concious Style（ハードコンシャス記述）である。本稿ではこのハードコンシャス記述の概略を紹介する。

ハードコンシャス記述は、ハードウェア化を意識したANSI-Cの書き方の作法であり、新しいハードウェア記述Cを提案しているのではない。しかし、ハードコンシャス記述で書かれたコードは、様々なハードウェア記述言語にほとんど直接的に変換可能である。また、ハードコンシャスの記述を直接変換してできたハードウェア記述Cで、優れたハードウェアがそのまますぐに生成されるわけではない。しかし、この記述は機能合成ツールの能力を最大限に生かすと共に、設計者にとって設計上のトレードオフの検討を容易にし、デバイスと要求仕様に合わせた優れたハードウェアを最終的に設計するために大いに助けとなるはずである。さらに、ハードコンシャス記述は、アルゴリズム設計者に無理にハードウェアを意識してアルゴリズムを書いて欲しいと要求しているのではなく、無理にプログラム格納型計算機に合わせてアルゴリズムを変えず、自然に記述することをお願いしている。

１つ問題なのは、この作法を守ることによって、そのプログラムは、一般のコンピュータプログラマから見ると稚拙なものに感じられるものになってしまう点である。ハードコンシャス記述という、耳慣れない名前を付けたのは、記述者がプログラムの技量不足のため、記憶領域の動的確保やポインタによるリスト構造が使えないのではなく、ハードウェア設計者に配慮して、作法に従って使わなかったことを明確にしようと考えたためである。我々ハードウェア設計者は、ハードコンシャス記述に則ってアルゴリズムを書いてくれたプログラマに対しては、技量不足等と思うことは決してなく、心の底からの感謝と尊敬の念を持つであろう。

2. ハードウェアを意識するということ

2.1 ３つの原則

プログラミング言語自体、プログラム格納型の計算機上での実行を意識して作られたものである。このためプログラムの記述の隅々までプログラム格納型モデルが染み付いており、これをすべてクリアすることは思いもよらない。ハードコンシャス記述では、以下の3点のみ注意していただくよう御願いする。

・変数のサイズ、符号を常に意識し、管理する。これらはハードウェアのコストに直結する。
・単一メモリではなく、複数のメモリモジュールを利用するという考え方をする。すなわち、配列等のデータ構造は基本的にそれぞれ独立の幅と深さを持ったメモリモジュール上に実現されると考える。
・すべての資源は静的であり、動的に生成、変更することはできないと考える。

ポインタの問題等は、複数メモリモジュールの利用とサイズの管理を意識すれば自然に制限される。後はアルゴリズムをごく自然に記述してくれればよい。以下、具体的にこれらの項目を解説する。

2.2 変数の符号とサイズを意識する

ハードコンシャス記述の第一歩は、変数の符号とサイズを意識することである。

以下に示すのは、C言語の勉強でもっとも早い時期に習う記述である。この記述の正しさに疑いを持つ者は多分いないだろう。

```
#define MAXSIZE 256
int i;
int A;
int x[MAXSIZE];
...
for (i = 0; i < MAXSIZE; i++)
   x[i] = x[i] + A;
```

しかし、よく考えるとこの記述はおかしいのではないだろうか？　変数iは配列の添字として使われている。したがって、iは負の数になることはあり得ない。また、iはこの記述の範囲ではMAXSIZEを超えることはないので、8bit幅でなければならない。少なくとも配列をアクセスする際に、iが255を超えれば明らかにエラーである。したがって、本来変数iは、以下のよう宣言するべきである。

```
unsigned char i;
```

iがintであろうがunsigned charであろうが一向に差し支

えないのは、プログラムが通常の計算機上で実行されるためである。ハードウェア化を意識する場合、符号付き数と符号なし数では演算回路やカウンタの動作が異なる。このため、本当に負の数を扱う場合以外はunsignedと宣言することで、ハードウェア量を節約することができる。変数のサイズはさらにハードウェア量に影響を与える。iに相当するレジスタが直接ハードウェアとして生成される場合、32bitレジスタとそれに要する加算器のコストは、8bitに要するそれと比べて4倍以上になる。C言語では、変数のサイズを表す手段が限定されているので、表現できないサイズが生じる。この場合、以下のようにして、対象とするサイズより一回り大きいサイズでタイプを宣言し、ハードウェア設計者がサイズを理解できるようにしてもらえると嬉しい。下の例では実際に演算を行うデータが24bitの場合を示している。

```
#define MAXSIZE 256
typedef UINT8 unsigned char;
typedef INT24 int;
UINT8 i;
INT24 A;
INT24 x[MAXSIZE];
...
for (i = 0; i < MAXSIZE; i++)
   x[i] = x[i] + A;
```

プログラマにとってすべての変数のサイズを意識するのは苦痛であろうから、可能な範囲、アプリケーションにとって重要な変数についてのみでよい。実際、多くの組込み用アプリケーションは扱う変数のサイズは一定に決まっており、数種類に定まるので、慣れればさほど不便ではなくなる。上記の記述を用いれば、この部分を直すだけで宣言部分はHandel-C、Bach-C、BDLに直接変換可能である。

さて、記述を書き直すことで、実は新たな問題が生じている。unsigned char が本当に8bitならば、iはMAXSIZEになるはずがない。したがって、このfor文の終了条件は決して満足されず、ループは終わらないことになる。実際Bach-Cでこの記述を行うとシミュレーション上無限ループになってしまう。Handel-Cでは、iとMAXSIZEの比較のビット数が合っていないというエラーが生じる。このことを考慮に入れたハードコンシャス記述は、以下のようになる。

```
#define MAXSIZE 256
typedef UINT8 unsigned char;
typedef UINT9 unsigned short;
typedef INT24 int;
UINT9 i;
INT24 A;
INT24 x[MAXSIZE];
...
for (i = 0; i < MAXSIZE; i++)
   x[(UINT8)i] = x[(UINT8)i] + A;
```

iがMAXSIZEを超えたことを判断するためには、8bitでは不可能だが、配列の添字としては8bitでなければならない。これを満足するには上記の記述が最も厳密である。

