2010年10月25日発行
第6巻第3号（通巻24号）
ISSN 1349-8622

SEC journal 22

Software Engineering Center

巻頭言
藤江 一正 IPA理事長

所長対談：岩野 和生　日本アイ・ビー・エム株式会社　執行役員　未来価値創造事業
新しい社会サービスの実現に向けて その課題と解決策を考える

技術解説
九州工業大学における パーソナルソフトウェアプロセス教育
ーソフトウェア品質向上のためのスキル修得ー

組込みソフトウェアによる信頼性及び安全性
信頼性の高い組込みソフトウェアを開発するには 安全性向上への要求工学の貢献の可能性

論文
Eメールアーカイブのクラスタリングによる 開発コンテキストの可視化
大蔵 君治，川口 真司，飯田 元

SEC成果活用事例紹介
日本アイ・ビー・エム株式会社
CoBRA法を活用し、組織として統一された見積りモデルを実現

アングル
独立検証及び妥当性確認と形式手法がもたらすソフトウェア開発プロセスの高信頼化

海外レポート
米国主要機関の取り組み状況から見た 統合システムのディペンダビリティ確保に向けたエンジニアリング的課題について

組織紹介
ETIひろしま　Embedded software Technology Initiative

Column
ソフト業界の構造変化への対応を急ごう

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

SEC journal No.22
2010 年 10 月 25 日発行
第 6 巻第 3 号（通巻 24 号）
ISSN 1349-8622

巻　頭　言

情報処理推進機構理事長に就任して

独立行政法人 情報処理推進機構
理事長
藤江　一正

このたび情報処理推進機構の理事長を拝命いたしました。IPA はソフトウェア産業のみならず、IT 社会全体の発展に取り組んでいる、日本の IT 社会推進の中心的な機関です。大変な重責ですが、全力で職責を果たしてゆきたいと思います。

日本のソフトウェア産業の動向

ご承知のとおり、ソフトウェア産業は、他の産業と同じようにグローバルな競争が激化しています。さらに日本の得意とする組込み分野にも、低価格などを武器に、アジア圏をはじめとした諸外国の台頭により、日本の優位性は徐々に脅かされつつあります。しかし、2009 年度の貿易統計によれば、我が国の輸出総額の 50％強は組込みシステム関連製品が占めており、組込みソフトウェア開発力の一層の強化は、我が国の成長を支えるために極めて重要だと考えています。

第二点は、話題のクラウドコンピューティングです。これまでのように自らがすべてを「所有する」IT から、必要なときに必要なだけ「利用する」IT へのパラダイムシフトは、受託開発やソフトウェアパッケージ販売を主たるビジネスモデルとしてきた日本のソフトウェア産業界にとっては、サービス提供型、知識集約型へのシフトなどの変革が迫られています。そして、同時にユーザ側にとっても、特注開発・ベンダ依存型からパッケージ利用、サービス活用型へのシフトを一層加速することが必要になっています。今まさに、ベンダ、ユーザ両者にとって、これまでのソフトウェア開発、システム開発に対する考え方を大きく切り替える時期を迎えているのではないでしょうか。

第三点は、IT の活用による社会インフラの高度化の進展です。例えば、道路に沿って設置された交通情報システムと車両や車両相互を連携させることによって、安全でかつ効率良く、しかも環境負荷も低減出来る優れた ITS システムが日本ではすでに実証段階にあります。このような IT 活用を、道路だけではなく電力や水道をはじめとする社会システムに広げ、公共的サービスの高度化を進める実験が急速に立ち上がりつつあります。

このようなシステムは、単独の孤立したシステムではなく、組込みシステムと情報システムなど、異なるシステム同士が連携して機能する「統合システム」です。これまでは、組込みシステムと情報システムの分野は、それぞれ本来固有の目的のために設計・開発・運用され、独立に発展してきました。しかし、今後はこのような「統合システム」に向けて、相互の技術的な連携、人的な交流がますます必要になってくると考えています。

SEC の役割

新しい価値を作り出し、豊かで便利な生活を実現する中核はソフトウェアにあることは言を待たないと思います。一方で、それが私たちの生活の安心・安全を脅かすものであってはなりません。そのような観点からソフトウェアの企画・開発・運用・保守のすべてのライフサイクルに目配りをしながら、ソフトウェアエンジニアリングについて広く世の中に情報発信をしていってもらいたいと期待しています。

IPA 全体としては、SEC の活動と共に、情報セキュリティ、人材育成、システムの相互運用などの活動と合わせて、健全な情報社会の発展に尽力していきたいと考えております。

皆様の更なるご理解とご協力をお願い申し上げます。

新しい社会サービスの実現に向けて その課題と解決策を考える

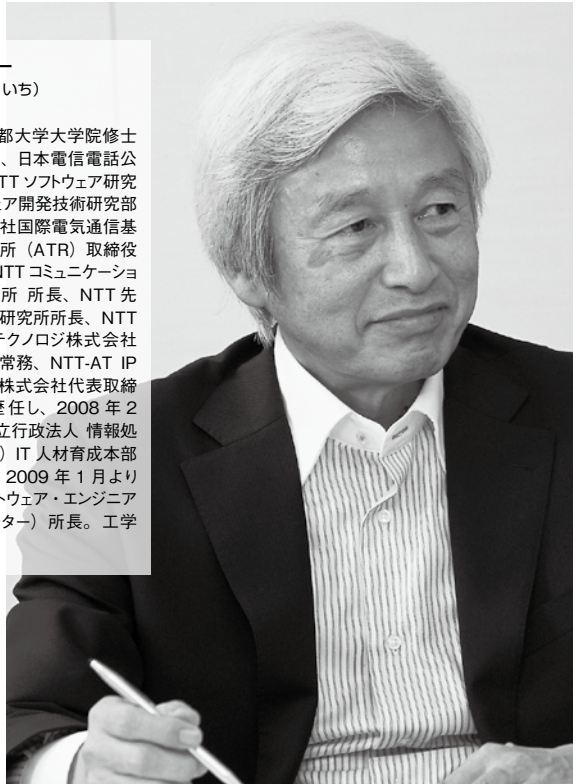
日本アイ・ビー・エム株式会社
執行役員　未来価値創造事業
岩野 和生

SEC 所長
松田 晃一

RFIDやセンサ技術とインターネットが融合し、森羅万象に関するデータを取得することが可能となっている。ITの進化を背景にエネルギーや水等、社会の物理インフラを最適化する動きが具体化している。しかし、それを実現していくために解決すべき課題も控えている。日本アイ･ビー･エム株式会社　執行役員の岩野和生氏に新しい社会サービスを実現するためにソフトウェアが担うべきこと、そして社会制度のあり方等について伺った。

松田：ご担当をされている未来価値創造事業とは、心躍るタイトルですが、具体的なイメージがつかみにくいので、実際には、どのようなことをされているのかご紹介ください。

岩野：日本アイ・ビー・エムでは、今まで、ハードウェアやソフトウェア、そしてサービスを提供してきましたが、それら個別の提供だけでは世の中から必要とされている価値を創り出すことは出来ないという考えに到達しました。そこで、2008 年に社長直轄で設置された組織が未来価値創造事業です。価値創造は特定の業種に限ったものではありません。この組織は業種をまたがって、また製品やソリューションもまたがって、新しい流れを作っていくことを目指しています。未来価値を創造していく流れの中で、今年、IBM は Smarter Planet というビジョンを掲げるに至り、スマートエネルギー等の推進に取り組んでいます。



松田 晃一
（まつだ こういち）

1970 年京都大学大学院修士課程修了後、日本電信電話公社入社。NTT ソフトウェア研究所 ソフトウェア開発技術研究部長、株式会社国際電気通信基礎技術研究所（ATR）取締役企画部長、NTT コミュニケーション科学研究所 所長、NTT 先端技術総合研究所所長、NTT アドバンステクノロジ株式会社代表取締役常務、NTT-AT IP シェアリング株式会社代表取締役社長を歴任し、2008 年 2 月 IPA（独立行政法人 情報処理推進機構）IT 人材育成本部長に就任、2009 年 1 月より SEC（ソフトウェア・エンジニアリング・センター）所長。工学博士。

松田：価値の創造には、IT が大きな役割を果たすと思います。従来、IT は主に企業におけるバックオフィスの効率化、合理化に用いられてきました。それが現在では、企業の価値創造や新商品の提供、新しいビジネスモデルの構築へと活用されています。しかし、社会システムにおける IT 活用の度合いは、このような企業の IT 活用に比べて格段に遅れています。その領域に IT の活用が広がれば付加価値のある新しい公共サービスを生み出せる可能性があるはずです。

岩野：Smarter Planet は、技術と社会的な必然性から自然に生まれてきたものです。技術的な観点で言うと、RFID 等の技術とインターネットが結び付き、森羅万象のデータや活動を補足出来るものです。2011 年には 1 兆個のデバイスがインターネットに接続され、また、集めたデータを処理する技術も非常に優れたものがあります。そうすると、森羅万象の活動を解釈し、目的や価値観に応じた最適なアウトプットを持つことが可能となると考えています。

松田：その最適化の内容はどのようなことを指すのでしょうか。

岩野：私たちは、物理インフラとデジタルインフラが融合する世界が来ると主張しています。物理インフラは水や交通、都市インフラのことを指します。これまでは、それらのインフラ全体に何が起きているかを把握出来ていなかったのですが、物理インフラとデジタルインフラを結び付けることによって、物理インフラ全体を最適化することが出来ると考えています。そうすると、漏水や交通渋滞等いろいろな社会的な損失や無駄を解消することが出来ます。

松田：いろいろなところからリアルタイムにデータが集まってくることによって、どこに損失や無駄があるかが把握出来るようになってきたわけですね。

岩野：そうです。また、単にデータを集めるだけでなく、データを整理するデータベース技術も進歩してきています。そして、最適化を実現するために重要なアルゴリズムも進歩しています。

グローバルに進む社会インフラのスマート化

松田：分析する技術の進歩を含めて、物理インフラを最適化するための環境が整いつつあるのですね。では実際に社会インフラのスマート化をどのように進展させておられるか、実例をお話しいただけませんか。

岩野：IBMがかかわっているリファレンスは500以上あります。同時に、世界で約100のプロジェクトが進行しています。例えば、イタリアの電力会社のエネル社では、5年間をかけて全2,800万世帯にスマートメータを配置し、15分間隔で電力使用データを集めています。その分析の結果、新しい取り組みが出来るようになりました。例えば、電力使用のピーク時間帯をシフトさせるためには、どのようなインセンティブを付けると良いのか、といった電力料金プランの実現です。実際、電力使用のピーク時間帯のシフトが可能となったことによって、将来的にはピークの発電量を200万kWほど下げることが出来ると予測されています。これは、およそ沖縄電力の発電設備出力に相当します。また、世界では盗電が多い国もありますが、スマートメータによってどこで盗電されているかも分かるわけです。

松田：電力使用のピーク時間帯をシフトさせる電力料金プランには、例えばどのようなものがあるのですか。

岩野：電力供給が危なそうになったとき、電力供給の停止に応じるといったプランがあります。

松田：その代わり、電気料金が安くなるということですね。水も環境問題の重要テーマです。

岩野：水の管理は世界的に重要なテーマです。IBMでは「アドバンスト・ウォーター・マネージメント」と呼ぶ取り組みを行っています。その1つとして、ニューヨークのハドソン川流域にセンサを張り巡らし、リアルタイムの情報を得て、降雨時における下流への影響を分析しています。また、アイルランド政府と共同で、ガルウェイ湾でも同様のことをしています。これらには、大量に流入するデータをリアルタイムに処理するストリームコンピューティング技術が、活用されています。

松田：漏水や盗水も問題のようですね。情報を集めて、ITを使って分析することによって、どこでどのくらい漏れているか、また盗まれているかも含めて、水道全体を合理的にコントロール出来るようになっていくのですね。

岩野：ビルのエネルギー管理もそうです。箱崎のIBM本社ビルは電気や水等のエネルギー使用量をモニタリングしエネルギー削減を推進した結果、1989年の竣工以来入居人数が倍増したにもかかわらず、エネルギー使用量は半分近くに削減しています。ニューヨークでは、2030年までに全体を環境都市とする目標を設定し、プランニューヨーク（PlaNYC）というイニシアチブを起こしています。そこでは地域全体ですべてのビルのエネルギー使用量を可視化してコントロールすることも目指しています。IBMは、その中で約1,400棟のビルを対象としたエネルギーコントロールに協力をしています。一般にビルディング・エネルギー・マネジメント・システム（BEMS）と呼ばれているものです。ところが、日本には中小のビルが70万棟あり、BEMSハードはある程度の導入は進みデータは取得しているのですが、データの活用はまだまだです。そこで経済産業省は、データ活用の促進のためにデータの標準化を検討しています。標準化が行われると、中小のビルからも情報を集めて管理することが可能になりますが、ここではクラウド技術が役に立ちます。

大切なのは住民が価値観を共有すること

松田：交通分野も環境問題の面からも重要ですよね。

岩野：ストックホルムにおいて、街の周りに設置したセンサとカメラにより、街に入ってくる自動車を認識します。混雑する時間に応じた渋滞課金を設けて交通量を制限し、公共の交通機関の利用を促進しています。それによって渋滞を減らし、深刻な大気汚染も防ぎ、また、CO_2を約14%ほど減少させることが出来ました。ストックホルムだけでなく、ロンドン、シンガポール等各都市にも展開しています。しかし、このようなインフラ整備には相応の財政負担がかかります。ストックホルムの場合は、費用を賄うための渋滞税を導入しようと考えました。住民にとっては大きな問題です。そこで、全市規模での社会実験を行った結果を公開後、住民投票を行い、その結果、やるべきだということになったのです。

松田：理解のある住民ですね。

岩野：ストックホルムのケースは、住民が価値観を共有し、かつある程度のコストを理解しているわけです。その効果として、良い環境社会環境を手に入れているわけです。そういうことが出来たストックホルムは、すごいと思います。

岩野 和生
（いわの かずお）

1975年東京大学理学部数学科卒業後、日本アイ・ビー・エム株式会社入社。1977年米国プリンストン大学コンピューターサイエンス学科よりPh. D. 取得。1995年から2000年までIBM Research東京基礎研究所所長。2000年から2003年までIBM T.J. ワトソン研究所勤務。2002年から2005年までIBMアジアパシフィック先進事業部担当。執行役員。2005年から2008年まで大和ソフトウェア開発研究所所長。2009年から未来価値創造事業担当。現在に至る。

松田：確かにそうですね。

岩野：新しい社会システムのように、全体系を捉える取り組みには、みんなで価値観を共有することが必要です。そしてもう1つ大切なことは、電話会社や電力会社等のように何十年も先を見越した計画を立て、インフラに対する投資を考えるということです。しかし、残念ながら最近ではインフラ投資に対する考え方が短期的になる傾向があるようです。20年後にハッピーになるからそれに備えましょう、という論理が通用しなくなっているように感じます。そうすると、どんどん社会的な競争力が失われてしまいます。

松田：日本でも、そういう状況が続いているように感じます。

岩野：何を引き換えに、何が得られるのかをきちっと説明するには力がいります。また、納得する力もいる。そこをうまく作ることが全体系の価値を生み出していくための成功要因だという気がします。

サービスを評価する第三者機関が求められる

松田：社会インフラをスマートにすると、無駄を無くし環境問題を解決することが出来る。その一方で、陰の部分を心配する向きもあります。日本人は、自分の家の電気の使用量がリアルタイムに知られることに不安を持つことでしょう。先程紹介していただいたイタリアの例では、そういう問題はどうなのでしょうか。

岩野：今は使用している量が分かるだけで、問題は無いようです。今後、何に電気を使っているかが分かるようになってくると、そういう問題が出てくる可能性があります。ただ、個人が特定出来ないようにデータを収集するプライバシー・プリザービング・データマイニングといった技術が進歩しています。

松田：そういうことは一般の人は理解出来ないですね。プライバシーを担保する技術を作ることに加えて、制度的な仕組みも作ることが必要でしょう。

岩野：制度的な面は非常に重要です。社会サービスにはお金がかかわることもあります。お金は円ではなくてエコポイントのようなものが登場するかもしれない。そのように、様々な形のトランザクションが登場してきたときに、それをウォッチして評価する第三者機関の役割が求められると感じています。そうして初めてみんなが安心して社会サービスを利用出来るわけです。そういう意味でSECは信頼が置ける第三者になり得る立場だと思います。

松田:なるほど。個人情報の管理については、プライバシーマーク制度がありますが、これから次々生まれる社会サービスについても、市民が利用しても「大丈夫」というお墨付きがあると安心ですね。

岩野：ITの世界はあまりにも速く進歩した結果、専門家以外には分からない状況になっています。ですから、技術も、その技術がもたらす意味も分かった上で社会サービスを判断する機能をどこかに委ねることが求められます。方法としては法律でレポートを義務付けたり、監査制度や認定制度を設ける等が考えられますが、今、そういう社会的な整備に取り組んでおくべきです。

松田：ITの進歩のためにも重要ですね。

岩野：私たちは、クラウドコンピューティングが社会サービスのインフラとして機能すると見ています。そうすると、クラウドコンピューティングで議論しているセキュリティやプライバシー、業務の標準化、トランザクションの保証方法というテーマが、新しい社会サービスの動きと密接に絡んできます。そこで、サービスレベルをきちんと管理するディペンダビリティを担保することが大切になります。

松田：クラウドコンピューティングの懸念事項の1つに、自分の情報を第三者に預けて大丈夫なのかという点があります。その解決への試みとしてサービス・レベル・アグリーメント（SLA）を宣言することで対応しようとする動きがあります。しかし、現状ではSLAとして宣言しているのはサービスのアベイラビリティのみです。それ以外に、情報をどのように管理しているか、あるいは障害が起きたときに何時間以内に通知するかといったことも含めてサービスレベルとして定義し、公開すべきでしょう。ユーザのクラウドベンダの選択時の判断基準になるでしょうし、また、定義したサービスレベルが実績として確保されているかどうかを第三者が確認し、その情報を公開するということも重要になると思います。

岩野：銀行にお金を預けることが当たり前のことになっているのと同じように、クラウドサービスによって情報を外部に預けることが当たり前になりつつある現在、それを担保する仕組みが求められています。しかし、日本の社会の特性として、ネガティブな面がクローズアップされ、それだからダメなんだという方向へ議論が進んでしまいます。対象が日本国内のみであればそのように議論がシュリンクしてもいいのですが、物理インフラとデジタルインフラをくっつける新しい社会インフラシステムというのはグローバルな動きです。しかも、それは日本の成長戦略にも大きくかかわっています。そういう意味で、ネガティブな側面があることを認め、将来に向けてどのようなことが必要になるのかをちゃんと伝えていく義務を果たしていくこと、質のいい議論を世の中に起こすことが今求められていると思っています。

世界に通用するコンポーネントの開発を目指そう

松田：社会インフラのスマート化は、環境問題へ大きな貢献が期待されますが、その中でITの役割は幾つかあると思っています。1つには、物の流れや人の動きで消費されるエネルギーから無駄を取り除くことがあります。例えば、配送に関する情報を整理して配送ルートを最適化することです。物流を情報の流れに置き換えることで使用するエネルギーを削減出来ます。

もう1つは、現在の環境の状況を「見える化」することです。環境問題は五感で分かるものではありません。CO_2や窒素酸化物等は人の目に見えないのでピンときません。ITを使って環境に関する情報を「見える化」して人に訴えるようにすることが大事だと思います。

岩野：それと、早い時期から生活者が常に環境のことを考えられるようにすることも大切です。私たちは九州大学等キャンパスのエネルギーマネジメントをやらせてもらっているのですが、それだけではなく、擬似キャップ・アンド・トレード（排出量取引）をすることによって学生の意識を高めることが期待出来ます。そうしたことを小中学校から体験して、小さいときから自分たちが社会的なことに関与しているという意識を醸成することが大事だと思います。

松田：小さいときから社会的な関与を体験し、実感した技術者が増えれば増えるほど、新しい社会サービスをITで生み出すこともそれを活用することも自然に可能となっていくことでしょう。

岩野：可能性は非常にあると思っています。私たちは、今までおつきあいの無かった業種の人たちとも協力関係を増やしています。すると新しいサービスが出てくるものです。例えば、産業廃棄物のクラウドインフラを作ろうと豊田通商様と研究を進めているのですが、これにより産業廃棄物の排出事業者や運搬業者、最終処分業者がすべてクラウドを使っていくことになるのです。そうすると、今までの産業関連図よりも大きな産業関連図が見える可能性があります。それによって経済行為の無駄が見え、そこに新しい中間業者が入ってこられる可能性があるわけです。

松田：クラウドにより、異なる産業がつながることによって、新しいビジネスが生まれるわけですね。

岩野：そうです。ある意味で情報のインテグレータやサービスのインテグレータが登場してくる可能性があるのです。ITの世界はこの10年でGoogleが飛躍したのですが、これからもGoogleが3つ4つポンと出てくるくらい大きく新しいサービスが出てくる可能性があると思います。