2.3 メモリに対する意識を変える

ソフトウェアによる処理とハードウェアによる処理の根本的な違いは、ソフトウェアがすべてを単一の広大な（仮想的には無限の）メモリ上で処理するのに対して、ハードウェアはデータを複数の限定された容量のメモリモジュールに分けて格納する点にある。したがって、単一メモリをあまりにも意識しすぎた書き方は、並列性を損ね、性能向上を阻害する。単一メモリを意識した場合には優れた記述でもハードコンシャス記述上は避けた方がよい場合もある。

下記は、JPEG encoder中に登場する記述で、RGB信号をCB信号に変換するテーブルctabをアクセスしている。

```
y = (UINT8)
  ((ctab[r+R_Y_OFF] + ctab[g+G_Y_OFF] + ctab[b+B_Y_OFF])
    >> SCALEBITS);
```

本来、RGBからYCBCRへの変換は、その組み合わせにより9通りあるので独立したテーブルが9個（正確にはB-CB、R-CRの変換は同じテーブルで済むので8個）必要なのだが、多くの配列を扱うのを避けるため（あるいは可読性を上げるため？）同一のテーブルctabに格納して、オフセットで区別してアクセスしている。この種の記述はメディア処理のプログラムに往々にして見られる。

しかし、ハードウェアでの実装を考えると、上記の記述は処理の並列性を損ねてしまう。単一メモリctabは一度に１つのデータの読み書きしかできないため、上記の記述では読み出しに３クロック要してしまい、その後に演算とシフト処理をスタートせざるを得ない（その前にオフセットの計算というおまけが必要かもしれない)。本来の意味どおり別のテーブルを用いて以下のように記述すれば、r_y_ab、g_ tab、b_y_tabを別々のメモリモジュールに割り付けることにより、１クロックですべての読み出しを終わらせることができる。

```
y= (UINT8)((r_y_tab[r] + g_y_tab[g] + b_y_tab[b])
  >> SCALEBITS);
```

さらに高速性が要求される場合は、加算器とシフタを多数設けて、すべての変換を同時に行うことにより１クロックで処理を終了させることすら可能である。

並列処理にはコストがかかるため、テーブルを分けることが常によいとは限らない。実際、当初の記述どおり、9つのテーブルをまとめて同一メモリモジュールに格納することで、メモリの面積効率を上げる（大きなサイズのメモリほどビット単位の面積が小さい）と共に、出力マルチプレクサのコストを倹約する場合も考えられる。実際のハードウェア化に際しては、デバイスと要求仕様を見ながら、コストと性能のトレードオフを考慮する必要がある。ハードコンシャス記述では、元々のアルゴリズムでテーブル等物理メモリが独立して存在する場合、わざわざそれを単一メモリに割り当てることをせず、極力そのままの形にしておく。このことで、並列性の可能性を明示し、ハードウェア設計者が行うトレードオフの検討範囲を広げる。

2.4　ポインタの利用は自然に限定される

ハードコンシャス記述だからといって、ポインタを使うな、と言っているわけではない。実際、C言語で記述するためにはポインタは避けては通れず、すべてのハードウェア記述用C言語で、限定的にポインタの利用を許している。ハードコンシャス記述でポインタの利用を避けるのは、ポインタが単一メモリを使うという思想と一体化している場合である。メモリのサイズが限定されており、かつ、幅もまちまちなのが複数存在する、ということを頭に入れることにより、ポインタの利用は自然に制限される。

ポインタが通常のソフトウェアで便利なのは、単一メモリ空間上でどのような場所でも、対象データのサイズを意識しないで相対的に指すことが可能であるためである。違ったサイズのデータに対してもキャストすることで扱うことができるし、C++等の動的バインディングでは、動かしてみるまで、扱うデータの型を決めないことすら可能である。ところが、ハードウェア化する場合には、物理的なメモリが複数別々に設けられ、そのサイズとデータ幅は、まちまちである。サイズが違えばアドレスの最大値も違うはずで、これを同じアドレスレジスタで扱うことは基本的にはあり得ない。したがって、ポインタのキャストは禁止される（幸いなことにポインタのキャストは作法ガイドでも制限されている（[ESCR]:R2.7.1))。しかしポインタは同じデータ幅とサイズの物理メモリが複数存在する際にこれを指定する方法として有用である。これは、関数間でデータをやりとりしたり、ある関数で複数のデータブロックを入力として切り替えて用いる場合等である。

```
UINT16 qtbl0[DCTSIZE2];
UINT16 qtbl1[DCTSIZE2];
...
compress_one_block(comp_y, y_index, DCTSIZE, 1,
                           qtbl0,dctbl0_co,dctbl0_si,
                           actbl0_co, actbl0_si,
                           &last_dc_val_y);
...
compress_one_block(comp_cr, cr_index,MBSIZE,2,
                           qtbl1,dctbl1_co,dctbl1_si,
                           actbl1_co, actbl1_si,
                           &last_dc_val_cr);
```

上記の記述は、１つの関数compress one blockが違った配列（qtbl0, qtbl1）に対して同じ処理を行うことを示すために配列名をポインタとして与えている。また、最後のデータを取り出すため、last_dc_val_y、last_dc_val_cr等の変数（レジスタ）のアドレス（名前）をポインタとして与えている。すなわち、ポインタは、アドレスという形で物理的なデータ構造の名前を渡すことで、関数の共有を実現している。

もちろん、ハードウェアによる実装レベルでは、この関数を本当に共有すべきか、それともコピーを作って並列に処理すべきかを考慮しなければならない。しかし、これは対象デバイス、要求仕様が揃った時点で、ハードウェアデザイナが行うべき仕事であろう。ポインタを物理的なデータ構造に対する名前を渡すもの、として考えれば、このトレードオフの検討が容易になる。ハードコンシャス記述では、このような用途でポインタを用いる。

2.5 ポインタの二重構造、リンクトリスト

ポインタの二重構造やリンクトリストは、それ自体ハードウェア化できないわけではない。一定の容量のメモリを設けて、その上で、アドレスレジスタと制御回路を生成して、ソフトウェアで実行するのと同じようにして、実現すればよい。しかし、これを実行すると動作速度はメモリのアクセス速度に律速され、ソフトウェアに対してほとんど高速化できない。メモリアクセスの途中に複雑なメモリ量圧縮用の演算が必要な場合等は例外的にハードウェア化による効果が見込まれるが、多くの場合は、リンクトリストを実現する最も効率のよいハードウェアはCPUそれ自体になってしまい、ハードウェアオフローディングの意味がない。さらに、ハードウェアではメモリが沸いて出るわけがないので、mallocのような動的なメモリ確保は禁止というよりはまったく不可能である（幸いなことにコーディング作法でも動的生成を使用しないように勧めている[ESCR](M5.2.1)）。したがって、データ構造として、ポインタの二重引きを行って、リンクトリストを作る必要性はほとんどないと考えられる。

そこで、ハードコンシャス記述では、リンクトリストは考えず、ポインタの二重引き、すなわちポインタに対するポインタは、通常の場合、使わない（幸いにして三重引きは作法ガイドでも禁止されている[ESCR](M3.4.1)）。組込み用途の処理において、多くの場合はリンクトリストは通常の配列アクセスに変換可能である。リンクトリストがどうしても必要なアルゴリズムは本来オフロードの対象とすべきではないのである。

2.6 配列とポインタ

ハードウェア記述用C言語でも、単純な配列に対するポインタは許されるが、添字を使って書ける場合は、極力配列の添字で表すことをお勧めする。

例えば、以下の記述はJPEG-encoderで用いられているもので、わざわざ配列演算にポインタを用いている例である。

```
unsigned char *y_ptr, *cb_ptr, *cr_ptr;
  b_ptr = &comp_b[0];
  y_ptr = &comp_y[0];
  cb_ptr = &comp_cb[0];
  cr_ptr = &comp_cr[0];
  for (i = 0; i < org_h; i++){
    for (j = 0; j < exp_w; j++){
      r = (j < org_w) ? *r_ptr++ : *(r_ptr-1);
      g = (j < org_w) ? *g_ptr++ : *(g_ptr-1);
      b = (j < org_w) ? *b_ptr++ : *(b_ptr-1);
      ...
```

この記述は下の添字を用いたハードコンシャス記述に比べて何ら優れた点はないように思える。スマートに書けて格好よいのだろうか？　確かに行数は少なくなっているが、可読性はかえって悪くなっているように見える。

```
INT24 y_index, cb_index, cr_index;
  r_index=0; g_index=0; b_index=0;
  y_index=0; cb_index=0; cr_index=0;

  for(i=0; i < org_h; i++) {
    for(j=0; j<exp_w; j++) {
      if(j<org_w) {
        r = comp_r[r_index]; r_index++;
        g = comp_g[g_index]; g_index++;
        b = comp_b[b_index]; b_index++; }
      else {
        r = comp_r[r_index-1];
        g = comp_g[g_index-1];
        b = comp_b[b_index-1]; }
  ...
```

ポインタを用いた記述は、ハードウェア化を考えると、ポインタ自体のサイズが見えないという問題点がある。ソフトウェアでは、ポインタは単一の決まったメモリを指すために、ポインタ自体の型は、CPUによって決まっていて問題は生じない。しかしハードウェア化する場合は、ポインタを設けたら、それに相当するアドレスレジスタが個別に必要になる。ポインタを使う場合、対象の

変数の型のみを指定して宣言するため（これがポインタの利点なのだが）、この例では対象の型が8bitであることはわかるのだが、ポインタ自体のサイズは記述方法上指定することができない。これに対して、ハードコンシャス記述では、r_index等の添字変数に相当するのがアドレスレジスタとなり、このサイズは、アクセスする配列のサイズから明解に計算でき、宣言することが可能である。

機能合成ツールが十分賢ければ、初期化する配列のサイズから計算することができるであろう。しかし、実際は、シミュレーションは可能でも、実際に機能合成をするとエラーになる場合が多い。わざわざポインタにするメリットが見いだせない以上、ハードコンシャス記述では添字を用い、アドレスレジスタのサイズを明確化すべきである。

2.7 実行前にできることはやっておいて初期値として与える

メディア処理等では、あらかじめ値の入ったテーブルを用いる場合があるが、これらは、ハードウェアではROM上に置くと効率的である。テーブル等の初期化は、それが常に同じように行われるならば、あらかじめ初期化した結果をROMに入れておく方が多くの場合、有利である。例えば、以下の記述はJPEG-encoderで、ハフマン符号化に用いる構造体を初期化する記述である。

```
void init_htbl(unsigned char *bits,
               unsigned char *huffval,
               c_derived_tbl *htbl
               )
{
  int h, i, j;
  int n, v, c;

  for (i = 0; i < 256; i++) {
    htbl->ehufsi[i] = 0;
    htbl->ehufco[i] = 0;
  }
  h = 0;
  c = 0; /* Huffman code */
  for (i = 1; i <= 16; i++) {
    n = bits[i];
    for (j = 0; j < n; j++) {
      v = huffval[h++];
      htbl->ehufsi[v] = i;
      htbl->ehufco[v] = c++;
    }
    c <<= 1;
  }
}
```

この初期化プログラムは複数の構造体に対して適用され、小さなテーブルhuffvalから複数の構造体に対してデータをコピーしつつ若干の前処理をすることにより、テーブルの初期化を行っている。ところが、この処理をハードウェアで実現する場合、まず初期化をする制御回路とデータパス、加算器等が必要になる。次に元のhuffalとコピー先の構造体用のメモリが必要になる。

一方で、この処理をあらかじめ行った結果をファイルに落としておき、これを直接対象のテーブルにROMとして格納（FPGAの場合は構成データとして読み込む）すれば、これらの回路はすべて不要になり、他で使ってなければhuffval用のメモリモジュールも不要となる。

```
static c_derived_tbl dctbl0 = { {
#include "ehufco0.txt"
},
{
#include "ehufsi0.txt"
} };
```