松田：新しいサービスの可能性、そのためのサービスインテグレータの登場は、全く同感です。しかし、日本人は単体の物を作ることは上手ですが、それをシステムにして大きなサービスとすることには弱いですね。ソフトウェアの話もそこにつながると思います。日本人は、テーマに対応したソフトウェアはきちっと作れるのですが、テーマ立案には弱い面を感じます。今のお話のシステム化あるいはサービス化とイコールの問題だと思っています。社会のスマート化はソフトウェアで実現されるものです。ハードウェアであるスマートメータだけがたくさんあっても実現出来るものではありません。ところが、そういうソフトウェアを立案出来る力が日本は弱い。そこを何とか出来ないか。装置や部品全体をシステム化するためにソフトウェアが重要であり、そこを何とかしたいと考えています。

岩野：同感です。ソフトウェアはどんどんモジュール化、コンポーネント化しています。そのモジュール、コンポーネントは世界的に通用するものでないと意味がありません。日本国内でデバイスに付随するエンベデッドソフトウェアを作ればいいというものではなく、世界の中でいいものを組み合わせていって、世界のミドルウェアに入っていくものを初めからデザインしないと結局は使われないでしょう。その戦場をソフトウェア開発者自身でデザイン出来るようになることが大切です。戦場を作られてからではもう遅い。

松田:そういう意味では、スマート・グリッドやスマートコミュニティ等の新しい社会サービスは新しい戦場を作っていると言えますね。

岩野：日本は、ブロードバンドも携帯電話も普及しているため、社会サービスをITから受け取ることに慣れてきているので、スマート化によって何が出来るかを考え、実践に移す土壌があると思います。ただし、下手をすると地方の自治体がソフト会社と一緒になって限定的な住民サービスをするということがスマートコミュニティです、となってしまう可能性があります。そうではなく、もっと大きな社会インフラサービスのコンポーネントを作って世界に展開するというストーリーがあると思っています。

松田：SECの組織は、エンタプライズ系と組込み系とに分かれて仕事をしてきたのですが、ここ1年、両者を1つにしたような統合プロジェクトを進めています。社会システムのスマート化は、技術的にはエンタプライズシステムや組込みシステムが連携したものですね。そういう領域を両方の技術を総合してサポートしようという趣旨でプロジェクトに取り組んでいます。

岩野：新しい社会サービスやクラウドにおけるサービスのアカウンタビリティが議論されていますが、今後、個々のデバイスや組込みソフトウェアの状態を標準化されたポリシーに則ってレポートが出来る機能が必要となります。その機能は、トータルでしかもグローバルにつながるディペンダビリティがあるものとしてデザインされる必要があります。そのような全体系を作っていくところにSECの役割があると思います。

松田：SECとして社会サービスの新しい展開をソフトウェアの立場から支援する活動に取り組んでいこうと思います。本日はありがとうございました。

文：小林秀雄　写真：越昭三朗

九州工業大学における パーソナルソフトウェアプロセス教育

ーソフトウェア品質向上のためのスキル修得ー

元九州工業大学　情報通信技術教育センター
教授
秋山　義博

九州工業大学大学院
情報工学研究院
片峯　恵一

九州工業大学大学院
情報工学研究院
梅田　政信

九州工業大学
名誉教授
橋本　正明

九州工業大学大学院
情報工学研究院
乃万　司

PSP※1は、Watts S. Humphrey博士が米国･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（CMU/SEI※2）において、ソフトウェアプロフェッショナルが身に付けるべきプロセス改善能力とスキルを明らかにし、その修得を可能にする教育プログラムとして開発したものである。九州工業大学大学院 情報工学府は、2007年から3年間、SEIと連携してこのPSP[HUMPHREY-4]トレーニングに取り組み、品質とエンジニアリングの改善能力の向上を実現した。この報告では、導入の狙いとアプローチの概要、PSPインストラクタの育成、大学院学生への教育の具体的アプローチ手法について述べる。また、研究室を構成する学生、研究員、教員による効果的なプロジェクトマネジメントへの期待についても述べる。

1 はじめに

ソフトウェアの品質と開発マネジメントの改善能力を高めることは、先進システム技術の実現を容易にし、その技術移転と新しい製品開発を確実にする。ソフトウェアプロセスはこの両面をカバーする重要な基盤であり、大学教育には産業界で定量的に実証されたベストプラクティスを導入し、これを習得した新しい人材を育成することが強く要望されている。

W. Humphrey 博士は、1980 年代後半に産業界のソフトウェア課題を調査し、SE がソフトウェアプロセス改善能力を身に付けることにより、ソフトウェア開発プロジェクトの多くの問題解決が可能になることを指摘して、まず個人レベルのベストプラクティスを学ぶための PSP トレーニングコースを開発した。以下に、この PSP の概要と PSP トレーニングの導入動向、九州工業大学での導入までの経緯、導入具体事例と成果、そして、PSP トレーニングと TSP※3 導入による PBL※4 や研究プロジェクトへの適用の可能性を述べる。

2 PSP とプロセス教育

2.1 PSP の概要

ソフトウェアの品質はそれに含まれる最低品質の部品により制限され、そのような部品は最低品質の個人プロセスを通して作成される。すなわち、個人のプロセス改善能力がソフトウェア品質に直接影響する。

PSP では、図 1 に示す 3 段階に沿ってエンジニアが基本的なプロセススキルを修得しプロセス改善能力を身に付けることを目的とする。第 1 段目は、コーディング、規模勘定、欠陥記録（作り込みと除去それぞれのフェーズ、タイプ、修正時間）の標準を定めて、規模測定、欠陥記録、簡単な時間と規模の見積り、プロセス改善提案の基本スキルを確実にする。これに加えて第 2 段目では、要求の実現に必要な部品の同定とそ

の規模と開発資源の見積りを履歴データから統計を活用して求め、開発計画を作成し進捗追跡を行う。第3段目では、品質見積りと品質計画を行い追跡する方法、レビュー、設計テンプレート利用と検証方法を学ぶ。最後に、全演習を通してパフォーマンス分析を行い優先度の高い改善目標の設定とその達成シナリオを定量的に明らかにする。ソフトウェアエンジニアリングスキルを品質マネジメントに生かすためのPSPフレームワークを図2に示す。

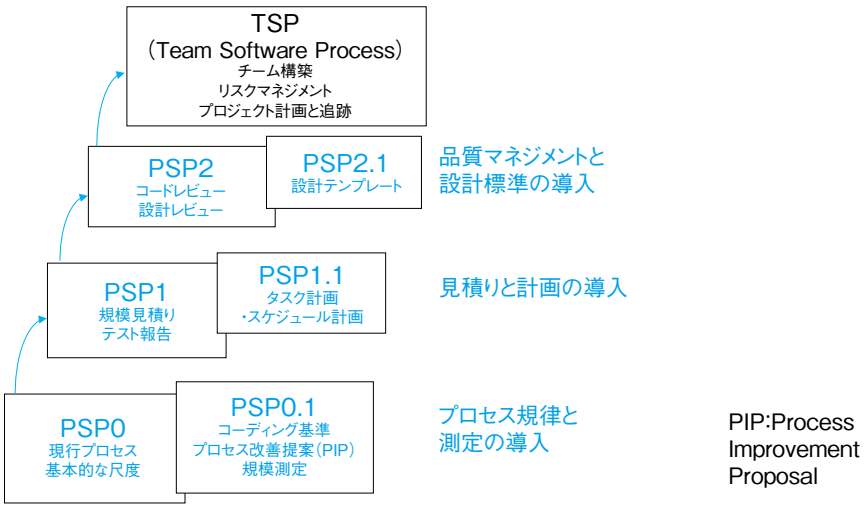


図1　PSPプロセス段階ステップ

2.2　プロセス教育の世界の動向

PSPトレーニングは、米国・カーネギーメロン大学[MSE]、米国・エンベロー大学[HIL2000]、米国・カルフォルニア大学[ISSHIKI2009]、メキシコ・モントレィ工科大学[BARRY2010]、南アフリカ・ウィツ大学[BARRY2010]、タイ・ナレスアン大学[TYLPDE]に、また産業界としては、米国、メキシコ、南アフリカ、日本、中国、ドイツ、エジプト、イギリス、ポルトガル、ブラジル、インド等に導入されている[SEI-PTNR]。国家レベルでは南アフリカのJCSE※5とメキシコのMexican TSP Initiative[HUESCA2010]が有名である。中国とインドでは米国ベンダがPSP/TSPを子会社関連企業向けに導入を進めてTSPをプロジェクトマネジメントに活用している[SEI-PTNR]。その代表的な企業としてはAdvanced Information Systems、Microsoft、EDS/HP、Motorola、ABB、BOSH他等が挙げられている[TSPSYMP]。

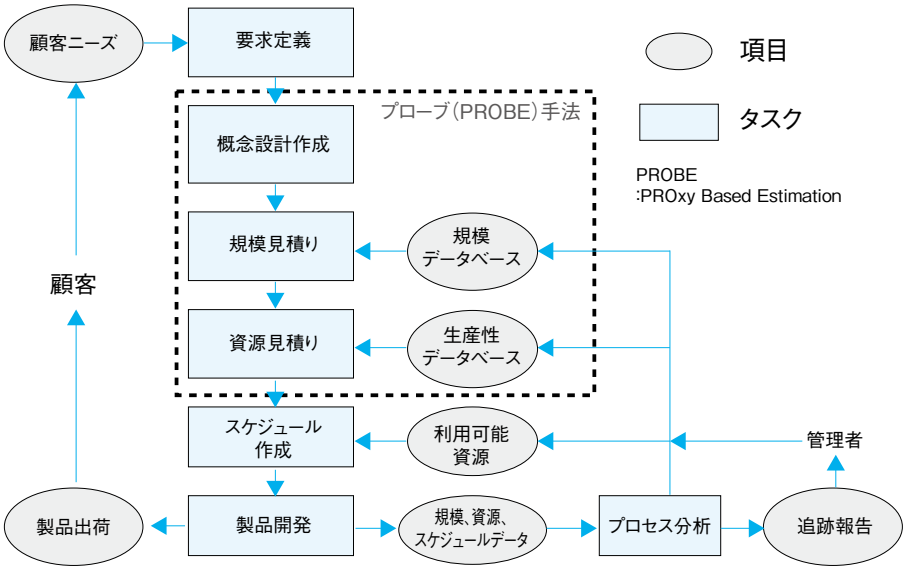


図2　PSPフレームワーク

脚注

PSP, TSPはカーネギーメロン大学のサービスマークです。
※1　PSP：Personal Software Process, パーソナルソフトウェアプロセス
※2　CMU/SEI：Carnegie Mellon University/Software Engineering Institute
※3　TSP：Team Software Process
※4　PBL：Problem Based Learningまたは Project Based Learning
※5　JCSE：Joburg Centre for Software Engineering

2.3 日本における PSP トレーニングの現状

W. Humphrey 博士が“A Discipline for Software Engineering”[HUMPHREY-1] を著し、この日本語訳を藤野喜一のリーダシップの下に松本・佐谷・砂塚・小宮山らが「パーソナルソフトウェアプロセス技法」として世に出した [HUMPHREY-1]。これを受けて創価大学でその勉強会が開始され、その後、早稲田大学での PSP ゼミへと発展した。

大学 1 年生向けに“Introduction to Personal Software Process”[HUMPHREY-2] が作成され、米国大学教育への導入が開始されたことを受け、これを PSP ネットワークが翻訳して「パーソナルソフトウェアプロセス入門」[HUMPHREY-2] を世に出した。これは、法政大学の社会人コースで使用された。

九州工業大学は、2007 年 3 ～ 4 月に希望教員向けにパーソナルプロセス規律を学びプロセス改善能力を修得するための SEI 認定の PSP コース“PSP for Engineers”を実施した。テキストは W.Humphrey 博士の最新著作“PSP A Self-Improvement Process for Software Engineers”[HUMPHREY-4] である。また、全国 8 大学 22 名の教員に対して SEI Faculty Workshop を提供し普及を狙った。続いて、SEI 認定インストラクタを 2008 年 1 月に 4 名（九州工業大学 3 名、会津大学 1 名）育成し、これを受けて大学院博士前期課程の学生向けに 2008 年度より PSP 教育を開始した。会津大学では“Process Week”のタイトルの下に 2009 年と 2010 年の 3 月に上級マネジメントとエンジニア向けセミナーを実施している。九州大学は 2009 年に PSP コースの試行を学生向けに開始した。産業技術大学院大学は SEI 提供のアカデミック版 PSP for Engineers コースを、九州産業大学と信州大学は手作り PSP コースを実施している。

企業としては SEI パートナーとして、富士フイルム株式会社、日立ソフトウェアエンジニアリング株式会社（現、株式会社日立ソリューションズ）、九州日立マクセル株式会社、他数社の名前が挙がっている。

3 PSP プロセス教育導入の経緯

3.1 CMU/SEI 連携計画

九州工業大学は、CMU/SEI との間で、大学院生と教職員がソフトウェア開発のプロセス改善能力を修得す

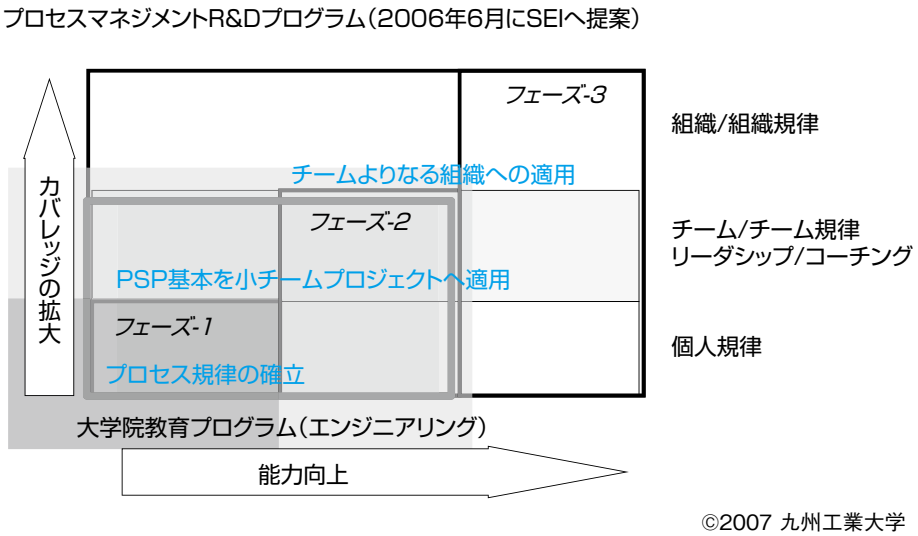


図3 九州工業大学ソフトウェアプロセスマネジメント導入戦略

ることを達成するための3段階アプローチについて、支援を受ける連携合意に至った（図3）。フェーズ1では個人レベルのソフトウェアプロセス教育と改善能力の実現、フェーズ2では小規模チームを構成しTSPによるプロジェクトの実施[HUMPHREY-3]、フェーズ3に至って研究室あるいは研究所全体のプロジェクトの効果的運営を可能にする計画である。これを確実にするために、SEIより講師を招聘して、プロセス原理・規律、自律チーム構築、システムズエンジニアリングとTSP、ファーカルティワークショップ、ソフトウェアプロダクトライン、超大規模システム等のセミナーを各フェーズの進捗に合わせて計画し、一層の深い理解を得ることを狙った。

3.2　導入までの経緯

2007年2月より準備を開始して、関心を持つ教員（九州工業大学より6名、九州大学より5名）に対して2つのPSPトレーニングコース（PSP-I[PSP-I]、PSP-II[PSP-II]）を同年3月と4月の1週間（計2週間）に集中的に実施した。同年5月にSEI所長Paul Nielsen博士が情報通信教育センター開所式で記念講演を“Software Engineering and the Performance Improvement-My Personal View”と題して行い、21世紀の重要分野におけるソフトウェアとエンジニアの役割と活躍の重要性を指摘し、この計画を参加者・関係者に周知した。そして、2008年1月にまず3名の教員が新たにSEI認定PSPインストラクタ資格を取得した。図4は、希望教員に対するPSPトレーニングの結果の一例である。テスト時に検出される欠陥密度について、狙い通り、受講開始時のソフトウェア品質のベストパフォーマンスよりも受講終了時のソフトウェア品質のワーストパフォーマンスが良くなるという素晴らしい成功を確かめた（図4での第2四分位及び第3四分位参照）。同図の横軸は実施順に課題番号を表し、縦軸は欠陥密度を表す。トレーニングを受けた学生のパフォーマンスもこの傾向に従って達成されている（後述）。この結果は、その後の統合とシステムテストのワークロードが大幅に減少することを示唆している。

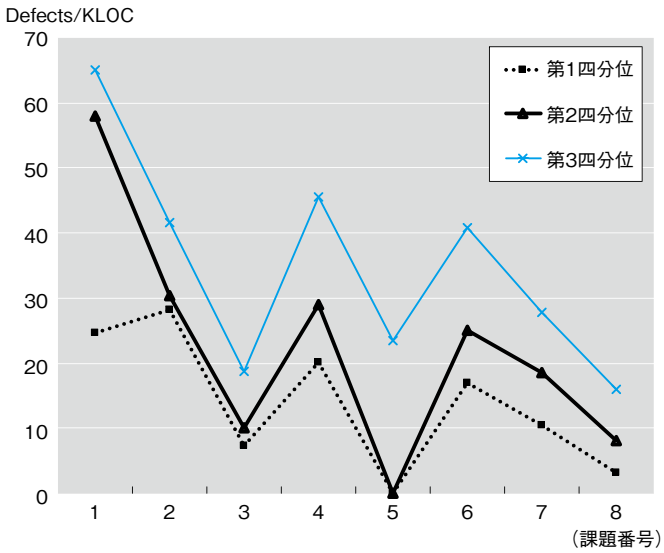


図4　九州工業大学教員によるテスト欠陥密度の改善

4　PSPコースの運用

4.1　PSPコースの位置付けと特徴

九州工業大学のPSPコース（以下、とくに混乱の無い限りPSPコースと呼ぶ）は、大学院博士前期課程の学生を主な対象として、SEI認定コース“PSP for Engineers”を実施するものである。受講対象となる学生の多くは、学部教育を通して、プログラミングやデータ構造とアルゴリズム、ソフトウェア設計等の情報処理に関する基礎教育を受けている。PSPコースの目的は、この基礎教育の上に、ソフトウェア開発のための実践的なプロセス教育を施すことにより、プロセス改善に必要な知識とスキルとを習得させ、専門家としての自覚を持った情報通信技術者を育成することにある。

PSPコースの修了者にはSEI認定コース修了証が授与される。従って、前期課程修了要件としての単位が得られるだけでなく、将来社会人になって後に適切に経験を積んだ暁でのPSPインストラクタ資格取得にも直接的に生かすことが出来る。

4.2 PSP コースの実施方法

SEI 認定のコース要件を満たしつつ、大学院科目として PSP コースを導入するために、筆者らが検討を要した主な課題は、次の通りである。

(1) 授業の構成について

PSP for Engineers は、おのおの 5 日間をかけて実施する 2 つの PSP トレーニングコース（PSP-I、II）からなる。PSP-I、II は、午前中の講義の後に演習課題が毎日与えられ、最終日にはレポート課題がそれぞれ課せられる。

一方、大学院のカリキュラム上では、PSP-I、II にそれぞれ 22.5 時間（1.5 時間× 15 コマ）の授業時間が確保されている。この限られた授業時間の有効活用の方法が課題となる。

(2) プログラミングスキルの不足について

最初の演習課題で、Linked List 程度の簡単なデータ構造とその操作を設計し実装出来るプログラミングスキルが要求されている。PSP コースの受講生の一部は、情報処理の基礎科目を 1 年生～ 2 年生で学んでから長期間遠ざかっているため、プログラミングスキルが不足している恐れがある。

これらの演習課題に対して、当初から受講生のプログラミングスキルや演習時間等を予想することは難しいため、課題の進捗やプロセス改善の結果を見ながら実施方法の改善を試みた。表 1 は、2007 年度から 2009 年度までの授業実施方法の概略である。

2007 年度の著者らのインストラクタ資格取得直後に、SEI の“PSP for Engineers”コース実施スケジュールに倣って、PSP-I を 5 日間連続して実施した。この方法では、午前に講義を受け、続いて午後に 1 課題を完了させることが 5 日間連続するため、コース完了がかなり困難な受講生が見られ、全 4 課題を完了出来た受講生は約 43% にとどまった。そこで、これに続く PSP-II においては、講義を 1 日おきに実施し、演習期間を 1 日以上にわたって確保出来るように配慮した。

2008 年度は、完了率の向上には十分な演習時間の確保が必要と考え、通常の科目同様に講義を 1 週間ごとに実施し、7 日間の演習期間を取れるように配慮した。また、プログラミングスキルの不足も見られることから、受講希望者に対して設計と実装の事前演習課題を

表1　PSPコースの実施方法

年度	内容	実施方法
2007	PSP-I	2 コマ / 日の講義を連続 5 日間に集中して実施
	PSP-II	2 コマ / 日の講義を 1 日おきに集中して実施
2008	PSP-I	4 コマの事前学習後に、2 コマ / 2 日の講義を 1 週間おきに実施
	PSP-II	2 コマ / 日の講義を 1 日おきに集中して実施
2009	PSP-I	2 コマ / 日の講義を 1 週間おきに実施し、数日後に計画のレビューを 1 コマ実施
	PSP-II	2 コマ / 日の講義を 1 週間おきに実施し、数日後に計画のレビューを 1 コマ実施