このように、ハードコンシャス記述では、プリプロセッサ等の利用により、コンパイル時にできる計算は可能な限り済ましておくことが望まれる。もちろん、データが規則的で生成される対象が膨大であるため、手続き的に初期化した方が有利な場合や（このような場合はテーブルを使うこと自体が意味がないような気もするが）、入力データにより初期化プロセルが動的に影響を受ける場合は、このようなことはできない。

2.8 構造体の扱い

構造体は、異なったデータをまとめて扱うことにより、可読性を上げることができ、有効な記述法である。すべてのハードウェア記述C言語で利用が可能であり、実際に利用されている。構造体のメンバがポインタである場合は、添字に変換可能な場合がほとんどであるため、先

の原則に従って添字を用いれば問題は生じない。また構造体へのポインタも、物理的なデータ構造に対する名前を示すだけならば、基本的には問題は生じない。しかし、構造体の配列については議論の余地がある。

例えば、

```
struct _NODE
{
  int iDist;
  int iPrev;
  int Cost[NUM_COST];
};
typedef struct _NODE NODE;
NODE rgnNodes[NUM_NODES];
```

という配列rgnNodesは、ハードウェアでどのように実装すればよいのだろうか？　単一の物理メモリ上にべたに並べると、添字から要素へのアドレス変換に手間がかかり、さらに、rgnNodes[i].Cost[j]とrgnNodes[i].iDistが同時にアクセスできなくなってしまう。このため処理の並列性が制限される。

結局、要素ごとに独立の物理メモリを用いるのが最善の策である。つまり下記と全く同じである。

```
UINT16 rgnNodes_iDist[NUM_NODE];
UINT16 rgnNodes_iPrev[NUM_NODE];
INT16 rgnNodes_Cost[NUM_NODE][NUM_COST];
```

この場合、rgnNodes0、 rgnNodes1等複数の構造体を扱う場合、記述のスマートさが失われるが、やむを得ないと考える。Bach-Cでは構造体の配列自体を許しておらず、Handel-Cは可能だが、合成上の制約が大きい。構造体の配列を使わないことにより、実際のハードウェアイメージに近い記述となり、変換時の手間を省くことができる。

3. おわりに

現在、組込みシステム評価用アプリケーションプログラム集MiBench[MIBENCH]のハードコンシャス記述化を進めている。現在のところ、ハードコンシャス記述から、Handel-C及びBach-Cへの変換は人手で数時間で可能であり、とりあえず動作可能なコードができることを確認している。ただし、もちろん、これがそのままの形で、いつでもうまく動くハードウェアになるわけではない。ハードウェアの設計は、実装するデバイスと要求仕様が幅広い。デバイスにより搭載可能なゲート数やメモリ、それらの遅延時間が決まっており、消費電力の要求も時には厳しい。性能を上げるためには、メモリを分散させ、多数の演算装置を用いて並列処理を行うのが有効だが、このような設計はコストと消費電力を必要とするため、常に最良とは限らない。ハードウェア設計者の仕事は、これらのバランスを考え、デバイスと要求仕様に合わせて最適な設計を行うことである。すなわち、どんな場合にも優れた設計というものは存在せず、場合に応じて様々なトレードオフを検討する必要がある。ハードウェア記述用C言語による記述と機能合成ツールの利用はこのトレードオフの検討を素早く行える点にそのメリットがある。ハードウェア記述用Cのコードを何度も書き直し、コスト、性能、消費電力の組み合わせが最もうまくいく点を見つける作業をすること自体が、ハードウェアの設計、実装に相当する。

この作業を行うためには、元々のアプリケーションがハードウェア設計者にとってわかりやすく、トレードオフの検討がしやすいように書かれていることが重要である。ハードウェア設計者はアプリケーションのプロではないため、実はこのプロセスが非常に困難であり、大きな障害となっている。この時点で深い理解が行われないと、設計の選択肢が限定され、ソフトウェアによる逐次処理をそのままハードウェア化しただけの回路が出来上がる。これではCPUによるソフトウェアの実行を大きく高速化することができず、オフロードの意味がなくなる。ハードコンシャス記述はこの作業を行うに際して、アプリケーションプログラマとハードウェア設計者のインタフェースとして利用されることが期待されている。

ハードコンシャス記述は、ハードウェア設計者からのアプリケーションプログラマに対する御願いである。コーディング作法ガイド[ESCR]にならって「作法」と呼ぶが、実際は「御願い」あるいは「懇願」である。あなたの書くプログラムが莫大なメモリを持ち、きちんとしたOSが動くPCやサーバ上で動くだけならば、何の問題も

ない。オブジェクト指向の概念、動的バインディング、継承、ポインタのキャスト等の技法をどんどん用いて、高度に抽象化された生産性、保守性に優れたプログラムを書いてほしい。我々は、C++、JAVAにけちを付けているわけではない。しかし、あなたのプログラムが組込みチップ上で動くことを想定しているならば、そしてハードウェアにオフロードされて高性能のSoC として動作することが期待されているならば、最低コーディング作法[ESCR]を守り、できればハードコンシャス記述を用いて欲しい。

動的バインディング、複雑なクラス構成と継承を用いた記述を使ってプログラミングを行うことは、ハードウェア設計者に向って「お前ら、アルゴリズム自体を理解して一から書き直せ」と言っているに等しい。これらのプログラムはその計算結果がリファレンスとして使えるだけで、ほとんどハードウェア設計の役には立たない。要するに、目的に応じて記述方法を変えてもいいじゃないか、ということである。また、我々は仕事の一部をアプリケーションプログラマに御願いしてサボろうと考えているわけではない。ハードコンシャス記述は、記述における意識を変えてもらうだけで、特定のハードウェアを念頭に置いて書いてくれることを要求するわけではない。あくまでアプリケーションを、単一メモリのプログラム格納型の抽象化の考えに酷く侵されることなく自然に書いて欲しいだけである。そこから先、すなわち、ハードウェア記述言語に変換して、与えられたデバイス、要求仕様を満足させる作業は純粋にハードウェア設計者の仕事である。

ちなみに、ハードコンシャス記述は、書き方自体があまりエレガントでなく、この点が非常に気になっている。これは提案を行った筆者が、基本的にハードウェアの設計者であり、エレガントな言語センスを持っていないことに起因する。にもかかわらず、敢えてこのような提案を行ったのは、C++を用いた絶対にハードウェア化できない組込み用アプリケーション記述がどんどん蓄積されていくのを座視するに耐えなかったためである。コンピュータサイエンスとハードウェア設計の両方の文化を理解し、しかもセンスに優れた若手が、ハードコンシャス記述をもっと洗練してくれることを期待する。また、この記述に賛同してくれるアプリケーションプログラマが増え、多くのアプリケーションが蓄積され、少しでも現状が改善されることを願う。

参考文献

[BACH2004] Bach-C 言語マニュアル，シャープ株式会社，2004
[ESCR] IPA SEC：組込みソフトウェア開発向けコーディング作法ガイド［C言語版］，翔泳社，2006
[Handel2005] Handel-C Language Reference Manual: Celoxa, 2005
[MIBENCH] http://www.eecs.umich.edu/mibench/
[WAKABAYAQSHI] 若林一敏：LSI 開発はこう変わるC言語設計の最前線（1），http://techon.nikkeibp.co.jp/NEWS/dac2003/030317 1.html

The Open Group

http://www.opengroup.org/

オープン・グループ日本代表・会長　グローバル情報社会研究所株式会社　代表取締役社長
信州大学大学院客員教授　京都大学講師　CRM協議会理事長
藤枝　純教

The Open Groupは、次世代ITシステムの標準化を推進するグローバルな非営利団体であり、世界の25ヵ国を超える300社近い会社／組織が参加している。ITユーザ／ITサプライヤが一体となって活動する中立的なコンソーシアムであり、ユーザの目線でのインターオペラビリティ（Interoperability）とクオリティ・オブ・サービス（Quality of Service： QoS）を追求し、真のオープン・システム実現に向けて活動している。

1　活動の要はグローバル・オープン標準とグローバル・インターオペラビリティ

世界には100以上のIT標準団体があるといわれているなか、The Open Groupは、グローバル・オープン標準及びグローバル・インターオペラビリティに関する中心的な活動を展開している。重要な問題はボードで議論し、合意ができれば、これらを解決するために関連団体と話し合い、提言・助言を行う。他の団体が策定した標準仕様・製品をもテスト・認証する役割も果たしてきた。UNIX、 DCE、Motif、CORBA、CDSA、Linux Standard Group、携帯のWAP、DOD/DISAのCOE標準認証、等これらはThe Open Groupの成果である。

現在の活動テーマは、アーキテクチャ、プラットフォーム、リアルタイム・エンベデッド、SOA、Net-Centric Operation Industry Consortium、Semantic Interoperability、ソフトウェア品質検証など、10以上のテーマがあり、海外において年4回コンファレンスやフォーラムを開催している。

また、The Open Groupは、ITアーキテクト、IT管理者、CTO、CIO各メンバ同士の情報と人的交流の場を提供すると共に、次世代CIO、CTOの育成プログラム等も開発し、人材育成にも力を注いでいる。

2　グローバル・オープンへの展開

20世紀の物価・産業社会は「モノ」の価値が中心であり専有を価値と考えていたが、現在のグローバル情報社会は、知の創造と共有、ソフトウェアの標準化とその使い方、サービスの標準化を土台としたネットワーク中心の社会に突入したといえる。

「組織と情報のグローバル化、オープン化とガバナンス能力に関し、経営競争力は？」、「オープン標準化創造活動への参画は？」、「国や企業としてオープン標準の利用成熟度と経営生産性の関連の把握は？」、「オープン技術のCTOは育っているか？」等の課題を突き詰めて考えていくと、グローバル・マーケットに入るためにはオープン標準の製品やサービスを利用することが重要である。また、様々な機器・装置は単機能製品から、複雑なネットワーク型複合システム製品へと進化をした結果、より統合しやすいソフトウェアやサービス、そしてアーキテクチャの「オープン標準」が求められている。

グローバル・オープンへのこのような展開に対し、The Open Groupは、低コストで、安全性・相互運用性の高いITシステム標準をユーザに提供すべく、ソフトウェアからサービスに到る、高品質でインターオペラブルなグローバル標準の確立に向けて取り組んでいる。

3　日本企業への期待

世界をリードしているトヨタやキヤノン等、日本のグローバル企業は、昭和30年代から半世紀もの業界標準・グローバル化を行い、事業における海外比率はトヨタにおいては自動車販売台数で70%を超え、キヤノンは売り上げで同じく70%を超えた。これら先行成功者が排出さ

れたのは、50年近い歳月をかけてグローバル化やオープン化を進めた結果によるところ大であろう。

いま日本には、IT産業はもうだめだという声がある。それは旧来のクローズ・アーキテクチャではだめということであって、グローバル・オープン標準とその認証に基づいたグローバル製品やサービスを創造・開発していけば、先行成功者のような企業が続々現れてくるに違いない。

The Open Groupの活動に参加している20社超の日本のメンバ企業は、ITの技術的テーマのみならず、スキルやベストプラクティス、ITマネージメント等のテーマにも参画・活動している。2007年1月末、米サンディエゴで開催されたThe Open Groupのカンファレンスに初めて参加した、日本のあるIT企業のトップは「日本の企業は、このような場を通じてIT外交をもっと積極的に行うべきだ。日本はITでも外交下手だ」と感想を述べた。

現在日本からは、ボードメンバであるNECをはじめ、富士通、日立、東芝らITメーカのほかに、経済産業省のIPA、総務省のNICTの参画を得て、トヨタITC、みずほ総研、リコー、松下電工、アサヒテクネイオン、ベリサーブらがメンバ活動をしてきた。ここにきて日産や野村総研、SIOS社らが新しくメンバになった。