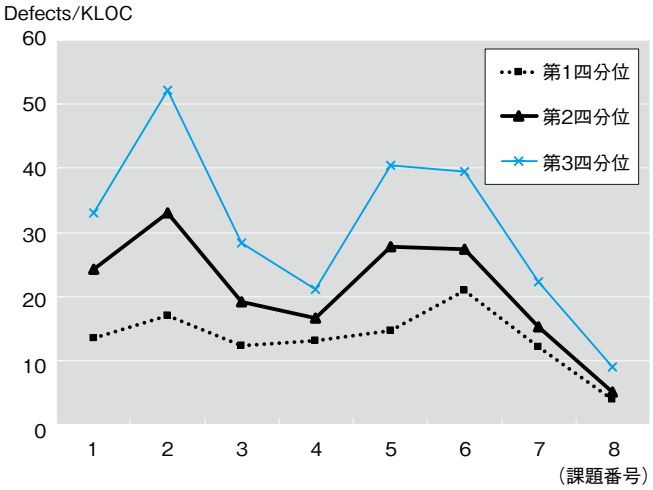


図5　学生によるテスト欠陥密度の推移

与え、全体的な底上げを図った。この結果、完了率がPSP-Iでは75%、PSP-IIでは100%にそれぞれ向上した。しかし、事前演習時にプログラミングスキル不足を自覚した受講生が履修を断念したことも完了率が高まった一因とも考えられる。

2009年度は、受講希望者を可能な限り受け入れた。事前演習課題に代えて、講義後数日おいて、要求仕様の理解確認、概念設計の不備や見積りプロセスの正しい適用を確実にするために、計画レビュー時間として1コマ設けた。手戻りによる意欲低下や演習断念を低減するためである。このような実施方法の改善により、事前演習による実質的なふるいが無かったにもかかわらず、PSP-Iの完了率は約80%に改善された。

4.3 結果と評価

2007年度から2009年度にPSPコースを受講した博士前期課程の学生16名のデータを用いて、ソフトウェア品質と生産性を議論する。また、学生へのインタビュー結果も掲載する。

ソフトウェア品質の評価には、単体テストにおける欠陥密度を用いるのが良く、1,000LOC※6当たりの欠陥数で表す。図5にテスト欠陥密度の四分位数の推移グラフを示す。図中、横軸は課題番号（1～8）を表し、縦軸は欠陥密度を表す。第1四分位数、第2四分位数(中央値)、第3四分位数共に、後半に進むにつれ減少傾向がはっきり現れており、ばらつきも少なくなっている。また、コース当初のソフトウェア品質のベストパフォーマ（第1四分位数）の欠陥密度より、コース後半のソフトウェア品質のワーストパフォーマ（第3四分位数）の欠陥密度の方が低い。課題5からコードレビュー、課題7から設計レビューを本格的に導入しており、テストの欠陥密度は、設計レビューの効果が高く出ていると考えられる。図6に、ソフトウェア開発全体におけるテスト時間割合の四分位数の推移グラフを示す。この図においても、図5と同様の成果が見られる。PSPコースが大学教育においてもソフトウェア品質向上に十分有効であることが分かる。

図7に生産性の四分位数の推移グラフを示す。生産性とは、ソフトウェア開発の全工程の1時間当たり生産されるプログラム規模（LOC数）で表す。図の縦軸

脚注

※6 LOC：Lines of Code

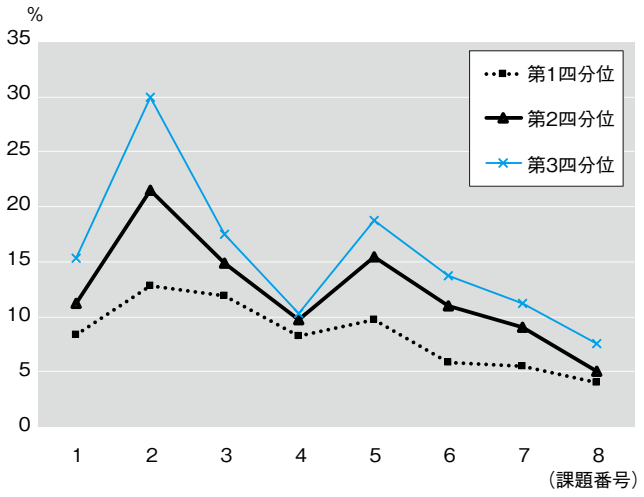


図6 学生によるテスト時間割合の推移

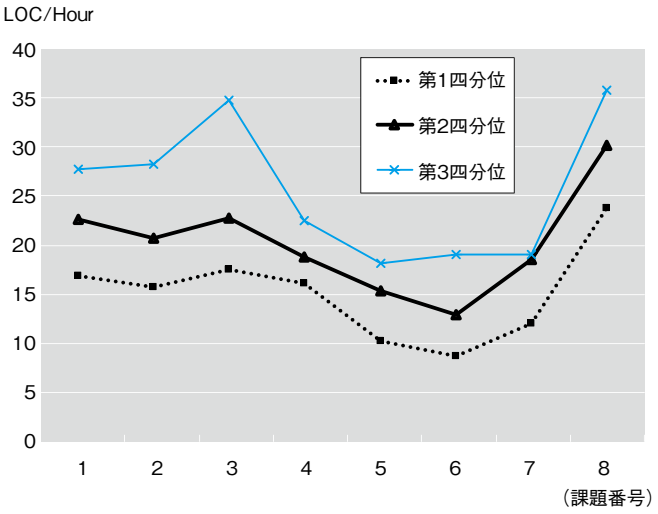


図7 学生による生産性の推移

は生産性を示す。この図から生産性は、課題6まではプロセスが増加するため低下傾向を示すが、その後は増加傾向を確実に示す。これは、後半ではプロセスが安定し同時にテスト時間が減少したためと考えられる。SEIが提供している過去のPSPデータによると、生産性はほとんど変化しないという結果が出ているが、このコースの受講開始時にはプログラミング経験が少なく、後半では規律正しいソフトウェアプロセスを身に付けるため生産性が向上したと考えられる。

また、PSPコースの受講生にインタビューを行ったところ、以下のような意見が得られた。

・ソフトウェアの計画立案の方法を理解出来た
・設計とレビューの重要性を理解した
・欠陥ログ情報から作り込む可能性のある欠陥を予測出来る。それから欠陥数を減らすことが出来た
・欠陥発生による時間のばらつきの減少により、計画時間通りにソフトウェアを開発出来た
・PSPコースを受講していない通常の学生との差が理解出来た
・PSPコースで学習した概念設計や見積り法を学会の論文作成時に適用して、その効果を確認出来た

以上のように、学生自身がトレーニングコースを体験することにより、おおむねPSPコースの意義を理解出来たと考えられる。

5 PSPと教育、研究との関係

5.1 PBL型教育との関係

(1) PBL型教育の現状

プロジェクトの体験が主であり、企業のソフトウェア開発の現場に類似した環境を体験出来る。そのメリットは、受講生に観察眼があれば、プロジェクトの問題を認識出来る。しかし､その解決策は与えられないので､デメリットは、未熟練の受講生が企業のソフトウェア開発現場の不合理を受け入れる恐れがある。

(2) PBL型教育におけるPSPの活用

要求分析とそれを満足するアーキテクチャ設計はチームで順序立てて遂行する必要がある。ここにPSPで学んだスキルと改善能力が使われる。各エンジニアのエンジニアリングスキル、生産性、品質パフォーマンス等の指標が他のエンジニアとは大きく異なることを認識した上でチーム戦略と順序立てのチームプロセスを作成する。これらはすべてPSPスキルにより可能になる。更に、システム問題がプロセス問題と渾然一体となって現れることが多い近年のプロジェクトにおいて、その切り分けが出来ない問題は深い。その切り分けのスキルを養うためにもPSPによってプロセスの本来の姿を教育しておくことは、PBL型教育にとって重要である。

(3) PSPを活用したPBL型教育

上記の理由により、PBL型教育に入る前に、PSPを訓練し、TSPも最小限経験しておくことが重要である。それによって、企業のソフトウェア開発現場の現状を批判的に見て改善提案の出来る学生を輩出することが望まれる。

5.2 学部教育との関係

PSPについて見ると、プログラミングとソフトウェアエンジニアリングを修得していればPSPトレーニング受講は可能なので､学部の高学年においてPSPトレーニングを実施することが望まれる。これは、単にソフトウェア開発のスキルを修得するだけにとどまらず、知的作業の多い現在の知識社会におけるエンジニアが身に付けるべきスキルとして必須である。

5.3 研究活動との関係

国際競争の激しい業界では、企業の研究開発に早くからプロジェクトマネジメントが導入されている。大学における研究プロジェクトでは先進的あるいは未解決の研究テーマに取り組むことが多い。学生を含めた

要員の入れ替わる場合や経験やエンジニアリング能力に大きな格差が認められる場合等が起こり得るので、作業の段取りや順序やその遂行のマネジメントが重要で、そのためのプロセスを実装することが重要になる。とくに、研究においては、システムや要素技術が研究対象となるので、プロトタイピングやアジャイル等の個々のステップに、PSP の考え方を導入することが望まれる。

5.4　業界や行政との関係

PSP と TSP を企業に導入するのに、訓練のコストと期間を確保する困難さが企業側に挙げられている。今回は、九州工業大学大学院 情報工学府の科目として PSP と TSP を導入し成果を得たが、今後は大学と産業界と行政とがますます連携して、国内導入を加速する施策が望まれる。

6　まとめ

今、CMU/SEI が提供している PSP トレーニングが、産業界において SE のソフトウェアプロセス改善能力を高めている。九州工業大学大学院 情報工学府において、2007 年から 3 年間、SEI と連携してこの PSP トレーニングを大学院生への教育に取り入れ、改善能力の大幅な向上を実現した。この報告では、その導入戦略と PSP 教育のコア教員育成の重要性、PSP トレーニングの導入アプローチの概要、その具体的アプローチと成果等を紹介し、PBL や今後への期待を述べた。確実に教育とトレーニングを行うこと（PSP コース実施プロセスがプロセス改善能力向上を達成する）が出来れば、期待する成果は得られることが分かる。品質向上を図ることは決して新たな投資を必要としない（Quality is Free!）。大学におけるこのような教育が着実に行われることによって、プロセス規律を身に付けた多くのエンジニアや若い世代が生まれることが期待される。

謝辞

本稿は、文部科学省「先導的 IT スペシャリスト育成推進プログラム」の「次世代情報化社会を牽引する ICT アーキテクト育成プログラム」（平成 18 ～ 21 年度，代表校：九州大学）によるものです。

参考文献

[BARRY2010] Barry Dwolatzky, Wan Peng Ng, Rafael Salazar Chavez : Process Improvement on a Regional Scale, Keynote Panel of SEPG-NA, 2010

[HIL2000] Thomas B. Hilburn : Teams need a process, ACM SIGC SE Bulletin, Vol. 32, Issue 3, 2000

[HUESCA2010] Agustin de la Maza Huesca : Best Practices for TSP Implementation on Outsourced Application Development Projects, Keynote of SEPG-NA, 2010

[HUMPHREY-1] Watts S. Humphrey : A Discipline for Software Engineering, Addison–Wesley, 1995（邦訳：松本 正雄 監訳，ソフトウェア品質経営研究会訳：『パーソナルソフトウェアプロセス技法』，共立出版，1999 年）

[HUMPHREY-2] Watts S. Humphrey : Introduction to Personal Software Process, Addison-Wesley, 1997（邦訳：PSP ネットワーク訳：『パーソナルソフトウェアプロセス入門』，共立出版，2001 年）

[HUMPHREY-3] Watts S. Humphrey : Introduction to Team Software Process, Addison-Wesley, 1999

[HUMPHREY-4] Watts S. Humphrey : PSP A Self-Improvement Process for Software Engineers, Addison-Wesley, 2005（邦訳：秋山 義博 監訳，JASPIC TSP 研究会訳：『PSP ガイドブック ソフトウェアエンジニア自己改善』，翔泳社，2007 年）

[ISSHIKI2009] 一色 浩一郎，松田 晃一：所長対談 米国から見た日本の IT 教育と CIO の役割，SEC journal, No.19, Vol.5, No.6, pp.340-347, 2009

[MSE] Software Engineering Master Programs : http://mse.isri.cmu.edu/software-engineering/web1-Programs/MSE/index.html

[PSP-I] SEI Courses : PSP for Engineers – Planning（SEI の TSP パートナーが提供可能）

[PSP-II] SEI Courses : PSP for Engineers – Quality（SEI の TSP パートナーが提供可能）

[SEI-PTNR] http://www.sei.cmu.edu/partners/directory/

[TSPSYMP] Software Engineering Institute : http://www.sei.cmu.edu/tspsymposium/{2006, 2007, 2008, 2009}/

[TYLPDE] http://www.thailandspin.com/ThailandSPINCalendar/tabid/63/articleType/ArticleView/articleId/45/Default.aspx

信頼性の高い組込みソフトウェアを開発するには

東海大学大学院　組込み技術研究科
准教授
山浦　恒央

電子製品、家電品が発達した現在、マイクロプロセッサを搭載していない製品を探すのは極めて難しい。社会の末端まで組込み系の製品が浸透しており、大規模な汎用システムと共に、社会の基幹機能を担っている。

ソフトウェアは、ハードウェアと共にコンピュータシステムの根幹をなすが、ソフトウェアの生産性、開発方式、品質制御技法は30年前からほとんど進化しておらず、コンピュータシステムの革命的な処理性能、機能、低価格化は、ハードウェアの驚異的な技術進化に全面的に依存している。

なぜ、ソフトウェア開発の技術革新、生産性、品質制御技術は過去30年間停滞し、今後も改善を望めないのか？　組込み系のソフトウェアのテストは、なぜ、難しいのか？　SEC journal No.21の『特別セミナー「組込みソフトウェアの信頼性を考える」より』[SEC journal21]、の続編として、本稿では、まず、他の工業製品には無いソフトウェアの特徴を挙げて、開発の難しさを分析し、続いて、テストや品質制御の難しさや限界を述べる。最後に、一般のソフトウェア開発より、更に条件が厳しい組込み系の開発、品質制御について分析する。

1　ソフトウェア開発

1.1　ソフトウェア開発の難しさ

有史以降、人類がこれまで作った物の中で、圧倒的に複雑、巨大で、開発が困難なのはソフトウェアと言っても過言ではない。ソフトウェア開発がこれほど難しい理由として以下の３つが挙げられる。

(1) プログラムは、規模が大きい

ソフトウェア開発が困難な最大の理由は、規模の大きさにある。例えば、一人乗りの舟を作れと指令されても、材木を買ってきてロープで縛るだけで完成する。流体力学も船舶工学も必要ない。しかし、千人乗りの大型客船を建造する場合、最高峰の技術力、設計のセンス、製造実績が不可欠となる。どんな分野でも、量が多く、規模が大きくなると、作り方は急激に難しくなる。

(2) 制約となる自然界の法則が無く、自由に作れる

電器製品を設計する場合、オームの法則やキルヒホッフの法則をはじめ、いろいろな法則でがんじがらめになる。同様に飛行機を設計する場合も、流体力学の法則、ニュートン力学を遵守しなければ飛ばない。

工業製品の中で、ソフトウェアの設計は例外的であり、縛られる法則が無い。法則に縛られず自由に作れるのは、良いことか？　束縛が厳しくなるほど、開発側の考え方が同じになり、画一的な作り方になる。逆に、完全に自由に作って良いとなると、設計上の定石が存在しなくなる。

「何でもよいので、『喜び』を表せ」と言われた場合、絵画、小説、写真、音楽、芝居等いろいろな表現方法がある。文字通り芸術の世界であり、喜びを表す「開発プロセス」は存在しない。逆に、「自分が経験した『楽しい話』を400字で書け」と制限を付けると、非常に書きやすくなる。

「自由すぎるのは不自由」なのだ。あらゆる法則や定理に縛られているハードウェアが革命的な進歩を遂げ、完全に自由なソフトウェアは、昔から生産性が向上せず、生産技術も進化しないのは、これが原因の１つである。

(3) ソフトウェアは目に見えない

高層ビルは、手で触れて目で見えるため、作る大変さを容易に推察出来る。一方、プログラムは目に見えない。手でも触れず、大きさ、重さを感じることは出来ない。

目に見えないため、ソフトウェアシステムの全体像を簡単には捉えられない。プログラム構造も見えないため、プロジェクトで開発中のソフトウェアが、シンプルで整然とした構造か、スパゲッティ的に乱れた構成かは、ドキュメントから推察するしかないのだ。

1.2　ソフトウェアと長編小説

ハードウェアの生産性、性能、機能は、過去 30 年間で革命的に進歩したのに、なぜ、ソフトウェア開発の生産性や品質は飛躍的に良くならないのか？　上記の３要素と、ソフトウェアが「命令文の集合」であることを考えると、現実の作業で、「ソフトウェア開発」に一番近いのが「長編小説の執筆」と筆者は考えている。

2008 年は、紫式部が書いた世界初の長編恋愛小説、『源氏物語』の千年紀だった。『源氏物語』は、400 字詰め原稿用紙に換算すると、2,500 枚の大作である。取材をし、小説の構想を練り、登場人物のキャラクタを考える作業も含めて、1 日平均 10 枚書けるとすると、250 日かかる計算になる。紫式部から 1,000 年後、執筆技術やワープロ等のツールが進歩しても、現代の長編恋愛小説の名手、例えば、林真理子や山田詠美が同等の小説を 125 日で書けるとは思えない。30 年後も変わらないだろう。

2 ソフトウェアのテスト

2.1 完全テストの不完全性

図1は、テストに大量の時間がかかることを例示するため、昔からよく使う流れ図である。図の中央に判定文が3つあり、全部で5通りの経路に別れる。ループ回数が0～18回の場合、取り得る経路は 4.77×10^{12} もある。1本の経路の実行に1秒しかかからなくても、完全テストと言われている「全経路の実行」には10万年かかる。

このプログラムは、100ステップ以下で書けるし、1時間もあれば簡単にコーディング出来るが、全経路を完全にテストをするには10万年かかる。また、テスト設計書は、要求仕様を基に設計するため、機能の漏れがあると、テスト設計書通りにテストを実行しても、機能の漏れを検出出来ない。すなわち、10万年かけて全経路をテストしても、仕様の漏れという致命的なバグは摘出出来ない。

以上から、①全経路テストは理論的には可能だが、実践的には不可能であり、②テストだけでバグを検出するのは実質的に不可能、の2点が分かる。

2.2 ソフトウェアのバグと犯罪

ソフトウェア開発において、品質とバグの関係は単純ではない。高品質ソフトウェアとは、必ずしもバグがゼロのプログラムではない。例えば、10,000ステップ以上のプログラムをバグ無しで作るのは、ほぼ不可能だし、逆に、バグがあっても、高品質のプログラムはいくらでもある。

高品質ソフトウェアの開発という「永遠のテーマ」に正面から取り組むのは、対象が漠然としているため、簡単ではない。こんな場合、強力な「解決ツール」になるのが「アナロジ」だ。現実に存在する別の単純な物やモデルに置き換え、それを分析して本体の課題や問題点を解決する。

2.3 犯罪とバグの類似性

筆者は、ソフトウェア開発の課題や問題点に取り組んで30年以上になるが、いつも思うのは、「ソフトウェアのバグは、社会の犯罪に似ている」ということだ。例えば、犯罪には下記の特徴がある。下記の「社会」を「ソフトウェア」に、「犯罪」を「バグ」に置き換えても、成立する。

$5+5^2+5^3+\cdots\cdots+5^{18}=4.77\times10^{12}$

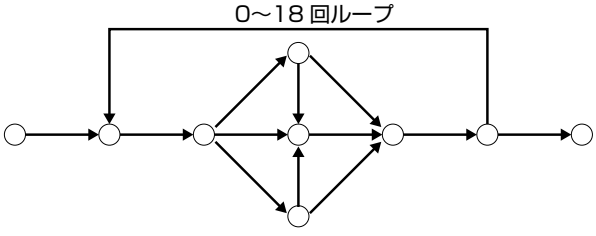


図1　完全テストと、テスト経路の数

① ある規模以上の社会には、犯罪がある

数人しかいないと犯罪は起こらないが、何万人かを超えると、犯罪の発生を防ぐことは不可能になる。同様に、10ステップ程度のプログラムをバグ無しで作ることは可能だが、100KLOC[※1]のプログラムでは、まず不可能だ。大規模プログラムの開発では、「バグはある」とのリスク管理的手法で対策する必要がある。

② 犯罪には、駐車違反レベルから、社会全体の破壊までいろいろある

バグがシステムに与える深刻度は、メッセージの誤字脱字のような軽微なものから、システムを破壊する深刻なバグまで、いろいろある。

③ 犯罪を無くすための予算、時間、人員は限られている

2.1節に示したように、ソフトウェアの開発では、作成より、テストや検証作業にはるかに多くの時間が必要となるが、現実のプロジェクトでは、多くても開発時間の半分程度しかテストやデバッグに割けない。少ない時間で、品質を確保する方法が重要になる。