更なる日本のITサービス企業・ユーザ企業もグローバルでオープンな標準を創造する国際活動に積極的に参加し、出来ればこれをリードし、オープン標準に準拠し、きめ細かく使い易い、高品質な製品やサービスを日本から創出すべきである。そのためには具体的に、大・中企業は勿論、世界を目指すベンチャー企業も、自社に社長直轄のオープン担当役員（CIO、CTO兼務も可）を置き、自社の組織のオープン化と製品・技術・組織のオープン化標準対応をグローバルに推進する必要がある。

大小を問わず日本の企業は、2〜5年先のグローバル・オープン標準の流れを汲みとり、世界のオープンリーダ達と協業し、製品やサービスをグローバルにオープン展開することで、日本のIT産業及び産業全般のグローバルビジネスの成功を引き寄せるべきであろう。

4　次世代を担う人材育成の場として

The Open Groupは近年、「境界のない情報の流れ」（Boundaryless Information Flow）を組織目標に掲げ、経営に役立つITシステム、組織の壁を越えた情報の流れを実現すべく、CIO、CTO、ハイレベルのITアーキテクトが関心を持つテーマのもとにカンファレンスを開催し、マネージメント・レベルの啓蒙に重点をおきつつある。

また、経営戦略を反映したITシステム構築の基盤であるEA（エンタープライズ・アーキテクチャ）については、グローバルEA標準手法であるTOGAF（The Open Group Architecture Framework）を完成した。そのTOGAFを使用して最適なITアーキテクチャを策定、実装、ガバナンスについて指導できるアーキテクトの育成とスキル認証プログラムなど、人材育成面にも焦点を当て、2007年初めに4,000名のアーキテクトの認証者を創出し、日本でも現在36社172名の認証者、2007年末までに200名を超える見込みである。

オープン標準の問題ないし課題をThe Open Groupがすべて解決できるわけではない。The Open Groupがすることは、ユーザの経営層に近いCIO、CTOの声を中心に、世界の標準団体の提供するオープン標準をユーザセントリックな目線で捉え、インターオペラビリティとQoSに関して問題ないし課題を発見した場合に、ガバニングボードでのプライオリティに基づき、そのテーマごとに盟友であるW3C、OMG、OASIS、WAPなどの関係標準団体と協調しつつ問題解決に全力を尽くすことである。

このような活動内容は、ユーザ企業や団体から見れば、The Open Groupに参加しさえすれば、ワンストップ・ショッピングでオープン標準のCIOやCTOが問題にするべきテーマと解決の方向についての十分過ぎるほどの知見を得るだけでなく、世界の300社に及ぶグローバル企業のCIO、CTO等と交流し人脈ができることにより、大学院やCS学科に海外留学すること以上の人材育成効果が期待できる。

The Open Groupは、オープン化、グローバルなオープン活動への国際的参画を今後も提言し続ける。

BOOK REVIEW

ずっと受けたかったソフトウェアエンジニアリングの授業（1）（2）

鶴保 征城・駒谷 昇一　共著

ISBN：（1）4-7981-1154-6　（2）4-7981-1155-4　翔泳社刊
A5判・216頁・（1）（2）巻それぞれ定価2,100（税込）　2006年10月刊

愛情溢れる「ソフトウェア開発の仕事」

柔らかい装丁に比して、著者の実力を反映した深い内容である。ソフトウェア開発にかかわるすべての人に贈られた、驚きと愛情溢れる良書である。大学生は素直に教科書として読めばよいが、産業界の人々は現代の学生と産業界の距離を認識し、若者に何を伝えなくてはいけないか体得するのに素晴しい内容である。学界の指導者にとっても、今必要な教育内容について全体を俯瞰できる好著である。このような教科書で授業を受けられる学生は幸せである。「産業界では一人で行う仕事は無い」とか、「評価を受けるだけでなく、他人を評価できることの重要性」を訴える冒頭部分から一気に引き込まれる。中央部の実務の説明は平易に見えるなかに含蓄のある内容が続く。示されている「バグ票」の書式ひとつとっても、1970年代からの歴史ある管理項目のエッセンスがさりげなく示されている。後段の「ソフトウェア産業の課題」は非常に充実しており、最終章の「ソフトウェア産業での仕事のやりがい」は著者のこの産業、そして学生への愛情溢れた感動の終章となっている。本書の前段で「ソフトウェア開発は生半可な気持ちでは成功しない」と断言し、結びでは「ソフトウェア開発とは非常に多彩な才能を必要とする、大変面白い仕事である」と述べて、読者をソフトウェア産業へ誘っている。　（神谷芳樹）

初めて読むドラッカー【技術編】テクノロジストの条件 ものづくりが文明をつくる

P.F. ドラッカー　著　上田 惇生　訳

ISBN：978-4-478-30072-5　ダイヤモンド社刊
四六判・295頁・定価1890円（税込）　2005年7月刊

技術者の重要性を再認識せよ

ビジネス界のカリスマであるドラッカーの入門シリーズとして発刊されている書籍である。ドラッカーの数ある名著から技術に関するテーマを抜粋した書籍であり、執筆は40年以上前の著書も含まれているが、全く古さを感じさせない。このあたりがドラッカーのカリスマたる由縁かもしれない。本書でいうテクノロジストとは、高度な知識の裏づけのもとに高度な技能をもつ人材のことである。

蒸気機関という発明を、技術として整理し、高度なものづくりのスキルを発揮し製造を行ったテクノロジストの存在が、産業革命を実現した功労者である。また、ここ100年の爆発的な生産性向上は機械化でなく、1800年代後半のテイラーが行った、仕事の分析と仕事への知識の適用であった。

しかし、ITなど知識労働者と呼ばれる者の生産性向上は、まだまだ手付かずといわれている。製造における機械は、技術者のスキルがそれに相当するといえる。仕事のプロセス分析と、技術者のスキル向上によって生産性を高めなければならないと痛感する。

ドラッカーは、巻末にある2005年のインタビューにおいて、テクノロジストは、先進国で唯一といっていいほどの競争要因と断言している。体系的な知識を獲得させ、高度な技能を確実に発揮できる技術者の育成に対して、戦略的に取り組むことが求められる。本書は、その戦略におけるベースとなる思想を理解・整理するには最適の書籍である。　（渡辺登）