④ 犯罪があっても、社会は正常に機能する

例えば、NASA[※2]のスペースシャトル制御プログラムのような高品質プログラムは、バグがあっても正常に機能する。逆に、バグがあってもソフトウェアが正常に機能するよう、リスク管理的に品質管理を進めねばならない。

⑤ 犯罪の検挙だけでは、犯罪を撲滅出来ない。犯罪防止の教育、犯罪の原因を取り除く等の予防対策が必要

これは、ソフトウェアの品質管理での永遠の真実である。開発時に、まず、バグを作り込まない工夫をし、その網を潜って入り込んだバグをデバッグやテストで摘出するのが本当の品質管理と言えよう。「作ってから、後でまとめてテストでバグを取る」方法では、時間がいくらあっても高品質ソフトウェアの開発は期待出来ない。

2.4 テストと犯罪の検挙

犯罪の「取り締まり」は、テストやデバッグに相当する。犯罪を検挙する場合、犯罪が集中している可能性の高い場所を集中的にパトロールする必要がある。この「目的意識」が非常に重要で、密輸を取り締まる場合、幹線道路から1本内側へ入った道路をパトロールしても、駐車違反は大量に検挙出来るが、密輸を挙げることは出来ない。ましてや、

脚注

※1　KLOC：Kilo Lines of Code
※2　NASA：National Aeronautics and Space Administration, アメリカ国立航空宇宙局

漫然とパトロールしていては、犯罪を検挙出来ない。

テストやデバッグでも目的意識は極めて重要で、①まずどんなバグが発生し得るか想定し、②そのバグを摘出するためのテスト項目を設計しなければならない。

有力なテスト技法に、「境界値分析」がある。これは、要求仕様での境界値、例えば、「データが 1,024 件を超えた場合、磁気ディスクを使用する」と定義してあれば、1,024 件と 1,025 件をテストする方法である。この技法は、「境界上にはバグが最も多く存在する」という経験則を基にしてテスト項目を設計する手法であり、目的とするバグを明確に意識している。

この反対が、いわゆる「モンキーオペレーション」だ。これは、適当にキー入力してはエンターを押すという動作を繰り返す技法である。このランダムテストは、操作が派手でバグを叩き出せそうに見えるが、摘出するバグを想定していないため、効率の良いテストは出来ない。

「検出するバグを想定する」を裏返すと、「想像出来ないバグは摘出出来ない」ことになる。犯罪でも、「想像を超えた犯罪は、犯罪の被害を受けていることに気がつかない」。想像出来ないバグは、眼の前にあっても、見えないのだ。「ソフトウェアは、開発するよりテストするほうが格段に難しい」理由の 1 つがこれである。

3 ソフトウェアの開発と品質

テストには、理論的に開発よりも、はるかに多くの時間が必要だが、現実の開発プロジェクトでは、多くても開発時間の半分程度しかテストやデバッグに割り当てられない。少ない時間で、品質をどう確保するかが重要になる。ソフトウェアの品質制御の基本プロセスを以下に示す。

(1) バグを作り込まない開発プロセス

開発とテストは一体であり、バグを作り込まない開発プロセスを導入する必要がある。要求仕様フェーズが終わったら、仕様を検証してから設計フェーズに移る。設計書をテスト、修正してからコーディングフェーズに入ることが必要である。それでも潜り込んだバグを実機上でテストして摘出する。この意味で、「設計ドキュメントの作成を必要最低限に抑え、早くコーディングし、バグはテストフェーズでまとめて摘出する」というアジャイル系の開発方法には、筆者は、品質の不安を覚える。

(2) リスク管理に基づいた品質管理

完全テストが不可能であり、テストをする時間も十分に取れないならば、バグを完全にゼロにする「建て前論的」な品質確保プロセスではなく、重要な機能は十分時間をかけてテストし、重要度の低い機能は、テストをある程度割愛するリスク管理的な戦略を導入する必要がある。このためには、上流工程の要求仕様定義フェーズで、機能の重要度を、例えば、「重要」「普通」「軽少」に分類し、リスクに基づいて開発と品質保証を実施せねばならない。

(3) デスマーチプロジェクトとの兼ね合い

必要な時間の半分以下しか開発期間を割り当ててもらえない開発プロジェクトを「デスマーチ」と呼ぶ。現代のソフトウェア開発は、ほとんどがデスマーチ化しており、デスマーチへの対処法が開発での最重要テーマとなる。

デスマーチを生き抜くには、プロジェクト管理者は定期的に、①残作業量を見積り、②現在の開発資源（開発人員、出荷までの日数）での出荷時期を予測し、③出荷日に間に合わない場合、機能の削減、品質の劣化、出荷日の延長等の対応策を動的に取らねばならない。万一、品質劣化を採用せねばならない場合、品質の現状を具体的な数値で把握し、また、その製品ドメイン（例えば、携帯電話）の最低限の品質レベルに達していることが必要である。

(4) 再利用による品質確保

「再利用」は、既存のソフトウェアをコピーアンドペーストし、必要な箇所を変更する開発プロセスである。再利用は、生産性向上を意図して導入する場合が多いが、品質と性能の確保のためにも採用すべきだ。

4 組込み系ソフトウェアの開発と品質制御

4.1 性能 vs 保守性

組込み系ソフトウェアの目的は、携帯電話、デジタルカメラ等、特殊なハードウェアを制御することにある。組込み系ソフトウェアは、動作環境、開発方式とも、汎用ソフトウェアの開発に比べると、極めて特殊だ。

組込み系システムは、ハードウェアから受ける制限が非常に多くて強い。販売台数が多くなるほど、コスト削減のため、ハードウェアからの制限が強くなる。また、MPU とメモリから受ける「縛り」により、処理方式と、設計技法は大きな悪影響を受ける。

(1) 処理方式の制限

組込み系製品では処理性能が非常に重要であり、リアルタイム性を確保するために高速処理が必要な場合、MPU の観点では、コーディングステップ数やメモリ占有量を減らすより、実行ステップ数を可能な限り短くしなければならない。一方、組込み系のソフトウェアでは、メモリ内に全プログラムを格納するため、実行ステップ数より、コーディングステップ数を短くする方が良く、MPU の要求と背反する。組込み系ソフトウェアの基本は、メモリと

MPU の背反する要求を同時に満足させることであり、「小さく早く」作らねばならない。

(2) 設計技法の制限

50 年前のプログラミングは、処理速度を上げ、メモリ占有量を最小にするため、理解の容易性が犠牲になった。ソフトウェア産業界は、過去 50 年間、保守しやすく、書きやすく、バグの少ないプログラミング技法や設計方法論を模索してきた。その集大成が「構造化プログラミング」である。構造化プログラミングは、処理効率よりも、保守性を重視する。処理速度を最優先にする組込み系プログラミングとは、目指す方向が大きく異なる。

(3) プロ仕様の OS

組込み系 OS の基本は、MPU、メモリ、補助記憶等のシステム資源を効率良く運用することにある。効率化や性能のためには、使いやすさや分かりやすさが犠牲になり、例えば、メモリ保護機能、仮想アドレシング機能が無かったり、ファイルシステムさえ実装していないものも少なくない。誰でも使えるように、一般化、抽象化に向かう汎用 OS とは異なり、組込み系 OS はプロが対象の特殊な世界であり、汎用 OS とは対極の世界である。

「性能 vs 保守性」は、組込み系ソフトウェア開発における永遠のテーマであり、組込み系ソフトウェア開発は、50 年前から、性能重視と保守性重視の間を振り子のように揺れながら、少しずつ高度を稼いで進化している。

4.2 組込み系ソフトウェアの特徴とテスト

一般のソフトウェアに比べると、「性能 vs 保守性」の他に、組込み系ソフトウェアには、以下の 4 つの特徴がある。

(1) 大量に出荷するので、低出現頻度のバグもユーザ側で出る

発生確率が 0.001％のバグがあるとする。数十台しか出荷しないメインフレームでは、ユーザ側でバグが発生する回数は 0.05 回程度だろうが、10,000 台以上出荷する組込み系製品では、単純計算で 10 回も発生する。

重要度は極めて大きいが発生頻度が非常に低いバグがあり、再現条件を特定出来ず、従って、修正出来ていないとする。メインフレーム系の場合、再発したときにデータを採集するロジックを埋め込んで出荷することもあり得る。一方、出荷台数が非常に多い組込み系では、低発生頻度のバグでもユーザ側で必ず出るので、重大バグは絶対に修正する必要がある。

(2) OS の先にもバグがある

通常のコンピュータでは、ハードウェアや OS は先行開発をしているので、十分テストをして、ある程度の品質を確保している。また、OS がハードウェアを抽象化しているため、ハードウェアの詳細を知らなくてよいようにしてある。一方、組込み系の場合、特定のハードウェアを意識した制御プログラムを作らねばならないし、ハードウェアや OS の開発と、アプリケーションの実装が同時に進行することが少なくない。ハードウェアや OS にもバグが多く残っており、ハードウェアのバグを見つけるには、回路図を見る能力もソフトウェア開発技術者には必要となる。

(3) 生命にかかわる製品も多い

組込み系製品の範囲は非常に広く､心臓のペースメーカ、心肺維持装置等、生命に直接関係する製品にも MPU を内蔵している。従って、性能や応答時間に対する要求が非常に厳しいだけでなく、重大バグは完全に摘出する等、高品質を確保しなければならない。品質制御もリスク管理的な発想で開発をする必要がある。

(4) 再現しない複雑なバグが多い

組込み系ソフトウェアでは、プロセスのスケジューリングをアプリケーション側でも制御するため、先行の低優先度プロセスを後発の高優先度プロセスが追い越すことがある。また、例えば、携帯電話では、音声通話の最中にメールが着信すると、通話の質を落として MPU の割り当て時間を減らし、余った MPU 時間をメール受信に配分して、リアルタイム性を保証する。複数プロセスのタイミングのテストでは、バックグラウンドでどんなプロセスをどんな組み合わせで走るか等、複雑な項目を考えねばならない。

組込み系ソフトウェアでは、バグを見つけても、再現条件を特定することが非常に難しい。従って、バグを見つけやすいように、データを収集するロジックを埋め込んでプロセスの動作を可視化したり、それ以前に、可能な限りバグを作り込まない開発方式を採用する必要がある。

5 おわりに

組込み系ソフトウェアは、目的、動作環境、開発方式とも、汎用 OS 上で走るアプリケーションの開発に比べると、極めて特殊な事情がある。この特殊性を認識しつつ、開発プロセスの中で、ソフトウェアの品質制御を実施することが重要である。

参考文献

[SEC journal21] 平山 雅之：特別セミナー「組込みソフトウェアの信頼性を考える」より，SEC journal，No.21，Vol.6，No.2，pp.89-93，2010

安全性向上への要求工学の貢献の可能性

筑波大学大学院　ビジネス科学研究科
准教授
中谷　多哉子

SEC journal No.21では、特別セミナーの様子が報告されていた[SEC journal21]。本稿では、セミナーでの筆者の発言を補うために、関係する事項について解説を行う。セミナーが開催されたことから分かるように、最近では、身の回りのほとんどの機器がソフトウェアによって制御されるようになっている。そのためか、重大事故が組込みソフトウェアの不完全さに起因して起きることも多くなってきた。1986年の米スペースシャトル／チャレンジャー号の事故以来、セーフウェア[LEVESON1995]に対する関心が高まり、安全性を確保するための研究も進められている。しかし今も事故は後を絶たない。経済産業省の製品安全ガイド[経済産業省HP]には、事故やリコールに関する情報が日々更新されている。本稿では、組込みソフトウェアがかかわる事故の例を2つ挙げ、組込みソフトウェアによる機器の安全性を向上させるために、要求工学がどのように貢献出来るのかを検討する。

1　組込みシステムの安全性を向上させるためには何が必要か

組込みシステムの安全性を向上させるためには何が必要かを考えるにあたり、2つの事例を用いることにする。

（1）衛星誘導装置付きミサイルの誤発射事故

最初の例は、2002年にアフガニスタン進行中の米軍で発生した事故である。衛星誘導装置付きの米軍のミサイルが、発射後に、発射地点の自軍の施設を目指して戻ってきた。そのため、複数の米軍兵士が死傷した[LOEB2002]。この事故は、ミサイルの目標設定後に衛星誘導装置のバッテリを交換したことによって起きた。バッテリの消耗は普通に生じる現象である。バッテリの消耗サインが出されれば、兵士はバッテリを交換しなければならない。問題は、衛星誘導装置の仕様を定義するときに、攻撃目標を設定後にバッテリを交換するという事態を想定していなかった点にあったと想像する。もしそれが想定されていれば、兵士に対して、バッテリを交換したときには必ず攻撃目標を再設定するように訓練することは出来たかも知れない。しかし、それでも、事故が起きた可能性はある。兵士が攻撃目標の設定をし忘れた可能性もあるからである。この例は、ミサイルという、製品自体の安全性の定義自体が困難なシステムであるため話がやや混乱するが、安全なシステムとは、システムの使用者が誤った使い方をしても事故を発生させないシステムである。システムの信頼性が向上した昨今、組込みソフトウェアが目指す品質は、仕様を検討する段階で漏れやすいまれなケースを、いかに多く想定し、それらに対処出来るかに左右される段階に達している。これはシステムの利用状況を豊かに想像する力が技術者に求められるようになったという意味でもある。

（2）介護監視装置による孤独死の発見遅延

次の例は、2006年に横浜で起きた事故である。介護監視装置が完備された部屋で、60歳代の男性が孤独死し、1カ月間誰にも気づかれなかったというニュースがあった[KANAGAWA2006]。これは、訪問した介護者が、訪問後に部屋を外から施錠したため、部屋が「不在状態」となってしまったことが直接の原因であった。部屋には、住人の活動状態を検知するセンサが取り付けられた監視機器もあった。しかし、それらの監視機器からの信号は「不使用」が続いていた。住人は、介護者が部屋の外部から施錠した後に倒れたと思われる。このような状況を、システムは、「住人が不在で、かつ監視機器が使用されないのは、正常である」と誤った判断をしてしまったのである。

実際の介護現場では、訪問介護者が外から施錠することはあり得る。そして、その直後に住人が倒れることも、起こり得るのである。このようなシナリオを考慮していなかったため、事故が起きたと言えよう。

事故や事件が発生した後であれば、我々は、「もし…」を想定した組込みソフトウェアを開発することは出来る。しかし、社会が我々に求めているのは、事故や事件を発生させないシステムである。事故から学ぶことも重要であるが、事故が生ずる前に、ソフトウェアの不備に気づきたいものである。最近の事故の特徴は、単なる機器の故障によって事故が発生するというよりも、横浜の事故のように、複数のまれな事象が同時に発生することによって生じている例も少なくない。このようなシナリオは、トップダウンの分析手法であるFTA[※1]やFMEA[※2]で対応出来る範囲を超えている。組込みソフトウェアの要求分析では、これらの手法を補完する新たな手法が必要である。

2　要求工学技術マップ（REマップ[※3]）

要求工学の守備範囲は、ステークホルダを定めることから、ステークホルダへのインタビューや市場調査を行って課題を発見し、解決策を見い出し、市場の動向を見極めて

開発する製品の仕様を定義し、そして、システム開発予算や技術的な実現可能性を判断し、妥当性の検証を行い、更に、要求の変更を管理するまでを含む。その範囲は決して狭くはない。そのため、その工程に適用する手法も様々である。

要求工学で使われている手法の特徴を明らかにするために、要求工学技術マップ（以下、REマップ）[TSUMAKI2006]を用いることにする。図1に、代表的な手法を配置したREマップを示した。REマップを構成する2つの軸は、定量的な軸ではなく、定性的な軸である。縦軸には、手法が分析する対象の空間の性質が表されている。分析対象の空間が閉じていれば軸の上方に、開いていれば下方に手法が配置される。横軸は、分析者の思考法を表す軸である。分析時の思考が系統立てて行われるのであれば、その手法の思考法は静的であるとみなされ、手法は軸の左方に配置される。また、手法の分析に、直感やひらめきが求められるのであれば、その思考法は動的であるとみなされ、手法は軸の右方に配置される。このように、REマップでは、各手法を4つの象限に配置することで手法の特徴を概観する。

・REマップと障害分析の代表例

例えば、ゴール指向分析では、分析対象は限定されていないため、その空間は開いていると見なすことが出来る。また、分析の思考法は、トップゴールからサブゴールの導出、ゴール間の依存関係の定義といったように、系統立てた方法で進められる。従って、ゴール指向分析はREマップの第3象限に位置付けることが出来る。これに対してブレインストーミングは、その分析対象の空間は開いており、その分析過程では新たな発想やひらめきが求められる。従って、ブレインストーミングは、第4象限に位置付けられる手法となる。シナリオ分析やユースケース分析は、与えられた状況や条件に対してシナリオを定義する活動となるため、その分析対象の空間は閉じている。しかし、分析を行うときには、様々な状況の判断を行う必要があるため、その思考法は動的である。また、ソフトウェア工学で用いられる段階的詳細化の分析対象の空間は閉じており、系統立てられた思考法を用いて分析が進められる。以上のことから、これらの手法は、それぞれの象限に位置付けることが出来る。

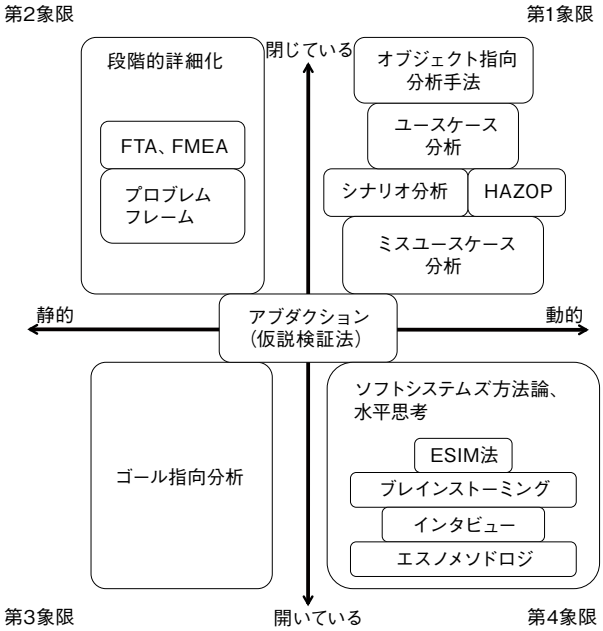


図1　REマップ

FTAやFMEA、そしてHAZOP※4は、どこに位置付けられるであろうか。FTAやFMEAは、限定された領域の分析対象に対して、系統立てられた思考法を用いて分析を進める。従って、第2象限に位置付けられる手法である。これに対してHAZOPは、ガイドワードを適用するときには状況のイメージを膨らませて故障や障害が生ずる可能性を想定するため、その思考法は動的であると考えられる。しかし、分析対象は閉じている。従ってHAZOPは、第1象限の手法と見なせる。

このように、REマップ上に障害分析の代表的な手法を配置してみると、想定困難な状況の可能性を広げてイメージするという、開いた分析対象に対して動的な分析を行う手法ではないことが分かる。

3　分析手法の検討

衛星誘導装置付きミサイルの事故を未然に防ぐためには、どのような手法を適用する必要があるだろうか。ここでは、複数の手法を適用して検討することにする。

（1）ミスユースケース分析 [ALEXANDER2003]

ミスユースケース分析は、第1象限の手法である。正規の要求であるユースケースの実現を阻害したり、悪用したりするミスユースケースを発見することによって、ミスユースケースがシステムに与える脅威を緩和する必要性に気づくことが出来る。そして、そのような手段となるユースケースを新たな要求として発見することが可能となる。しかし、「ミサイルを攻撃目標まで誘導する」という衛星誘導装置のユースケースに対して、「バッテリ切れ」というミスユースケースの脅威を緩和させる活動「バッテリ交換」が、攻撃目標の設定に影響を与えることを知っていな

脚注

※1　FTA：Fault Tree Analysis
※2　FMEA：Failure Mode and Effect Analysis
※3　REマップ：Requirements Engineering (Technology) Map
※4　HAZOP：Hazard Analysis and Operability Study

ければ､「攻撃目標の記憶」を新たなユースケースとして提示することは出来ない。

(2) FTA・FMEA

FTA を用いれば、攻撃を失敗させる「ミサイルが不発」となる原因を発見することが可能である。同様に、「ミサイルが衛星誘導装置によって攻撃目標に誘導されなくなる」原因が、「衛星誘導装置のバッテリ切れ」によって誘導障害が発生することも発見することが出来る。また、FMEA を用いることによって、バッテリ切れという一種の故障が「攻撃目標までの衛星誘導装置による誘導」を失敗させることを明らかにすることが可能となる。しかし、現実に起きた事故の原因やシナリオを発見することは困難である。なぜならば、このシナリオを発見するためには、衛星誘導装置の起動処理の仕様、及び他の機器と交換されるデータの内容を参照する必要があるからである。

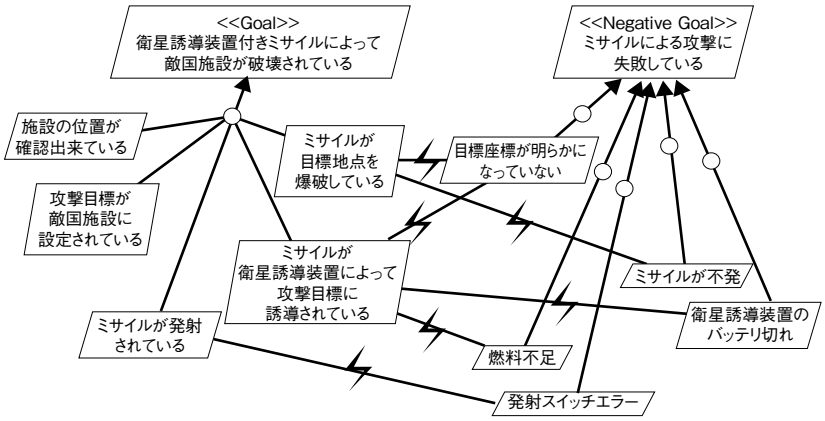


図２　ゴール指向分析結果（KAOS）

(3) モデル検査 [中島 2008]

起こってはいけない状態を発見するために、モデル検査という強力な分析手法を適用することが可能かも知れない。モデル検査はツールによる支援が可能な第２象限の手法である。バッテリや衛星誘導装置の状態モデルを分析対象としてモデル検査を行えば、「攻撃目標を設定した後にバッテリが切れ、その後にバッテリを交換してミサイルを発射したとき、自軍に向かって飛来する」というシナリオを見つけることは出来る。しかしそれは、「ミサイルが自軍を目指して飛来すること」が状態モデルに示されており、かつ、ミサイルが自軍目指して飛来することは起きてはならない事態であることが定義されている場合にのみ定義することが可能である。しかし、これらの装置には、「自軍」と「敵軍」の区別をつけることは出来ない。それを区別出来るのは、人間だけである。

(4) ゴール指向分析

ゴール指向分析の手法は、先にも述べたように第３象限に位置付けられる。その分析対象の空間は開いているが、思考は系統立てた方法によって進められる。そのため、論理的飛躍の余地は無い。

図2に､衛星誘導装置付きミサイルのKAOS [KAOS2007] の表記を用いたゴールモデルを示した。図中、上位に位置している四角形が上位ゴールである。上位ゴールを達成するために達成しなければならないゴールは、下位ゴールとして、上位ゴールの下に示されている。円は、下位ゴールがすべて達成されたとき（and 依存）に上位ゴールが達成されることを表すノードである。矢印で上位ゴールに接続された下位ゴールは、いずれかが達成される（or 依存）ことによって､上位ゴールが達成されることを表している。稲妻の絵が付与された線は、一方のゴールが達成されることによって､他方のゴールの達成が阻害されることを表す。とくに図２では、≪ Negative Goal ≫と書かれたゴール木のゴールが左側に示された正規のゴールの達成を阻害する。例えば、「ミサイルによる攻撃に失敗している」の下位ゴール「ミサイルが不発」は、「ミサイルが目標地点を爆破している」という下位ゴールを阻害する。また、攻撃の失敗は、複数の下位ゴールのいずれかが達成されると達成されてしまう。しかし､左方の≪ Goal ≫と書かれたゴール木では、すべての下位ゴールが丸いノードに接続されているため、下位ゴールがすべて達成されなければ上位ゴールを達成することは出来ない。このような思考過程で作られるゴール木には、システムを構成する部品の仕様を取り込むことが困難である。従って、ミサイルが自軍を攻撃するように作られていることを発見するのは容易ではない。

システムに期待された自国を守るという安全性を向上させるためには､ここで検討した手法だけでは不十分である。分析対象を限定しない動的な手法、すなわち第４象限の手法が必要である。第４象限の手法には、ブレインストーミングや、実際の作業状況を詳細に観察するエスノメソドロジといった手法がある。しかし、これらの手法では、分析者の発想能力に頼り過ぎる感が否めない。次の項では、動的な思考法を用いたESIM[※5]法を紹介する。

4 ESIM法[MISE2005]

ESIM 法の特徴は、人の活動や環境をシステムの一部として取り込み､その多様性を分析対象としている点にある。ただし、分析対象を必要以上に広げないために、FTA や FMEA を活用して分析対象の制御は随時行われる。

基本的な手法の流れは図３の通りである。

①環境や利用者を含めて分析対象のシステムとする。
②システムを構成する構成要素の仕様で定められた正常状態と異常状態を抽出する。例えば、機器が異常発熱する場合等は、あらかじめ異常状態として定義されていることが多い。
③各構成要素、及びそれらを接続する要素に対して、HAZOP のガイドワードを適用し、未定義の状態を非正常状態として抽出する。例えば、利用者がスイッチを短期間に繰り返し押下している、長押ししている、等が非正常状態となることがある。
④専門家が非正常状態へ対処する必要が無いと判断すれば、その状態を分析対象から外す。
⑤正常状態、異常状態、そして分析対象として残された非正常状態の状態リストに着目し、それが他の機器に与える影響を分析する。そのために、分析対象となる非正常状態が他に与える影響をイベントとして定義する。例えば、「バッテリ切れ」等がある。
⑥すべての正常状態、異常状態、非正常状態の各構成要素に対して、⑤で発見したイベントを送り込み、状態の遷移の有無を検討する。例えば、「バッテリ切れ」というイベントを待機中の兵士が受け取ると、兵士は、バッテリ交換作業中という正常状態か、または、バッテリ交換を忘れているという非正常状態に遷移する。
⑦⑥で新たな非正常状態が発見された場合は、それを⑤の状態リストに加え、分析を続ける。

図３　ESIM 法の基本的な流れ

図３に示す分析を続けることで、正常状態、異常状態及び非正常状態が複合して生ずるイベントを発見することが可能となる。その結果、複合事象による副作用を構成要素の状態として定義出来るようになる。例えば、兵士がバッテリ交換を行う副作用として、衛星誘導装置の初期化が起動されることに気づき、その結果、ミサイルが自軍を攻撃目標として発射されることに気づくことが可能となるはずである。

組込み製品の安全性を向上させるために、このような第４象限の分析手法を適用してみる価値はある。このような手法では、発見や気づきといった動的な思考法が使われるからである。

現在、ESIM 法は、家電製品の安全性向上に実用されている。この手法は、分析を進めても分析対象が閉じないため、状態の爆発が生ずることが危惧される。しかし、実際に製品の分析に適用したところ、適用可能な時間内で分析を完了することが出来た。この報告では、従来の手法では発見出来なかったシナリオを２件発見することが出来たことが記されている [MISE2006]。この２件のシナリオを発見出来たことによって、製品の安全性は向上したのである。

5　システムの役割に着目した要求分析

組込みシステムと利用者との間には、仕様に定められた役割分担がある。システムは、利用者に対して、使用に際しての約束事を守ることを期待する。一方の利用者は、システムに対して、仕様に定義されたサービスを提供する責任を求める。分析対象を閉じるということが、システムの利用者に対する責任を忘れ、システム内部の相互作用に焦点を絞り込むという意味になってはならない。システムは外界と相互作用するものだからである。そのため、要求工学では、利用者や環境を含めてシステムと呼ぶことが多い。

要求工学では、要求獲得や要求の仕様決定にかかわる人々をステークホルダと言う。以前はユーザの要求分析と言われていた。ユーザがステークホルダに置き換わった背景は、このように、システムと直接かかわる人々のみの要求に対応したのでは、様々な問題をもたらすことが明らかとなってきたからである。ステークホルダには、社会通念、社会の慣習や習慣、倫理等も含まれる。

衛星誘導装置の要求仕様を定義するとき、地球という環境を考慮に入れていたら、攻撃目標に自軍のみならず、自国も設定可能であることに気づくことが出来たかもしれない。分析対象を閉じるということは、これらの検討すべき事項を分析対象から外すことに他ならない。問題に気づくことが出来たならば、それへの対処を考えるのは難しくはない。

組込みソフトウェアの要求分析において、安全性を考えるのであれば、分析対象の空間を閉じ込めず、動的な思考法を用いた分析手法を試してみる価値はある。

脚注

※５　ESIM：Embedded System Improving Method

参考文献

[ALEXANDER2003] Alexander, I. : "Misuse Cases : Use Cases with Hostile Intent," IEEE Software, vol.20, no.1, pp.58-66, 2003
[KANAGAWA2006] 神奈川新聞：“入居者の死亡、１カ月気付かず／横浜の高齢者用住宅”, 神奈川新聞社ローカルニュース , 2006 年２月 10 日
[KAOS2007] Objectiver : "A KAOS Tutorial, Objectiver," http://www.objectiver.com/fileadmin/download/documents/KaosTutorial.pdf
[LEVESON1995] Leveson, N. : Safeware : System Safety and Computers, Addison-Wesley Professional, 1995（松原 友夫 監修：セーフウェア 安全・安心なシステムとソフトウェアを目指して , 翔泳社 , 2009）
[LOEB2002] Loeb, V. : "'Friendly Fire' Deaths Traced to Dead Battery," Washington Post, Sunday, Page A21, March 24, 2002
[MISE2005] Mise, T., Hashimoto, Y., Katamine, K., Shinyashiki, Y., Ubayashi, N. and Nakatani, T : "An Analysis Method with Failure Scenario Matrix for Specifying Unexpected Obstacles in Embedded Systems," Proc. of the Asia-Pacific Software Engineering Conference (APSEC'05), IEEE, pp.447-456, 2005
[MISE2006] Mise, T., Hashimoto, Y., Katamine, K., Shinyashiki, Y., Ubayashi, N. and Nakatani, T : "A Method for Extracting Unexpected Scenarios of Embedded Systems," Proc. of the Joint Conference on Knowledge-Based Software Engineering 2006 (JCKBSE'06), IOS Press, pp.41-50, 2006
[SEC journal21] 平山 雅之：特別セミナー「組込みソフトウェアの信頼性を考える」より，SEC journal，No.21，Vol.6，No.2，pp.89-93，2010
[TSUMAKI2006] Tsumaki, T. and Tamai T. : "A framework for matching requirements elicitation techniques to project characteristics," Software Process : Improvement and Practice, vol.11, no.5, pp.505–519, 2006
[経済産業省 HP] 経済産業省：製品安全ガイド , http://www.meti.go.jp/product_safety/
[中島 2008] 中島 震 : SPIN モデル検査 - 検証モデル技法 , 近代科学社 , 2008

Eメールアーカイブのクラスタリングによる開発コンテキストの可視化

大蔵 君治† 川口 真司†† 飯田 元†††



情報化の進む現代社会において，ソフトウェアはインフラの重要な構成要素となっている．しかしながら，ソフトウェア開発プロジェクトでは今もなおコスト超過や納期遅れといった“失敗”が相次いでいる．失敗の原因の1つに「プロジェクト分析の不足」が挙げられる．プロジェクトの事後分析は人的・時間的なコストが掛かるため，分析に十分な時間を確保することが難しいのが現状である．そこで，本研究ではソフトウェア開発においてよく用いられるメールに着目し，そのやり取りを時系列上に表示する手法を提案する．これにより，従来のメトリクスグラフで把握していたプロジェクトの「状態」に加え，「現場の開発背景」を同時に俯瞰することが可能となる．本稿では手法の詳細と共に，実際の開発プロジェクトに本手法を適用した3つのケーススタディを紹介する．

A Method for Visualizing Contexts in Software Development using Clustering Email Archives

Kimiharu Ohkura†, Shinji Kawaguchi†, and Hajimu Iida†

Software is a prominent factor of infrastructure in a modern information society. However failures in software development project such as cost or deadline over still remain. One of causes of the failures is considered a lack of postmortem analysis for project improvement. Because a postmortem analysis has high costs of time and human resource, it is difficult to perform the analysis continuously under the current circumstances. In this research, we focus on email conversations often used in software development, and we propose a method for visualizing them in the project. Our method helps to grasp "contexts" in development further to existing visualizing approaches such as metrics chart that can reveal only "state". In this paper, we describe details of our method and case studies of applying the method to actual projects.

1 はじめに

情報技術の発展によって，ソフトウェアは社会インフラの中枢をなす重要な構成要素となっている．それに伴い，ソフトウェア開発の需要は継続的に増え続けており，数々の開発プロジェクトが日々遂行されている．しかし，コスト超過や納期遅れに代表される「失敗プロジェクト」は，今もなお多く存在する [NIKKEI2008][データ白書 2009].

プロジェクトの失敗が減らない理由の１つとして，失敗の原因を分析することが難しいという点が挙げられる．プロジェクトの事後分析は，開発記録やプロジェクトメンバの記憶を頼りに行うこととなるが，膨大な記録から事後分析に有用な情報を取り出すには時間的なコストがかかる．また，プロジェクトメンバが開発中に起こった出来事をすべて思い出せるという保証は無い．このような理由から，開発プロジェクトの事後分析を継続的に行うことが困難となっている．

この問題に対し，開発過程で得られた定量データを用いて，プロジェクトの進捗やソフトウェアの品質を測る研究が数多くなされてきた [CHIDAMBER1994][KAMIYA2002][QUENTIN2006].

しかしながら，定量データから得られる情報はあくまでプロジェクトがそのときどのような「状態」であったかを数値で表したものであり，なぜプロジェクトがそのような状態になったのかという「原因」を調べるためには，プロジェクトメンバへのヒアリングが必要不可欠となる．ヒアリングは事後分析において有効な手段であるが，実施にかかるコストが分析者，開発者共に大きい．また，分散開発にお

† 奈良先端科学技術大学院大学 情報科学研究科 研究員, Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Reseacher
†† 奈良先端科学技術大学院大学 情報科学研究科 助教, Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Assistant Professor
††† 奈良先端科学技術大学院大学 情報科学研究科 教授,Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Professor

いては，モジュールごとの開発担当者があいまいであることも多く，ヒアリングの対象者を特定出来ないといった問題がある [ANVIK2006].

本研究ではこの問題に対し，開発に用いられたEメール（以下，メール）と，開発リポジトリから得られる定量データを対象とした分析手法を提案する．メールはソフトウェア開発プロジェクトにおいてよく用いられるコミュニケーションツールであり，数値データからは読み取ることの出来ない現場の情報が蓄積されていると考えられる．例えば，ミーティング時間の超過が開発の遅れの原因となっていた場合，開発が滞っているという「状態」はソースコード行数グラフ等，数値データの時系列上における変化から確認することが出来る．しかし，なぜ開発が滞っているかという「原因」をグラフから得ることは出来ない．このようなとき，同時期に交わされたメールのやり取りを確認すればミーティングの開催といった種々の情報が得られる可能性がある．

本研究ではこのような「数値データに直接現れない現場の情報」をコンテキストと総称し，中でも，蓄積されたメール群に存在するひとまとまりの話題（開発背景）として体現されるコンテキストを「メールに基づくコンテキスト情報」として可視化の対象とする．なお，簡略のため以下では，上記をメールコンテキストと呼ぶ．すべての有益なコンテキストがメールのやり取り中に記録されていているとは限らないが，開発者へのインタビューやドキュメントの精査を実地に行うことに比べると，はるかに低コストでの抽出が可能である．また，本論文の実験結果はメールコンテキストが十分有益な情報を含むことを示唆している．

人手でメールコンテキストを確認する作業は，メールの件数が多い場合の分析コストが高く，また，長期にわたって潜在的に存在する話題の把握が困難である．本研究では，以下に示すようなアプローチによってメールコンテキストを自動的に可視化し，分析を支援する．

・ベクトル空間法とクラスタリングを用いたメールコンテキストの自動抽出
・メールコンテキストの時系列上へのプロット

メールコンテキストを時系列に置くことで，ソースコード行数グラフをはじめとする他の時系列グラフとの重畳表示が可能となる．これにより，開発プロジェクトの「状態」と「原因」を相互に分析しやすくなる．例えば，数値データから開発が滞っているという「状態」が判明したときに，その時点に存在するメールコンテキストを見ることで，「原因」の推測がより容易になる．

本研究は開発プロジェクトの事後分析を支援するためのものであるが，当該プロジェクトのメンバ，及び外部分析者の両方を利用者に想定している．以下に，提案する手法を用いた事後分析のユースケースを簡単に示す．

(1)　外部分析者による利用

当該プロジェクトに関する予備知識に乏しい外部の分析者（研究者等）が，プロジェクトの要因分析を行ったり，状況把握を行ったりする場合には，分析の糸口を得るという観点から，本手法は非常に有用であろう．

(2)　当該プロジェクトのメンバによる利用

プロジェクトの当事者が失敗の原因を調査したり，過去のプロジェクトで発生した事例を調べたりするために本手法を利用出来る．当事者であればメール以外の開発データへのアクセスも容易であるため，多種の開発記録と本手法の出力結果を併せて，プロジェクトの精査が可能であると考えられる．

以降，第2節ではメール分析を扱った関連研究と既存のツールを使った事後分析方法について述べる．第3節では手法の詳細について説明し，第4節では実際に手法を適用した3つのケーススタディについて述べる．最後に，第5節にてまとめと今後の展望について述べる．

2　関連研究

本節ではメールの分析に関する関連研究と，本手法と既存のプロジェクト情報管理ツールとの位置付けについて述べる．

2.1　メールの分析に関する研究

メールを対象とした分析手法は，これまでにも数多く行われてきた．ソフトウェア開発分野におけるメールの分析手法は，大別すると，送信数等のメタ情報を用いたソーシャルネットワーク分析（以下，SNA※1）と，自然言語解析を用いた手法に分けられる．

Birdらは，SNAアプローチを用いて大規模なオープンソースソフトウェア（以下，OSS※2）プロジェクトに対して調査を行った [BIRD2006]．調査は開発用メーリングリストに

脚注
※1　SNA：Social Network Analysis，ソーシャルネットワーク分析
※2　OSS：Open Souce Software，オープンソースソフトウェア

対して行われ，各開発者のメール送信数や被送信数，媒介中心性等のメトリクスを用いて分析を行った．調査の結果，各開発者のコミット数とメール送信数には高い相関があることや，メーリングリストへの投稿は一部のアクティブな開発者らによって大半が占められていること等が明らかとなった．このように，SNA 手法は主に開発者に着目した分析を行うために用いられる．そのため，SNA によって各開発者の特性が明らかになったとしても，どの開発者がどのモジュールを担当していたかといった，ソフトウェア成果物との関連を調べるためには手動での調査が必要となる．Sarma らはこの点に着目し，SNA によって描かれた開発者同士のネットワークと，モジュール依存度を基に描かれたファイルネットワークを関連付けて提示するツール“Tesseract”を開発した [SARMA2009]．例えば，ある開発者に着目し，その開発者と関連の強い他の開発者，及びモジュールを同時に提示するといったことが可能となる．しかし，ファイルネットワークと開発者ネットワークの関連付けはすべて手動で行われているため，分析のための事前準備にかかるコストは高い．

また，時系列情報を持つ文書の到着頻度に着目し，そのトピックの活発性を測る研究も行われている [KLEINBERG 2002][崔 2004]．これらの研究は文書（記事）の到着時間間隔を手掛かりに分析を行っているため，本文の影響を受けずにトピック活性度の計測が可能である．しかし，これらの手法はリアルタイムに文書が到着することを想定している上，各種パラメータの設定は経験的に決められており，調整が極めて困難であると思われる．

Rigby らは，メール本文を解析し，その言語的手掛かりによってプロジェクトの動向を分析した [RIGBY2007]．分析には LIWC※3[LIWC] と呼ばれるツールを用いている．LIWC は辞書登録された文章中のキーワードをカウントし，その文章を 70 種類の心理的特徴に分類するツールである．例えば“may be”といったキーワードは心理的特徴「あやふや（tentativeness）」に，“important”等のキーワードは「確信（certainty）」にそれぞれ関連付けられている．Rigby らは分析の中で，優秀な開発者と一般の開発者で感情変化にどのような差があるか等を調べ，優秀な開発者ほど外向性が低いことや，感情性（神経質さ）は優秀な開発者と一般の開発者で差が無いこと等を示した．しかし，このような言語的手掛かりを用いた手法は，くだけた会話やコミュニティ依存の専門用語等を使用するインフォーマルな会話には適用しにくいため，分析対象は一部の OSS プロジェクト等に限定される．

また，メール本文の特徴語に着目した討議構造の検出に関する手法も提案されている [仁野 2008][中山 2008]．これらの手法は，メール本文の構造がある程度形式的であり，時間や場所に関連付けられた単語が本文から抽出可能であること，また，「ところで」，「なぜかというと」といった発話上の言語的手掛かりが本文中に存在することを前提としている．そのため，開発プロジェクト内でのみ用いられる専門的な用語やスラング，あるいはインフォーマルな会話には不向きである．

本研究はメールの本文を対象としてクラスタリングを行う．提案手法では意味解析，構文解析，討議構造の解析等は行わず，単語の統計情報のみを用いる．そのため，分析用の辞書を作成する必要は無く，異なる言語圏でも同じ手法を適用することが出来る．

2.2 プロジェクト管理ツールによる事後分析

今日ではソフトウェア開発に様々なツールが用いられており，その代表的なものに Bugzilla（バグジラ）[BUGZILLA] や GNATS（グナッツ）[GNATS] 等の障害管理ツールや，Microsoft Project 等の進捗管理ツールが挙げられる．このようなツールを導入しているプロジェクトにおいては，メールを確認せずともミーティングのスケジュールやバグの詳細情報を正確に把握することが可能である．しかしながら，事後分析を行う者が常にそれらのツールを利用出来るとは限らず，データの互換性にも問題が生じる可能性がある．