ソフトウェア・エンジニアリング関連イベントカレンダー

作成：SEC journal編集委員会

開催年月	開催日	イベント名	主　催	開催場所	URL
2007年5月	16（水）～18（金）	第16回ソフトウェア開発環境展（SODEC）	リード・エグジビション・ジャパン	東京都江東区・東京ビッグサイト	http://web.reedexpo.co.jp/SODEC/
	16（水）～18（金）	第10回組込みシステム開発技術展（ESEC）	リード・エグジビション・ジャパン	東京都江東区・東京ビッグサイト	http://web.reedexpo.co.jp/ESEC/
6月	6（水）～7（木）	Embedded Technology West 2007 －組込み総合技術展 関西－	社団法人 組込みシステム技術協会（JASA）	大阪府大阪市・マイドームおおさか	http://www.jasa.or.jp/etwest/
	28（木）～29（金）	IPAX2007／SEC Forum 2007	IPA	東京都文京区・東京ドームシティ・プリズムホール、東京ドームホテル	http://www.ipa.go.jp/
7月	19（木）～20（金）	ソフトウェアプロセスエンジニアリングシンポジウム（SPES2007）	社団法人 情報サービス産業協会（JISA）	東京都江東区・日本科学未来館7階ホール及び会議室	http://www.jisa.or.jp/
8月	27（月）～29（水）	ソフトウェアエンジニアリングシンポジウム2007（SES2007）	社団法人 情報処理学会 ソフトウェア工学研究会	東京都江東区・日本科学未来館	http://ses2007.cs.shinshu-u.ac.jp/
9月	5（水）～7（金）	FIT2007 第6回情報科学技術フォーラム	社団法人 情報処理学会、社団法人 電子情報通信学会 情報・システムソサイエティ、社団法人 電子情報通信学会 ヒューマンコミュニケーショングループ	愛知県豊田市・中京大学 豊田キャンパス	http://www.ipsj.or.jp/
	6（木）～7（金）	第26回ソフトウェア品質シンポジウム	財団法人 日本科学技術連盟	東京都文京区・東洋大学	http://www.juse.or.jp/
	10（月）～14（金）	第11回Software Product Line Conference（SPLC 2007）	ソフトウェアプロダクトライン国際会議国内実行委員会	京都府京都市・京都リサーチパーク	http://sec.ipa.go.jp/SPLC2007/
10月	1（月）	情報化月間記念特別行事	経済産業省	（未定）	http://www.ipa.go.jp/
	（未定）	IPA Forum 2007 SEC コンファレンス 2007	IPA	（未定）	http://www.ipa.go.jp/
	31（水）～11月2（金）	SPI Japan 2007（ソフトウェアプロセス改善カンファレンス）	日本SPIコンソーシアム	富山県富山市・富山国際会議場	http://www.jaspic.jp/
11月	14（水）～16（金）	Embedded Technology 2007 －組込み総合技術展－	社団法人 組込みシステム技術協会（JASA）	神奈川県横浜市・パシフィコ横浜	http://www.jasa.or.jp/

上記は変更される場合があります。参加の際に必要な詳細事項は主催者にお問合せをお願いいたします。

イベントのご案内

春から夏にかけて、SEC関連のセミナー等が多数開催されます。是非ご参加ください。

【Embedded Technology West 2007－組込み総合技術展 関西－】

http://www.jasa.or.jp/etwest/
開催日…6月6日（水）、7日（木）
会　場…マイドームおおさか（大阪市）
入場料…無料（事前登録制）
6月6日

●コミュニティセッションC－2（13:00～14:50　第3会議室）

「SEC2007年度成果プレビュー～組込みソフトウェアの品質と生産性向上へ～」
13:00～14:00 「組込みソフトウェア向け開発プロセスガイド：ESPRの読み方」
山崎太郎（SEC組込みプロジェクト研究員）
14:00～14:50 「組込みソフトウェア向けプロジェクトマネジメントガイド：ESMRの読み方」
室修治（SEC組込みプロジェクト研究員）

●コミュニティセッションC－3（15:10～17:00　第3会議室）

「地域活用セミナー」
15:10～15:40 「西日本地域における高度組込み人材育成」
田丸喜一郎（SEC組込みプロジェクトサブリーダ）
15:40～17:00 パネルセッション
「高度組込み人材育成で差別化を狙う地域西日本」
コーディネータ：田丸喜一郎
パネリスト：山添祥則氏（関西エンベデット技術者育成研究会設立委員長／松下電器産業株式会社人材開発カンパニーコーポレート技術研修センター所長）
上田斉氏（中国経済産業局地域経済部地域経済課参事官（電子情報産業担当）付情報化推進係長　調査官（電子情報産業担当））
廉屋則夫氏（北九州市産業学術振興局理事）
福田晃氏（九州大学大学院 システム情報科学研究院情報工学部門教授工学博士）

【IPAX2007／SEC Forum 2007】

http://www.ipa.go.jp/
開催日…6月28日（木）、29日（金）
会　場…東京ドームシティプリズムホール（東京都文京区）
入場料…無料（事前登録制）
●6月28日
13:00～17:00 SEC組込み系プロジェクト活動報告、経済産業省組込みソフトウェア産業産業実態調査報告
●6月29日
10:00～17:00 招待講演
株式会社東京証券取引所代表取締役社長CEO　西室泰三氏
SECエンタプライズ系プロジェクト活動報告

【ソフトウェアプロセスエンジニアリングシンポジウム（SPES2007）】

http://www.jisa.or.jp/seminar/spes2007/
開催日…7月19日（木）、20日（金）
会　場…日本科学未来館7階ホール及び会議室
入場料…無料（事前登録制）
●7月19日　13:50～14:40 「オープニングパネル（仮称）」
パネリスト：鶴保征城（SEC所長）他
●7月20日　13:50～16:00 「SECテクニカルセッション（仮称）」
プロセスに関して4セッションを実施予定。