本研究で入力として用いるメールアーカイブは，一般的な MUA※4 で読み書き可能な標準的なフォーマットであるため互換性の問題は発生しにくく，単一のファイルであるため可搬性にも優れている．更に，提案手法では「ミーティングの日程」といった単純な情報だけでなく，「なぜその日程に決定したか」といったコンテキストを抽出出来る可能性がある．一般に，このような管理ツールには形式的な情報（結果）が記録されるのみであり，スケジューリングのミスといった事象（原因）はログに残されていない．提案手法は，そのようなインフォーマルなやり取りにおいてのみ記録される情報を抽出出来る可能性がある．ただし，本研究は従来の分析手法に置き換わるものではなく，従来の分析結果に重ね合わせることによって，事後分析における情報量を増加させることを目的とする．

3 分析手法

本手法は，単一のメールアーカイブを対象とする．メールコンテキストの抽出手順は次の通りである．

① 各メールのベクトル化

すべてのメールの本文に対して形態素解析を行い，単語を抽出する．そして，抽出された単語の頻度情報を要素として持つベクトルを，すべてのメールに対して生成する．

② クラスタリングによる話題の自動分類

ベクトル化された各メールは，ベクトル演算によって類似度（距離）の計測が可能である．このとき，「類似度が高い」ということは「似通った単語が出現するメール同士」であることを意味する．すべてのベクトル間で類似度を計算し，クラスタリングアルゴリズムに基づいて各ベクトルを類似度の高いもの同士でカテゴライズしていく．

③ クラスタの時系列上へのプロット

クラスタリングによって得られた各クラスタは，似通った内容を持つメールの集合である．本研究ではこれを１つの話題であると見なし，メールコンテキストと呼ぶ．クラスタの要素であるメールには，送信日時が記録されているため，これを用いて時系列上にプロットする．

時系列上にプロットされた各クラスタを，他の時系列グラフと重畳表示することで，プロジェクトの状態とメールコンテキストを同時に俯瞰することが出来る．概念図を図１に示す．以降では，各手順の詳細と，実装したツールについて説明する．

3.1　各メールのベクトル化

各メールは，ベクトル空間モデル [SALTON1975] に基づいてベクトル表現へと変換される．ベクトル空間モデルでは，メール中に出現する単語（索引語と呼ぶ）の頻度がベクトルの要素となる．また，生成された各メールのベクトル表現のことをここでは文書ベクトルと呼ぶ．図２に，文書ベクトルの概念を示す．

このとき，単純に単語の出現頻度をそのまま文書ベクトルの要素とした場合，どのメールにも普遍的に表れる単語（例えば「私」等）が類似度に大きく影響してしまう．この問題を回避するために，TF-IDF※6 [SALTON1987] による重み付けを行う．TF-IDF とは情報検索等で用いられている単語のスコアリング手法である．単語の出現頻度である TF に，その単語が検索対象内においていかに希少かを示す値である IDF を乗ずることで，普遍的な単語の影響を少なくすることが出来る．TF-IDF の式を以下に示す．

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i + 1}\right)$$

$w_{i,j}$：文書 j における単語 i の重み（スコア）
$tf_{i,j}$：文書 j における単語 i の出現頻度
df_i：単語 i を含む文書 j の数
N：その文書空間における文書の総数

本研究における文書空間はメールアーカイブ全体であり，文書は各メールの本文に相当する．

提案手法では，あらかじめすべてのメールの単語に対して，TF-IDF によるスコアリングを行っている．その際，各メールでスコアの高かった単語をキーワードとしてリストアップしておき，それらのキーワードを連結したものを索引語として用いている．これは，索引語に普遍的な単語が入らないようにするためである．なお，次元数が増加すると計算量が膨大になるため，各メールからリストアップするキーワードの数は２とした．また，予備実験の結果から，抽出する単語は名詞と未知語のみとした．形態素解析システムには Sen[SEN] を用いている．

3.2　クラスタリングによる話題の自動分類

文書ベクトル間の類似度計算後は，クラスタリングアルゴリズムに従って各メールを話題別に分類していく．クラスタリングアルゴリズムには Ward 法※7，k-means 法※8，最長距離法等様々なものがあるが，どのアルゴリズムが最

図1　コンテキストの重畳表示（概念図）

索引語：	コンパイル	点検	例外	停電	エラー
文書１	0	1	0	2	0
文書２	2	0	3	0	1
・・・					

索引語の出現頻度

d_1=(0, 1, 0, 2, 0)
d_2=(2, 0, 3, 0, 1)

図2　文書ベクトルの概念

脚注

※3　**LIWC**：Linguistic Inquiry and Word Count
※4　**MUA**：Mail User Agent
※5　**LOC**：Lines of Code
※6　**TF**：Term Frequency，**IDF**：Inverse Document Frequency
※7　**Ward 法**：Ward's Method，ウォード法
※8　**k-means 法**：日本語では「K 平均法」と呼ばれる非階層型クラスタリングアルゴリズムの１つ。

適であるかは生成された文書ベクトルによって異なる．本研究では，使用するクラスタリングアルゴリズムやその閾値についてはとくに定めず，何度かの試行によって最適なアルゴリズムを分析者が判断し，選択することとする．類似度に用いる尺度についても同様に，ユークリッド距離，コサイン距離，マハラノビス距離等から最も精度が良いと判断したものを選択する．

3.3 クラスタの時系列上へのプロット

クラスタリングによって得られたクラスタは，類似度の高い文書ベクトルの集合である．時系列上へのプロットは，各文書ベクトルに対応するメールの Date ヘッダから日付情報を取得して行う．メールを個別にプロットするのではなくクラスタ単位でプロットすることで，メールコンテキストを俯瞰しやすくなる他，長期間にわたって散在する話題や，定期的に行われるイベント等の把握を容易にする．

3.4 ツールの実装

本手法は次の 3 つのツールによって実現している．

・データコンバータ（Java，コンソール）
・クラスタリングツール（C#，GUI）
・プロジェクトリプレイヤ（C#，GUI）

データコンバータとクラスタリングツールは，本研究のために独自に開発したものである（形態素解析システムを除く）．入力から可視化までの流れを図 3 に示す．データコンバータは mbox 形式のメールアーカイブをベクトルデータに変換する他，クラスタリングツールが出力するログデータから樹形図を画像ファイルとして出力したり，クラスタリング結果を html 出力する機能を備えている．図 4, 図 5 に，それぞれの例を示す．クラスタリングツールは，ベクトルデータを読み込み，クラスタリングを実行する GUI プログラムである．GUI では，使用するクラスタリングアルゴリズムや距離尺度，閾値等を設定出来る．設定画面を図 6 に示す．プロジェクトリプレイヤ（以下，リプレイヤ）は，本研究チームで開発を行っているプロジェクト分析支援ツールである [OHKURA2006]．リプレイヤは，自動収集された定量データを基に，開発プロジェクトをビデオのように再生する機能を持つ．リプレイヤが提示出来る情報の 1

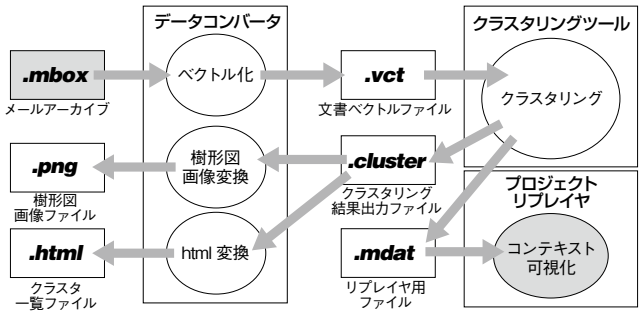


図3　コンテキスト可視化までの流れ

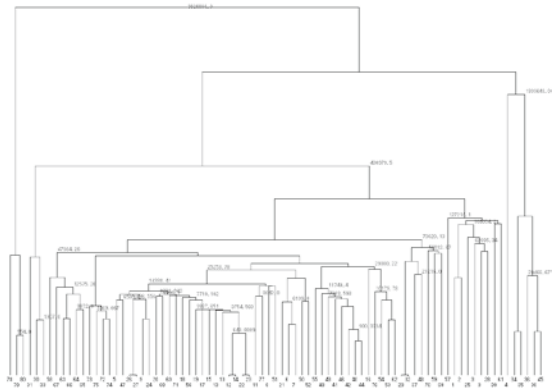


図4　出力される樹形図（画像ファイル）の例

図5　クラスター覧のhtml出力

図6　クラスタリングツールの設定画面

つとして，ソースコードメトリクスのラインチャートがある．我々は，このラインチャートにクラスタを重畳表示出来るよう，リプレイヤに改良を加えた．クラスタを重畳表示したラインチャートを図7に示す．水平に延びている線が1つのメールコンテキスト（クラスタ）を表す．実装上の問題で，キーワードをチャート上に表示することは出来なかったが，線上をマウスオーバすることで，そのクラスタを構成するメール中の，スコアの高い単語を表示することが出来る．クラスタ数が多くて視認性が低い場合は，右に表示されたクラスタ一覧から，見たいクラスタだけを選択して表示させることが出来る．黒い縦線は現在の日付を表す．現在の日付をまたがって存在するクラスタ（縦線と交差している横線）に含まれる個々のメールは，別ウィンドウの「メールビュー」にて確認することが出来る（図8）．

4 ケーススタディ

我々は本手法の有効性を検証するために，3つの開発プロジェクトデータに対して適用実験を行い，以下に挙げる3つの観点から有効性を検証した．

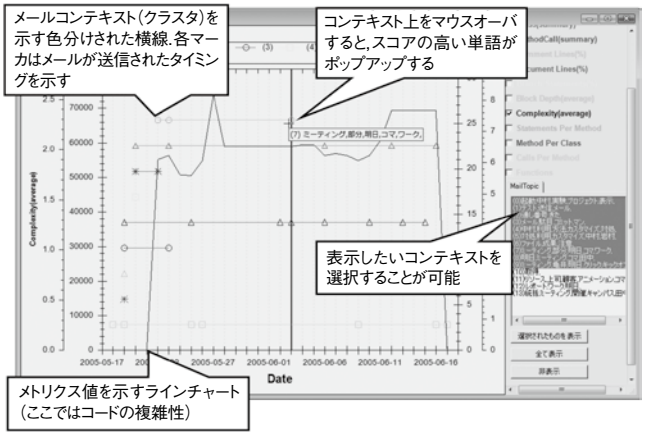


図7　メールコンテキストの重畳表示（リプレイヤ）

図8　メールビュー（リプレイヤ）

・提案手法の実現可能性
・提案手法の優位性
・得られたクラスタの正確性

以降4.1 ～ 4.3節にそれぞれの観点からの評価を示す．

4.1 小規模開発への適用

我々はまず，本手法の実現可能性（実際にメールコンテキストに相当する話題が抽出出来るかどうか）を確認するために，小規模なソフトウェア開発プロジェクトに対して適用を行った．クラスタリングアルゴリズムには，試行の結果，最もバランスの良い樹形図を描いたWard法を使用した．クラスタリングを終了させるための閾値にはクラスタ間類似度の平均値を使用し，類似度はWard法の定義によりユークリッド距離を用いた．対象プロジェクトの詳細は表1の通りである．なお，開発要員はすべて奈良先端科学技術大学院大学の学生で，開発は主に学内で行われた．

当該プロジェクトの開発用メーリングリストに対して手法を適用した結果，リプレイヤのラインチャート上にクラスタを重畳表示させることが出来た（図7）．我々は，1人の被験者（研究者，当該プロジェクトとは無関係）を用意し，このプロジェクトのメールコンテキストを分析してもらった．その結果，数値データには現れない以下のような開発状況を，30分程度の調査で把握することが出来た．

・各開発者の役割
・ミーティングスケジュールの調整
・コード修正のアナウンス
・リソースのアップロード報告

当該プロジェクトのメールコンテキストを表2に示す．表2のメールコンテキストは，当該プロジェクトの開発メ

表1　対象プロジェクト（小規模開発）

プロジェクト	ゲーム開発（大学オープンキャンパス用）
開発要員	7人
開発期間	27日
メールサイズ	272KB，81件

表2　プロジェクトメンバが手動で抽出したメールコンテキスト一覧

番号	メールコンテキストの内容
1	EPMの導入について
2	メーリングリストの設定議論
3	メーリングリストへのテスト送信
4	EPMの不具合について
5	開発ミーティングの内容調整
6	CVSコミットメールに関する話題
7	リソース（画像）作成に関する報告
8	開発するフレームワークに関する打ち合わせ
9	統括ミーティングの日程調整

ンバの 1 人によって手動で抽出されたものである．インタビューの結果，「ミーティングスケジュールの調整」，「コード修正のアナウンス」，「リソースのアップロード報告」は，順に表 2 における 5, 6, 7 番にあたるとメンバは判断した．「各開発者の役割」は表 2 のメールコンテキストのいずれにも該当しないが，被験者は複数のメールコンテキストを見ることでそれぞれの開発者の役割を判断した．また，その役割の判断が正しいものであるかをメンバによって確認してもらった．その結果，被験者が判断した各開発者の役割は正しいものであることが確認出来たが，インタビューを行ったメンバも一部のメンバについてはその役割の詳細を忘れていた．回顧を促すためにクラスタリング結果の html を閲覧してもらったところ，短時間で各メンバの詳細な役割を思い出すことが出来た．

また，被験者は「全くメールコンテキストが存在しない期間」にも着目した．同時期のコード行数グラフにも動きが見られないことから，被験者は何らかの理由でこの期間は開発が停滞していたと予想した．被験者は，停滞期間の前週に「試験」等の単語を含むメールコンテキストが存在することから，「学科試験のために開発が中断していた」という推測を行った．この推測は正しいものであったが，事実確認には当該プロジェクトメンバへのヒアリングが必要であった．ただし，メールの内容から各開発者の役割が大まかに把握出来ていたため，ヒアリングにかかる時間を抑えることが出来た．

4.2 企業開発（保守工程）への適用

我々は 4.1 節の実験において本手法，及び本手法を実装した一連のツールを用いて実際の開発プロジェクトを分析可能であることを確認した．本実験では単に MUA でメールスレッドを巡覧した場合と比較した際の提案手法の優位性を検証するために，ソフトウェア開発企業から提供を受けた大規模なデータに対して提案手法の適用を行った．対象となるデータが大規模であるため，クラスタリングアルゴリズムには計算量の少ない k-means 法を用いた．プロジェクトの詳細を表 3 に示す．なお，守秘義務により開発用メーリングリスト以外の各種メトリクスデータは，別ツールを用いて表示したもののスクリーンショットとなる．そのため，重畳表示ではなくグラフとメールコンテキスト表示（リプレイヤ）を並列に並べて分析を行った．

表3　対象プロジェクト（企業）

プロジェクト	企業開発プロジェクト（保守工程）
開発要員	10 人以上
データ収集期間	およそ 500 日
メールサイズ	65.1MB，38,365 件

実験では，提供されたコード行数グラフに見られる特徴的な遷移に着目した（図 9）．特徴的な遷移を見せているのは，10 月 28 日と 2 月 28 日の 2 箇所で，それぞれ急上昇，及び急降下している．我々は 2 人の被験者（研究者，当該プロジェクトとは無関係）に，この 2 箇所について各被験者 1 箇所ずつ原因を探ってもらった．1 時間程度の調査の結果，以下のように原因を正しく判定することが出来た．

① 10 月 28 日に行数が急上昇した原因

開発データの中央集約化の際に行った CVS 環境の新構築が原因であった．この情報は 10 月 28 日付近のメールコンテキストから確認出来た．また，同じメールコンテキストに含まれるメールから，中央集約化の実施に至った理由が，旧サーバのハードディスク障害であることが判明した．「CVS 環境の新構築」と「旧サーバのハードディスク障害」は時間的に離れた別々のメールスレッドに書かれた情報であるため，単に 10 月 28 日付近のメールスレッドを確認するだけではハードディスク障害について知ることは出来ない．また，本節で分析対象としているメールアーカイブは保守工程という性質上，ユーザからの質問や各種依頼のメールが多数転送されてきており，開発者同士のやり取りとユーザの質問に対する回答のメールが混在している．ユーザからのメールはすべて独立したスレッドであるため，開発者同士のやり取りを確認したい場合はノイズとなる．提案手法を用いた場合，同じような内容の問い合わせで 1 つのクラスタを形成するため，メールスレッドを巡覧するよりも効率良くコンテキストを把握出来ると考えられる．

② 2 月 28 日に行数が急降下した原因

開発言語の変更に伴い，不要となったファイルを一斉削除したことが原因であった．この情報は 2 月 28 日付近のメールから確認出来たが，「なぜ開発言語が変更になったのか」という情報は掴めなかった．

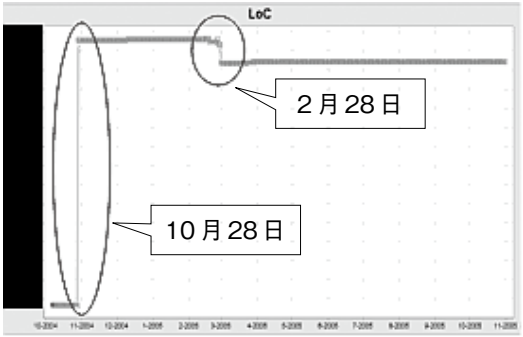


図9　コード行数グラフに見られる特徴的な推移

これら2つの事象に対する「原因」は明確にメールに書かれていたため，推測ではなく事実とした．ただし，これらの事象以外にも原因がある可能性があるため，2箇所の特徴的な推移に対するすべての原因が分析出来たとは言えない．

4.3 オープンソースソフトウェアプロジェクトへの適用

最後に，我々はクラスタリング結果が人にとってどれだけ直感的であるかを検証するための実験を行った．対象は，表4に示すOSSプロジェクトである．実験手順は次の通りである．

① 被験者に開発用メーリングリストを手動で分類してもらう
② 提案手法を用いて同メーリングリストを自動分類する
③ 手動分類と自動分類の結果と比較し，提案手法がどれだけ手動分類を再現出来ているかを調べる

表4　対象プロジェクト(OSS)

プロジェクト	TOMOYO プロジェクト（Linux アクセス制御）
開発要員	20 人
データ収集期間	2005/10/27 ～ 2009/11/16
メールサイズ	2.77MB，1,256 件

一般に，メールを返信した際には返信ヘッダが付与され，メールスレッド構成する．スレッドは既に1つの話題の集合であると考えられるので，我々はこれをメールコンテキストの最小単位とした．被験者（大学院生）には図10のような記入例を示した入力用データシートを渡し，スレッドを更に抽象化した中分類と大分類にメールコンテキストを分割してもらった．結果の一部を抜粋して表5に掲載する．被験者は，当該メーリングリストのスレッドを計24個の中分類と4個の大分類に分類した．本実験では提案手法によって中分類を再現出来るかどうかを検証する．なお，再現出来ているかどうかの判定は，クラスタ一覧を出力したhtml（図5）を被験者に閲覧してもらい，インタビュー形式で行った．クラスタリングアルゴリズムにはWard法を使用した．

【記入例】　※ルート(最初の発言者)のMLナンバーを記入して下さい

大分類	中分類	該当スレッド	キーワード（スペース区切り）
開発	○○機能実装 △△のバグ修正 etc..	392, 882, 501 22, 57, 212, 333 （以下略）	プレビュー jpg 画像 フォーマット ヒープ メモリ 不足 （以下略）
イベント	オフ会の開催 etc..		
アナウンス	リリース情報 etc..		
その他の相談	メンバーの追加 etc..		

図10　入力用データシートの記入例

表5　被験者による手動分類の結果(一部抜粋)

大分類	中分類	該当スレッド	キーワード（スペース区切り）	評価	HTMLとの相違点
開発	ファイルのパーミッション・アクセスに関する話題	751, 756, 776, 830, 831	ファイル パーミッション アクセス 許可	2	クラスタ2456に多く含まれている
	ファイルの置換・移動に関する話題	113, 587, 612	置換 衝突 議論 移動	3	全て大きなクラスタ2409に含まれる
	ツールのポリシーに関する話題	100，200，370，529，540，755，874, 931	ポリシー 環境変数 プログラム パラメータ	2	
	開発のメインラインを定義するための議論	369，438，448，490，522，551，856, 937, 988, 1095, 1103, 1028	ドメイン ディレクトリ構成 名前の由来 css tomoyo 仕様変更	3	369のスレッドの内容が多岐にわたるため，いくつかのクラスタに分散している
	コードのクリーンアップ	553, 662, 682, 687, 836	コード クリーンアップ	3	大きなクラスタ2453に多く含まれている
	メモリの使用量に関する話題	797, 944	メモリ 使用量 ポリシー 制限	1	クラスタ2421
	csstoolsに関する話題	73, 586, 900, 1089	csstools ポリシー ロード	4	csstoolsに関する話題ではあるが，その詳細はそれぞれ異なっている
	次期バージョンに向けての仕様の議論	1，8，84，96，101，561，587，908，1050, 1179	仕様 提案	4	どのような機能を実装するかによって，分類されるクラスタは異なっている
	カーネルのコンパイル	54，88，116，132，143，209，642，710, 757, 780, 813, 840, 1029	カーネル コンパイル テスト リポジトリ 協力者 リリース	2	
アナウンス	リリース情報	14，81，115，137，162，206，250，361，376，533，571，656，675，676，824，846，868，882，884，885，924，927，943，1030，1041，1081，1082，1085，1092，1177，1200, 1202, 1203	パッチ コンパイル リリース 変更点 サポート規則 アップデート	3	リリースは各機能の改善と一緒のクラスタに含まれていた
	バグ報告	8，368，383，794，818，852，925，1077, 1104, 1109	スペルミス バグ 報告 修正 特有	4	バグ報告はそれぞれの機能と結びついている
	ドキュメント修正に関する話題	350, 509, 826, 896, 915	GUI インストールドキュメント 文字コード	4	ドキュメントの修正も個々の機能の話題と結びついている

表6　インタビュー結果

質問	手動で分類した中分類に合致するクラスタが存在したかどうかを，以下の評価基準に従って各項目ごとに評価してください．
評価1	ほぼ合致するするクラスタが存在する
評価2	部分的に合致するクラスタが存在する
評価3	どちらともいえない
評価4	合致するクラスタは存在しない

回答	件数
評価1	2
評価2	6
評価3	8
評価4	8

閾値は数回の試行によって得られた中分類に適した値を手動で設定した．被験者の評価結果を表 6 に示す．ここで,「部分的に合致するクラスタ（表 6 の評価 2)」とは，手動で分類したスレッドのうちおおむね半分以上が含まれているクラスタを指す．

評価 1，評価 2 を正解とすると，提案手法は手動分類結果の 33.3%（8/24）を抽出出来ているものの，出力されたクラスタの総数は 56 となっており，提案手法が手動分類と十分に適合しているとは言い難い．しかし，4.2 節のケーススタディが示すように，提案手法は膨大な数のメールアーカイブに対して分析に有用なレベルの分類結果を返しており，本評価結果は手法の有用性を否定するものではない．とりわけ，時間的コストの観点から評価すると，被験者は 1,256 件のメールを手動で分類する作業におよそ 6 時間を費やしているのに対し，本手法は小規模なプロジェクトであれば数秒から数十秒，大規模（10,000 件～）なメールアーカイブであっても数分から数十分の間に分類を終わらせることが可能（PC スペック CPU：AMD Athlon 64 X2 4800+, RAM：2GB の場合）であるため，時間的コストにおけるアドバンテージは極めて高く，分析の糸口として用いるには十分に有用であると言える．今後，精度の改善を行うことで更に有用性の向上が期待出来る．

4.4 妥当性

本節では 3 つのケーススタディについて述べ，それぞれのケースにおいて提案手法の有用性を確認するための実験を行ってきた．しかし, いずれの実験も被験者の数が少なく，結果を一般化することは出来ない．

また，メールの分類にかかるコストについては手動分類との比較を行ったが，分析にかかるコストについては比較を行っていないため，検証が必要である．具体的には，単純にメールクライアント等を用いて分析した場合と，提案手法を用いた場合とで時間的コスト，対象プロジェクトへの理解度等にどのような差が出るかを検証する必要がある．

5 まとめと今後の展望

本論文ではソフトウェア開発プロジェクトのコンテキストを，メールアーカイブのクラスタリングと，クラスタの時系列プロットによって可視化する手法を提案した．また，提案手法を実現するためのツールを開発し，3 つのプロジェクトに対して手法を適用し，その有効性について検証した．本論文では第 1 節にて，定量データのみを用いた事後分析には以下のような問題があると述べた．

・観測事象に対応するコンテキストの特定が困難
・ヒアリング対象者の特定が困難
・開発者の記憶があいまい

以上のそれぞれについて，ケーススタディの中で解決出来たこと，解決出来なかったことについて次にまとめる．

① 観測事象に対応するコンテキストの特定が困難

4.2 節で述べた企業開発データへの適用実験において, ソースコード行数が特徴的な遷移を見せた 2 箇所についてそれぞれ原因を明らかにすることが出来た．しかし，明らかになった 2 箇所の原因のうち，1 つは本手法を用いずとも該当日付周辺のメールスレッドを確認することで明らかに出来た．よって，この結果をもって「単にメールを巡覧したときと比べて，本手法を用いた分析の方が常に優れている」と言うことは出来ない．しかし，メールの件数が膨大になればなるほどスレッドの巡覧にかかるコストは増大する．本手法はクラスタリングの際に閾値を調整することによって，スレッドよりも抽象的な範囲で話題を抽出することが可能であり，数年にわたるような長期プロジェクトであっても短期間でメールコンテキストの把握が可能である．

② ヒアリング対象者の特定が困難

4.1 節で述べた小規模プロジェクトの分析において，被験者は本手法を用いて各プロジェクトメンバの役割を把握出来ていた．分析対象のメーリングリストには，当該プロジェクトメンバ以外から送信されたメールも多数含まれていたが，ミーティング調整やコード修正アナウンスのメールコンテキストから，誰がプロジェクトメンバであるかを判断することが出来た．このように開発者の役割が判断出来ていれば, ヒアリング対象者を絞ることが容易となる．ただし，4.1 節は小規模なプロジェクトのみの検証であるため，一般性を述べるためには適用プロジェクトのサンプルを更に増やす必要がある．

③ 開発者の記憶があいまい

4.1 節にて行ったプロジェクトメンバへのインタビューは 2009 年に行われたが，当該プロジェクトは 2005 年に実施されており，プロジェクトメンバは開発の詳細を部分的に忘れていた．しかし，インタビューの際にクラスタリング結果の html を閲覧することで詳細を思い出すことが出来た．ただし，これについても当該プロジェクトが小規模であるため，提案手法を用いずともメールスレッドの巡覧のみでプロジェクトを回顧出来た可能性が高いため，この点をもって手法を一般化することは出来ない．しかし，メールスレッドから更に抽象化されたメールコンテキスト群は話題の俯瞰に適しており，MUA を用いたメールスレッドの巡覧よりも短時間で回顧が可能であると考えられる．

以上のように，本手法の一般性は今後の追加実験によって更なる検証を行う必要があるものの，現時点においても一定の有効性は確認出来たと言える．また，本論文で述べた3つのケーススタディは異なるスケール（小規模～大規模）のプロジェクトを対象としており，本手法が一定のスケーラビリティを保していることを示している．今後は更に大規模な実プロジェクトへ適用し，本手法がその対象規模にかかわらず適用可能であることを示す必要があると考えられる．

本手法は開発形態や組織形態にかかわらず，様々な実開発プロジェクトに適用可能であると考えられるが，プロジェクトによってメールの利用目的は様々であり，どのような開発においてどのようにメールが利用されているかの調査が必要である．また，手法の精度向上，及び信頼性向上のために適用サンプルを増やしていく必要がある．しかし，多くの開発組織は機密保持を理由に開発データを外部に提供しないという現状があるため，今後は産学の連携をより一層強めていくことが求められる．

また，索引語抽出のアルゴリズムや重み付けの調整についても，より人間にとって直感的なコンテキストの分類が出来るよう手法を改善していこうと考えている．更に，分析にかかる時間的コストをより小さくするために，閾値別の分析結果をシームレスに切り替えて表示するためのインターフェースを用意していきたいと考えている．

謝辞

本研究の一部は，文部科学省「次世代IT基盤構築のための研究開発」の委託に基づいて行われた．また，本研究は日本学術振興会特別研究員奨励費の助成を受けたものである．

参考文献

[ANVIK2006] John Anvik, Lyndon Hiew, and Gail C. Murphy : Who should fix this bug? In ICSE '06 : Proceedings of the 28th international conference on Software engineering, pp.361-370, New York, NY, USA, 2006, ACM

[BIRD2006] Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan : Mining email social networks. In MSR '06 : Proceedings of the 2006 international workshop on Mining software repositories, pp.137-143, New York, NY, USA, 2006, ACM

[BUGZILLA] http://www.bugzilla.org/

[CHIDAMBER1994] S. R. Chidamber and C. F. Kemerer : A metrics suite for object oriented design, IEEE Trans. Softw. Eng., Vol.20, No.6, pp.476-493, 1994

[GNATS] http://www.gnu.org/software/gnats/

[KAMIYA2002] Toshihiro Kamiya, Shinji Kusumoto, and Katsuro Inoue : Ccfinder : a multilinguistic token-based code clone detection system for large scale source code, IEEE Trans. Softw. Eng., Vol.28, No.7, pp.654-670, 2002

[KLEINBERG2002] Jon Kleinberg : Bursty and hierarchical structure in streams, pp. 91-101, ACM Press, 2002

[LIWC] http://www.liwc.net/

[NIKKEI2008] 中村 建助，矢口 竜太郎：2008年情報化実態調査プロジェクトの成功率は31.1%，日経コンピュータ 2008年12月1日号，pp.38-53，December 2008

[OHKURA2006] Kimiharu Ohkura, Keita Goto, Noriko Hanakawa, Shinji Kawaguchi, and Hajimu Iida : Project replayer with email analysis - revealing contexts in software development, In Proceedings of the 13th Asia Pacific Software Engineering Conference, pp.453-460, December 2006

[QUENTIN2006] Quentin W. Fleming and Joel M. Koppelman : Earned Value Project Management, Project Management Institute, 2006

[RIGBY2007] Peter C. Rigby and Ahmed E. Hassan : What can oss mailing lists tell us? a preliminary psychometric text analysis of the apache developer mailing list, In MSR '07 : Proceedings of the Fourth International Workshop on Mining Software Repositories, p.23, Washington, DC, USA, 2007, IEEE Computer Society

[SALTON1975] G. Salton, A. Wong, and C. S. Yang : A vector space model for automatic indexing, Commun, ACM, Vol.18, No.11, pp.613-620, 1975

[SALTON1987] Gerard Salton and Chris Buckley : Term weighting approaches in automatic text retrieval, Technical report, Ithaca, NY, USA, 1987

[SARMA2009] Anita Sarma, Larry Maccherone, Patrick Wagstrom, and James Herbsleb : Tesseract : Interactive visual exploration of socio-technical relationships in software development, In ICSE '09 : Proceedings of the 2009 IEEE 31st International Conference on Software Engineering, pp.23-33, Washington, DC, USA, 2009, IEEE Computer Society

[SEN] Sen Project : https://sen.dev.java.net/

[崔 2004] 崔 春花，北川 博之：到着頻度と関連性を考慮した時系列文書の連続的トピック分析（時系列とコンテンツ）（夏のデータベースワークショップdbws2004），情報処理学会研究報告，データベース・システム研究会報告，Vol.2004，No.72，pp.315-322，20040714

[データ白書 2009] IPA/SEC：ソフトウェア開発データ白書2009 ～2327プロジェクト　定量データ分析で分かる開発の最新動向～，日経BP社，2009

[中山 2008] 中山 祐貴，宮寺 庸造，横山 節雄，中村 勝一：電子メール中の議論過程抽出手法の提案（教育・学習評価／一般），電子情報通信学会技術研究報告，ET，教育工学，Vol.108，No.354，pp.35-40，20081206

[仁野 2008] 仁野 裕一，野田 潤，中尾 敏康：特徴語の共起性を利用した携帯電話メールの関係性検出手法（ライフログ活用技術とその課題，オフィス情報システム，デジタルドキュメント，一般），電子情報通信学会技術研究報告，OIS，オフィスインフォメーションシステム，Vol.108，No.156，pp.71-76，20080717

日本アイ・ビー・エム株式会社 CoBRA法を活用し、組織として統一された見積りモデルを実現

SEC journal 編集部

IPA/SECは、ドイツ・フラウンホーファ協会実験的ソフトウェア工学研究所（IESE※1）が開発した見積りモデルを構築する手法である「CoBRA※2法」の研究成果を「ソフトウェア開発見積りガイドブック」[SEC2006]の中に収めると共に、CoBRA法を簡易に利用可能とする「CoBRA法に基づく見積り支援ツール」を開発し、SEC Webサイトで公開している[CoBRAツール]。日本アイ・ビー・エム株式会社（以下、日本IBM）は、同ツールを利用することによって、組織として統一された見積りモデルの構築に成功している。同社がCoBRA法を導入した背景、成功のポイントを中心に、CoBRA法の導入を担当した酒井大氏と吉竹正貴氏に話を伺った。

参考となるプロジェクトデータが少なくても モデルが構築出来ることに注目

日本IBMにおいてCoBRA法の利用を進めているのは、同社の社内情報システムの開発・保守を担当しているIGA AS※3という組織である。プロジェクトの工数を見積る手法として、IGA ASではプロジェクトをタスクに分割して工数を積み上げるボトムアップ法（積み上げ法）を用いている。CoBRA法を導入したのは、ボトムアップ法によって導き出された見積りと実際の工数に乖離があったからではないと言う。同組織では、プロジェクトマネージャによってタスクの分割の仕方が異なる等、見積りのアプローチに違いが存在しており、社内のお客様からは見積り手法が統一されていないと受け取られていたという。そこで、「組織として統一された見積りモデルを実現したい」（酒井氏）と、見積りに関する情報収集を行い、CoBRA法に着目した。CoBRA法に注目した理由は、参考となる過去のプロジェクトデータの数が少なくても見積りモデルとして構築・利用出来ることが挙げられ、そのことがIGA ASという組織に合致すると判断された。

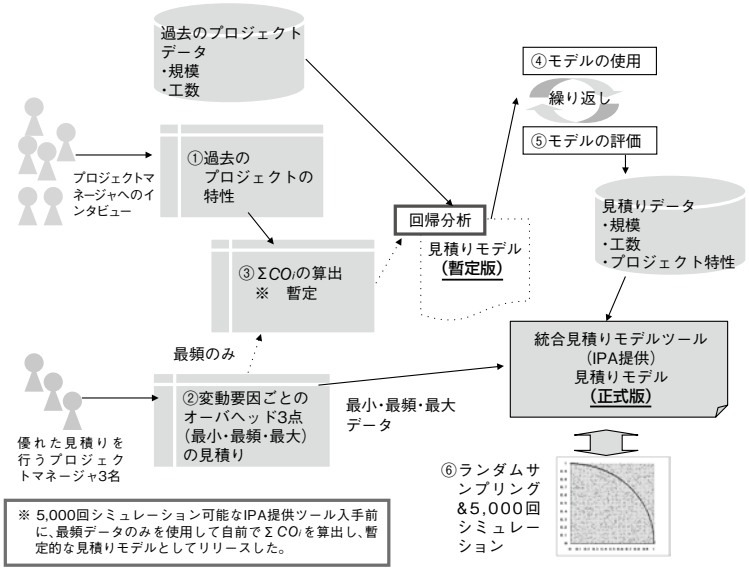


図１　見積りモデル構築手順

CoBRA法を導入の準備段階として行ったのは、プロジェクトマネージャにインタビューし、(a) 実施済みのプロジェクトデータ（プロジェクトの特性）を収集すること、(b) プロジェクトにおける変動要因を洗い出すことである（図1）。プロジェクトの規模データはFP※4を採用した。また、とくに優れた見積りを行う3名のプロジェクトマネージャにインタビューし、変動要因ごとのオーバヘッドの予測を得た。変動要因として選定したのは、20項目にわたる（図2）。これら変動要因の影響度の総和は$\Sigma\, CO_i$と表記され、プロジェクトの工数を定量的に見積るためのデータとなった。

- ・要件の確定度　・プロジェクトマネジャのスキル
- ・開発者のスキル　・顧客のレスポンスのスピード
- ・ステークホルダの数　・日本以外のステークホルダの数
- ・既存アプリケーションに関するドキュメントの整備状況
- ・既存アプリケーションの規模　等

図２　変動要因の項目

SECの見積り支援ツールをカスタマイズして モデルを構築

同組織はまず、変動要因ごとのオーバヘッドのうち、「最頻」のみを用いて工数の変動率を算出する暫定版の見積りモデルを自社で作成し、パイロット的にCoBRA法による見積りを開始した。2010年3月にIPA/SECよりSEC Webサイトで公開された「CoBRA法に基づく見積り支援ツール」のうち「統合見積りモデルツール」を、2010年6月から正式版として利用している。

同組織の見積り支援ツール活用の特徴は、IPA/SECのツールをベースとしつつも同組織における

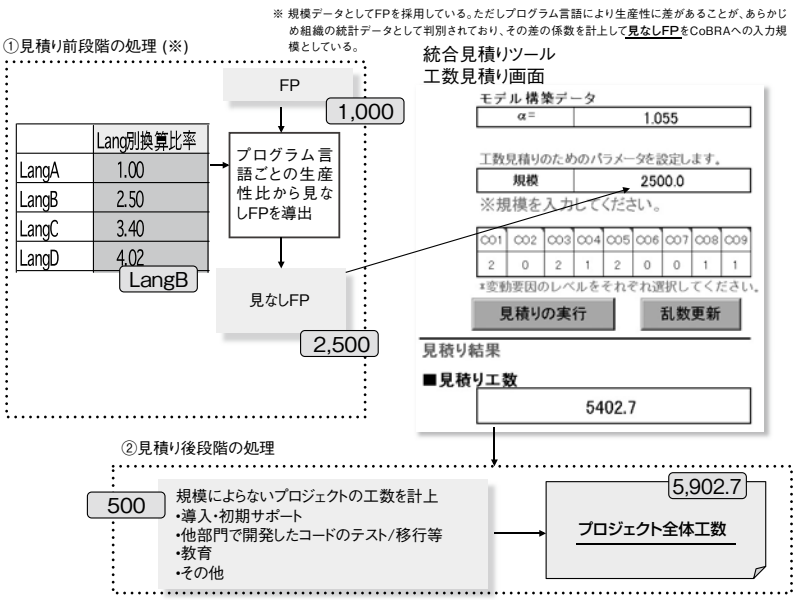


図３　カスタマイズ内容

従来の開発プロジェクトの経験を生かして２つのカスタマイズを行ったことである(図3)。1つ目のカスタマイズは、プログラム言語によって「見なし」FP を用いて規模を見積ることにしたことだ。その理由は、「プログラム言語によって生産性に差がある」（酒井氏）からである。そのため､プログラム言語ごとに「見なし」FP の係数を設定して、その係数を工数見積りのパラメータとしたのである。

２つ目のカスタマイズは、CoBRA が対象としていない、(a) 導入・初期サポート、(b) 他部門が開発したコードのテストや移行、(c) 教育等、規模に依存しない作業にかかる工数の見積りが行える機能を加えたことである。

そして、これら工数と、先の FP に基づいた見積り工数と加算することにより、プロジェクト全体の工数が見積られる仕組みを実現したのである。

セカンダリの見積り法として位置付ける

日本 IBM における CoBRA 法運用には重要な前提がある。それは基本原則として、ボトムアップ法をプライマリの見積り法として位置付け、CoBRA 法はセカンダリの見積り法と位置付けて両者を使用しているということである。これは前述したように、もともとボトムアップ法による見積り精度に問題があったわけではないことがその理由である。

CoBRA 法の運用結果は期待以上だった。まず、暫定版の見積りモデルを利用したケースでは実績の工数との誤差は±10%ほどだったという。SEC の成果をベースにしたツールを利用しはじめてからは、誤差が±22 ～ 23%とやや拡大したが、セカンダリの見積りモデルとして使うには十分な数値という。

大切なことはプライマリとセカンダリのギャップ分析をすること

大切なことは、「プライマリとセカンダリのギャップ分析をすること」（酒井氏）だという。酒井氏らはその考えに基づいてギャップ分析のためのガイドを作成している。そのガイドはボトムアップ法による見積りの仕方そのものを改善することにつながる。例えば、ユーザのレスポンスが悪いことがオーバヘッドとなっていると分かった場合、それを考慮してタスクを設定したかを確認することを推奨している。CoBRA 法活用ガイドは、その使い方を提示するにとどまらず組織としての見積り精度を一層向上させることを視野に入れたものと言える。

IGA AS の組織内で行った満足度アンケート調査で、10 人中 6 人が満足いく施策に CoBRA 法を挙げた。「組織として見積りを客観的に検証するモデルが統一されたことが評価されたと考えています」（吉竹氏）。

今後、同組織が取り組もうと考えているのは、CoBRA 法による見積り精度を更に向上させることだ。そのために、「実績データを蓄積すると共に、定期的に変動要因の見直しを図って更に適切なものにしていきたい」と酒井氏は語る。また、CoRBA 法の運用ノウハウをアプリケーションの改変・保守を事業とするアプリケーションマネジメントサービス（AMS）という組織や、IBM においてグローバルに見積りを支援する組織にフィードバックすることも視野に入れている。日本 IBM における CoBRA 法の活用は更に広がろうとしている。

脚注

※１ IESE：Institute for Experimental Software Engineering, 実験的ソフトウェア工学研究所
※２ CoBRA：Cost Estimation, Benchmarking, and Risk Assessment
※３ IGA AS：IGA アプリケーション・サービス
※４ FP：Function Point, ソフトウェアの機能規模を測定する手法の１つ。開発工数の見積りに利用される。