編集後記

今回の「SEC journal」は、昨年（2006年発行 第6号）と同様「SEC活動概要」特集号です。「エンタプライズ系」「組込み系」「共同研究」の各活動領域について、1年間の様々な活動を1～2頁ずつに凝縮いたしました。執筆者の皆様には短い納期での無理な執筆依頼となりましたが、ご快諾いただき感謝します。なお、SECの活動領域には「先進ソフトウェア開発」もありますが、現在COSEプロジェクトの最終報告書を作成している段階ですので、領域としての報告は次号に掲載させていただくことと致しました。ご了承ください。

なお、今回SEC研究員の立場としてご執筆いただいた石谷さん、吉田さん、佐藤さんは、4月より出向元に戻ってご活躍中です。

「SEC journal」は、現場重視の企業と、最新のソフトウェアエンジニアリング手法について研究している大学との情報共有を目的の1つとしており、ご投稿いただいた「SEC journal論文」（ソフトウェア開発現場のソフトウェアエンジニアリングをメインテーマとした実証論文）の掲載や、大学とSECの共同研究の成果報告を行っております。しかし、「SEC journal論文」は産業界からの投稿が少なく、論文審査委員の方々から、『企業の方に現場で実証した内容を論文としてまとめて頂くのは、難しいのではないか？　テクニカルレポートのようなものを考えてはどうか?』というご意見も頂いております。そこで、SECでは、企業の開発現場の方が投稿しやすい形として、「ソフトウェアエンジニアリング活用大賞」（仮称）の募集を検討中です。8月に応募を締め切り、10月に発表の予定で検討を進めておりますので、SECの成果物を企業現場にて活用し成果を得られている方は、是非投稿のご準備をお願い致します。詳細は追って、SEC-Webサイトにて発表いたしますので、今しばらくお待ちください。

また、SEC-WebサイトにおけるSECの成果物提供につきましては、報告書、書籍等のドキュメントだけでなく、ツール等も提供開始する予定です。見やすく利用しやすいサイトを目指してリニューアル準備中ですので、今後もSEC-Webサイトにご注目をお願い致します。

本journalに対してのご意見とSECへの情報提供・問題提起等には、SEC-Webサイト（http://sec.ipa.go.jp/）内の「SECへのお問い合わせ」をご利用ください。

（ヒゲ）

SEC journal 編集委員会

SEC journal® 第3巻第2号（通巻10号） 2007年5月28日発行

編集兼発行人　〒113-6591　東京都文京区本駒込2-28-8　文京グリーンコート センターオフィス16階
独立行政法人 情報処理推進機構　ソフトウェア・エンジニアリング・センター　所長　鶴保　征城
Tel.03-5978-7543　Fax.03-5978-7517
http://sec.ipa.go.jp/

編集・制作　〒101-8460　東京都千代田区神田錦町3-1　株式会社オーム社　Tel 03-3233-0641

お知らせ

SEC journal 論文募集

独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センターでは、
下記の内容で論文を募集します。

論文テーマ

ソフトウェア開発現場のソフトウェア・エンジニアリングをメインテーマとした実証論文

- 開発現場への適用を目的とした手法・技法の詳細化・具体化などの実用化研究の成果に関する論文
- 開発現場での手法・技法・ツールなどの様々な実践経験とそれに基づく分析・考察、それから得られる知見に関する論文
- 開発経験とそれに基づく現場実態の調査・分析に基づく解決すべき課題の整理と解決に向けたアプローチの提案に関する論文

論文分野

品質向上・高品質化技術
レビュー・インスペクション手法
コーディング作法
テスト／検証技術
要求獲得・分析技術、ユーザビリティ技術
見積り手法、モデリング手法
定量化・エンピリカル手法
開発プロセス技術
プロジェクト・マネジメント技術
設計手法・設計言語
支援ツール・開発環境
技術者スキル標準
キャリア開発
技術者教育、人材育成

論文の評価基準

a. 実用性（実フィールドでの実用性）
b. 可読性（記述の読みやすさ）
c. 有効性（適用した際の効果）
d. 信頼性（実データに基づく評価・考察の適切さ）
e. 利用性（適用技術が一般化されており参考になるか）
f. 募集テーマとの関係

論文賞

「SEC journal」では、毎年「SEC journal」論文賞を発表しております（前回は2006年10月24日SECコンファレンス）。受賞対象は、「SEC journal」掲載論文他投稿をいただいた論文です（論文賞は最優秀賞、優秀賞、SEC所長賞からなり、それぞれ副賞賞金100万円、50万円、20万円）。

応募要項

スケジュール

・12号（2007年11月発行予定）
応募締切　2007年7月21日
投稿締切　2007年7月28日
採否通知　2007年8月12日

採録決定後、1週間程度のカメラレディ期間があります。
詳細は別途通知されます。
採録の場合には「SEC journal」への掲載およびIPA SECのWebやイベント等での発表を行います。
詳しくはSEC-Webサイトよりお問い合わせください。

提出先

独立行政法人 情報処理推進機構　ソフトウェア・エンジニアリング・センター内 「SEC journal」事務局
eメール：sec-ronbun@ipa.go.jp

その他

- 論文の著作権は著者に帰属しますが、採択された論文については「SEC journal」への採録、ホームページへの格納と再配布、論文審査会での資料配布における実施権を許諾いただきます。
- 提出いただいた論文は返却いたしません。

応募様式

応募様式は、下記のURLをご覧ください。

http://sec.ipa.go.jp/secjournal/oubo.php

SEC journalバックナンバーのご案内

詳しくはhttp://sec.ipa.go.jp/secjournal/をご覧ください。

No.1　No.2　No.3　No.4　No.5　No.6　No.7　No.8　No.9

SEC journal No.10
第3巻第2号（通巻10号）
2007年5月28日発行

編集兼発行人
〒113-6591　東京都文京区本駒込2-28-8　文京グリーンコート センターオフィス16階
独立行政法人 情報処理推進機構　ソフトウェア・エンジニアリング・センター
所長　鶴保　征城

Tel.03-5978-7543　Fax.03-5978-7517
URL:http://www.ipa.go.jp/
定価1,470円（本体1,400円）

IPA®
独立行政法人 情報処理推進機構

ISSN 1349-8622