参考文献

[CoBRA ツール] IPA/SEC：CoBRA 法に基づく見積り支援ツール, https://sec.ipa.go.jp/
[SEC2006] IPA/SEC：SEC BOOKS 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～, オーム社, 2006

会社情報
商　号　日本アイ・ビー・エム株式会社
設　立　1937 年 6 月
本　社　〒 103-8510　東京都中央区日本橋箱崎町 19-21
資本金　1,353 億円
代表者　代表取締役　社長執行役員　橋本　孝之
社員数　非公開
事業内容　情報システムに関わる製品、サービスの提供

独立検証及び妥当性確認と形式手法がもたらすソフトウェア開発プロセスの高信頼化

名古屋大学　情報連携統括本部
情報戦略室　教授　工学博士
山本　修一郎

重要なシステムの事故が後を絶たない。このため、ソフトウェアの独立検証及び妥当性確認（IV&V※1）が注目されている。本稿では、ソフトウェア独立検証及び妥当性確認の重要性を示すと共に、形式手法が持つ客観性がIV&Vでも必要になることを示す。また我が国における独立検証及び妥当性確認の先駆的な事例を紹介し、これらの事例に共通する「ソフトウェア開発プロセス改革」の核心を明らかにする。

1. はじめに

重要システムには、重要安全システム、重要基幹システム、重要事業システム、重要セキュリティシステムがある[LORCAN2010]。重要安全システムの障害では、生命や人体の負傷あるいは自然環境への被害が発生する危険性がある。重要基幹システムの障害では、重要基盤システムやデータが消失することで、システム全体の実行やプロジェクト目標の達成が出来なくなる危険性がある。重要事業システムの障害では、事業の損失や評判の失墜等の著しい経済的損失が発生する危険性がある。重要セキュリティシステムの障害では、盗聴や事故によって機密情報が改ざんされたり消失する危険性がある。

これらの重要システムで、IV&V の取り組みが必要になることは明らかだろう。この理由は、もしこれらの重要システムで事故が発生したときに、その原因が追究され、十分な説明責任が求められるからである。

以下では、まず、ソフトウェア検証確認についての標準を定めた IEEE Std 1012-2004 に基づいてソフトウェア検証確認プロセスのポイントを紹介する。次いでなぜ IV&V が必要なのかについて解説する。また形式手法と IV&V の関係について述べる。

2. ソフトウェアV&Vとは

IEEE Std 1012-2004 ソフトウェア検証確認 [IEEE SWIV&V] によれば、ソフトウェア検証確認プロセスでは、開発活動による生産物が開発活動に対する要求に適合していることの検証と、開発されたソフトウェアが意図された利用法とユーザニーズに適合していることを判定する確認が必要になる。

2.1 検証

ソフトウェアを正しく作っていることを判定するのが検証（Verification）である。

すなわち、「与えられた開発工程の生産物としてのソフトウェアとコンポーネントが、その工程の開始時点で定められた条件を満たすことを判断する評価プロセス」が「検証」である。

検証活動では、ソフトウェアと関連生産物がライフサイクルプロセスの過程で、次に示す客観的な証跡を提供する必要がある。

・すべての活動に対する要求に適合すること
・標準、実践規約、慣例を満たすこと
・各活動を完了すること
・後続工程を開始出来るすべての基準を満たすこと

2.2 確認

ソフトウェアが正しいことを判定するのが確認（Validation）である。

すなわち、「ソフトウェアとコンポーネントが開発プロセスの完了時までにユーザのニーズとしての要求を満たすことを判断する評価プロセス」が「確認」である。

確認活動では、ソフトウェアと関連生産物がライフサイクル活動の完了時に、次に示す証跡を提供する必要がある。

・ソフトウェアに割り付けられた要求を満たすこと
・正しい問題を解決すること

2.3 独立検証及び妥当性確認 IV&V とは

ソフトウェア開発組織と、技術面、管理面、財政面で独立な組織によって検証及び妥当性確認を実施することを IV&V という。

対象プロジェクトの開発経費とは別に、全社的な観点から IV&V 活動に投資することによって財政面の独立性を確保出来る。

IV&V 標準作業構造に基づいて、対象プロジェクトに対して IV&V を実施するスコープとタスクを定義することによって、技術面での独立性を確保する。対象プロジェクトと独立な作業標準に基づく技術活動を実施することで IV&V 活動の客観性を保証する必要がある。

対象プロジェクトの開発管理とは別に IV&V 活動をす

る独立な組織を設けることで、管理面での独立性を確保することが出来る。例えば、NASAのIV&Vでは、OSMA※2によるIV&V活動に対する管理が実施されている[NASA IV&V]。

2.4 なぜIV&Vなのか？

まず、システムの信頼性には本質的な限界があることを認識しておく必要がある。

① システムに欠陥が無いことをテストによって完全に保証することは出来ない

この理由は、システムに欠陥が無いことをテストによって保証しようとしても、あらゆる場合を想定してすべてを検査することは出来ないからである。このため、テスト活動の現場では、システムに対してあらかじめ定められたバグ摘出基準を満たしたかどうかによって判断することになる。従ってシステムにはいつも何らかの欠陥が残っている可能性がある。

この結果、次の逆説が得られる。

② システムの信頼性が高いほど、重大な欠陥の発見が遅れる

この理由は、もしシステムの信頼性が高くなければ、システムの欠陥が早期に発見されるからである。またシステムの欠陥があまりに多ければ、そのシステムは使われなくなるから、重大な欠陥が発生することも無い。つまりシステムの信頼性が高ければ高いほど、重大な欠陥が発生するまで継続的に運用され、このシステムの信頼性は高いと信じられることになる。この結果、信頼性が高いと信じられていたシステムに重大な欠陥が発見されたときには、このシステムを開発したときよりも進んだ技術が世の中で利用されている可能性が高くなる。従って、次のような事態が発生することになる。

③ 最新の開発技術に基づいて、過去のシステム開発の妥当性が判断される

この結果、信頼性が高いと思われていたシステムに重大な欠陥が発見されると、何が起こるかは容易に想像がつく。つまり、こういうことになる。開発時には最高水準の技術を適用して苦労して完成させたシステムが、ある日を境に欠陥システムとなり、開発時に十分な水準の技術を適用せず手抜き工事を実施したとして重過失責任を問われることになるのである。そのシステムの提供者や開発者が、自ら、開発時には最高水準の技術を適用したことをどのようにして立証出来るだろうか？　また、これらの当事者がそれを立証したとして、客観性がどれほど認められるだろうか？

例えば、サービスが開始されてから10年間は重大な欠陥が発生せず、11年目に欠陥が発生して、社会問題になったとしよう。そうするとサービス開始以前にシステムを開発していることになるから、10年以上前のシステム開発の妥当性が社会的に問われることになる。技術進化の激しいITの世界での10年と言えば、大きな技術革新が何度も起きている。また10年前の技術水準がどのようであったか等、研究論文では追跡可能であっても、開発現場がどうだったか追跡するのは困難である。となると、社会的には、問題が発見された時点での技術水準で過去の開発のあり方が糾弾されることになる可能性が高い。従って、この場合システム提供者は相当不利な状況に追い込まれることになるのは間違いない。この理由は、システム開発の当事者の主張だけでは、客観的な証拠として認められにくいからだ。つまり、事故が起きた後では、残念なことだが次のようになるのは当然である。

④ 客観的な証跡が無い限り、当事者がシステムの信頼性をいくら主張しても信用されない

このような理不尽な事態を回避するためには、開発時に、システムの信頼性を第三者によって客観的に検証して証跡を残すしか方法が無い。開発が完了した後で検証して欠陥が発覚しても後の祭りになるかも知れない。それでは手遅れである。だから開発時に同時に検証する必要がある。これから、以下のことが導かれる。

結論：第三者による開発時の検証及び妥当性確認によって、システムの信頼性を客観的に保証する必要がある

3. 検証確認の形態と責任

事故が発生したシステムが、開発時点でどのような検証確認が実施されていたかについては2つの場合がある。それは、システムに対して、計画した通りの検証確認を実施していた場合とそうでない場合である。

そもそもシステムに欠陥が全く無いことを保証することは現実的に不可能であるから、前者の場合のように、計画通りに検証確認を実施していても、事故が発生する可能性がある。「どのような事故も発生しないように検証確認していなければ、システム提供者には重過失責任がある」と非難するのは、「不可能なことをやれと言っておいて、それが出来なければおまえは魔女だ」と断定するのと同じくらい非科学的である。この場合には、当然、免責されるべきである。

しかし、後者の場合には、開発者が「計画した通りに検証確認をやります」と言っておきながら、実際にはやらなかったために事故が発生したことになる。計画には挙げていたが、開発完了期限が迫るというプレッシャのために、プロジェクトマネジャが計画ではやるべきだった検証確認を省いたことになる。これは建設工事で言えば「手抜き工事」である。広辞苑第六版によれば、「手抜き」とは、「しなければならない手続・手数を省くこと」のことだ。このような事態が日本のソフトウェア開発現場で発生しているとしたら、日本の情報産業だけでなく、日本社会全体にとって大変危険な状態であると言わざるを得ない。また、「管理」

脚注

※1　IV&V：Independent Verification and Validation，独立検証及び妥当性確認
日本語訳として、①独立検証確認、②独立検証及び妥当性確認、③正当性検証及び妥当性確認の3案があるが、SEC journalでは一般の表記に倣い②とした。

※2　OSMA：Office of Safety and Mission Assurance

とは、やはり広辞苑第六版によれば、「管轄し処理すること。良い状態を保つように処置すること。とりしきること」である。やるべきことをやらなかったプロジェクトマネジャの管理責任が問われることになる。

それでは、検証確認の実施内容が、前者なのか後者なのかを誰がどのようにして判別するのかということが次の問題になる。当事者が「ちゃんとやりました」ということを信じられるかということである。「なるほどね。あなたがそう言うのだから信用しよう」となることもあるかも知れない。しかしすぐに当事者の範囲が問題になることは間違いない。システムの発注者と開発者以外のところから事故が発覚すると、事故の被害者は「発注者と開発者が申し合わせてうまくやったのじゃないか」ということに当然なるだろう。つまり、前者と後者の検証確認を判別するためには、①当事者による客観的な検証確認結果を示す証跡か、②当事者以外の第三者による客観的な独立検証及び妥当性確認（IV&V）の証跡が残されていることが必要になる。これらの客観的な検証確認の証跡があることで、当事者以外の第三者がやっても、そこまでしか検証確認出来なかったことが裏付けられるから、確かに「計画された検証確認が実施されていた」ことが初めて証明される。それが無ければ議論は平行線となって結論が出ないことや、「計画した通りに検証確認を実施していたにもかかわらず、検証確認をしていなかった」という誤った結論が導かれることも十分考えられる。このように客観的な検証確認結果の証跡は、検証確認の妥当性を示す根拠を提供して開発者を守るだけでなく、客観的な検証確認を開発時に組み込むことで、手抜き工事を未然に防ぐことが出来る点も重要である。

4. IV&Vの先駆的事例

我が国におけるIV&Vの先駆的な事例を示す。表1には、取り組み内容とフィードバック内容を示す。

4.1 IV&V と形式手法

事例1と事例2は形式手法とIV&Vを併用した例である。事例3はIV&Vを実施するが形式手法を適用していない例である。事例3で求められた世界最高水準の性能がその時点の技術で実現出来るかどうかについては専門家の判断が必要である。

形式手法を適用することの意義は、実施者が科学的な手法を用いることによって検証確認結果の網羅性や客観性が高まることである。同様に、第三者機関を活用することの意義も、客観性の高い検証確認結果を残すことで発注者責任を果たす点にある。

4.2 フィードバック型開発プロセス

紹介した3つの事例の開発プロセスでは、いずれも不具合を下流工程に残さないために、独立検証による工程内フィードバックと工程間フィードバックを導入している。

従来のV字モデルでは障害が先送りされテスト工程で検出されていたが、これらの先駆的事例では可能な限り早い段階で障害を発見し、自工程や直前の工程にフィードバックする開発プロセスに進化している。これらの事例では、いずれも高信頼性が要求されるシステム開発に対して、共通にフィードバック型開発プロセスを、互いに独立に採用していた点が注目される。

5. IV&Vの論点

5.1 冗長ではないのか

IV&Vは開発活動とは独立な活動となることから、冗長な活動であり無駄ではないかという指摘が予想される。開発だけを考えるなら指摘通り冗長な活動である。しかし、運用時のリスク対策を含めて考えれば、IV&Vは必要な活動である。既に述べたように、システムの事故が発生したときにシステム開発が妥当であったことを示す証跡が無ければ、重過失責任が問われても客観的に反論出来ないからである。とくにグローバル展開を考える企業であれば、グローバル環境でシステムを運用したときの事故対策として、IV&Vに取り組むことが冗長だとは思えないはずだ。事件が北米で発生したらどうなるかを考えれば明らかである。必要なのは当事者が高信頼システムを開発出来る能力を示すことではなく、開発されたシステムについての客観

表1　我が国におけるIV&Vの先駆的な取り組みと各事例のフィードバック内容

事例	取り組み内容	フィードバック内容
事例1 独立行政法人 宇宙航空研究開発機構（JAXA）	開発メーカ内部の独立組織によるIV&Vと、JAXA内部の別組織による形式手法によるモデル検査を適用して検証確認を実施している[JAXA2007]。	検証部門が課題や問題を発見すると、開発部門にその内容をタイムリーにフィードバックしている。
事例2 フェリカネットワークス株式会社	公的なセキュリティ評価・認証の枠組みに基づいて、第三者機関による形式手法を用いた検証確認を実施している[KURITA2010]。フェリカネットワークスでは、更にその検証結果を検証者とは別の第三者が認証することで検証結果を保証している。	各工程内で段階的作業による工程内フィードバック、工程間フィードバック、第三者による評価認証結果の開発部隊へのフィードバックを実施している。
事例3 株式会社東京証券取引所	外部機関が非機能要件を第三者検証している[TAKURA2010]。とくに、開発時に実在している技術及び実現可能と考えられる技術で世界最高水準の性能の実現が可能かを見極めるために発注者責任として第三者の見解を確認した点が重要である。	工程ごとに障害発生件数を設定しておき、許容下限値に接近すると品質強化対策を実施する「リアルタイム品質管理」と、後続工程で先行工程の障害を積極的に検出して先行工程にその内容をフィードバックすることで先行工程の不備を能動的に発見している。

的な根拠としてのIV&Vの証跡である。

それでも、事故が発生してからIV&Vをやっても遅くないのではないかという指摘もあるかも知れない。しかし技術は必ず進歩しているから、事後のIV&Vのほうがより多くの欠陥を発見出来る可能性が高いのは当然である。従って事後に実施されるIV&Vの結果は、開発対象システムに対して不利に働く可能性が高い。だから、開発時にIV&Vを実施しておくことは冗長ではなく、むしろ将来のリスクに対してどうしても必要なことなのである。

また、IEEE Std 1012-2004では次のような効果が期待出来ると指摘している。IV&Vの取り組みはこれらの効果を達成するための統合的な手法を提供していることになる。

・ソフトウェア異常の検出と修正の早期化
・開発プロセスと生産物のリスクに対する客観的な管理の推進
・生産性、工期、予算に対してライフサイクルプロセスの適合性
・ソフトウェアとシステム性能の影響評価を早期化
・ソフトウェアとシステムの適合性に対して客観的な証跡に基づく、公式検定プロセスを提供
・開発保守プロセスを改善出来る
・システム開発を統合的に分析出来るモデルを提供することでプロセスを改善出来る

5.2 第三者への情報提示は開発側に不利になるのではないのか

もし開発対象システムに対する信頼性について開発者に十分な自信があれば、第三者が検証確認しても品質に問題が出る可能性は少ないし、もし欠陥が摘出されたとしたら、早期に修正出来るのだから手戻りを抑制し、開発効率が向上することになるから、むしろ歓迎すべきことである。

例えば、医療サービスを考えてみよう。死因の判定に納得のいかない遺族が、別のお医者さんに亡くなった家族のMRI画像を見てもらって、その意見を聞いて初めて納得するケースがあるという。この例は、事後にMRI画像によって医療行為の結果を検証するということだが、医療行為と並行して第三者による医療行為の正当性を検証することが出来れば、より信頼性の高い医療サービスになると考えられる。つまり、医療行為の正当性を確認するために必要な情報を記録しておき、第三者に証明してもらうことを可能とすることで、医療機関の信頼性を向上出来る。

同じことがソフトウェア開発でも成立しない理由は無い。つまり第三者による検証確認をしておくことがむしろ開発者を守り、信頼度を高めることになる。

5.3 第三者検証機関の責任はどうなるのか

第三者検証機関が検証確認したとしても、欠陥が完全に無いことを保証出来ないから、不具合が発見される可能性が残る。もし第三者検証確認機関が作業した結果に対して不具合が発見されたときに、その責任はどうなるかという問題がある。正しいプロセスを実施したことであって、不具合が無いことを保証することではない。もし指定されたプロセスを第三者検証機関が実施していなければ、当然責任が問われることになる。従って第三者検証機関が実施すべき標準プロセスが定義されていることと、開発対象プロジェクトの状況に応じてどの範囲で検証確認を実施すべきかについての発注者側の判断が必要になる。つまり、標準プロセスと発注者の判断に対して誠実に検証確認を遂行することによって第三者機関の責任が不具合に対して免責されると考えられる。もし標準プロセスと発注者判断に基づいて必要な検証確認作業を実施していなければ、第三者機関の責任が問われることは当然である。

6. まとめ

ソフトウェアがビジネスや社会にとって重要な存在になっているということは、ソフトウェアの欠陥がビジネスや社会に重要な影響をもたらす、つまり経済的な被害が想定される。にもかかわらず、経済的社会的被害を想定したソフトウェア開発プロセスになっていないのが現状ではないだろうか？　重要なビジネスの中核となるソフトウェアに対する十分なリスク対策の取り組みが早急に求められている。本稿で紹介した独立検証及び妥当性確認（IV&V）手法や形式手法はビジネスや社会におけるソフトウェアリスクを軽減するのに役立つはずである。このような独立検証及び妥当性確認プロセスを確立することが出来れば、ソフトウェア開発を可視化出来るだけでなく、ソフトウェアリスクを客観的に管理出来るようになることでもある。

今後、第三者機関による独立検証及び妥当性確認や形式手法の取り組みが我が国においても進展し定着することを期待したい。そのときの課題は、独立検証及び妥当性確認が出来る第三者機関とはどういう組織なのかということだ。そのためには独立検証及び妥当性確認組織を公的に認証する仕組みを確立することが求められる。

参考文献

[IEEE SWIV&V] IEEE Std 1012-2004 Software Verification and Validation

[JAXA2007] 宇宙航空研究開発機構：ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスをめざして～，経済産業省，プロセス改善研究部会，2007，http://sec.ipa.go.jp/reports/20070514/JAXA.pdf

[KURITA2010] 栗田 太郎：モバイルFeliCaのソフトウェア開発における品質確保のための構造と実践 ～抽象度の制御やコミュニケーションの活性化に向けて，情報処理学会デジタルプラクティス，Vol.1，No.3，2010，http://www.ipsj.or.jp/15dp/Vol1/No3/dp0103.html

[LORCAN2010] Lorcan Coyle，et. al：Guest Editors Introduction：Evolving Critical Systems, IEEE Computer, vol.43, no.5, pp.28-33, 2010

[NASA IV&V] NASAのIV&V：http://www.nasa.gov/centers/ivv/about/index.html

[TAKURA2010] 田倉 聡史：事例紹介 上流工程での品質確保のための発注者責任，SEC journal, No.21, Vol.6，No.2, pp.94-99, 2010

米国主要機関の取り組み状況から見た統合システムのディペンダビリティ確保に向けたエンジニアリング的課題について

SEC副所長 立石 譲二
SEC調査役 新谷 勝利

2010年6月27日（日）から7月3日（土）にかけて、初夏の風吹く米国を訪問した。主要機関のSEI※1、MIT※2、NIST※3を順に回り、多くの知見を得ることが出来た。SEC journal No.22では、SEI、MITとの熱い議論をお届けしたい。

1 出張の目的

今回の出張の目的には、次の2点があった。

① 本出張では、米国において、情報システムの信頼性向上対策の検討・実証を推進している代表的な組織・研究者を訪問し、今後のSECにおける“検証可能な”ディペンダビリティの推進に寄与するアプローチ、考え方、実践について、ヒアリング及び議論を実施する。

② 今後のフォローアップとして、それぞれの組織・研究者と以下を計画しているが、それを確認する。

・SEIを訪問：AADL※4について、システムの抽象的な構造を、アーキテクチャとして記述可能な言語の習得をする。

・MIT、Nancy Leveson教授を訪問：2011年1月中旬に予定してSECとJAXA※5の共催ワークショップにて同氏による基調講演をお願いし、安全性に関する意識向上を図る。

・NIST/ITL※6/SSD※7を訪問：2010年12月初旬に予定されているIPAとNISTの定期協議に、従来の情報セキュリティ部門に加えてソフトウェア・エンジニアリング部門を追加すべく準備協議を行う。

2 SEI

SEIとの会合における特記事項を下記にまとめる。

① IPA及びSEIの双方にて概要について説明した。

② 結果として、両者には、現行のプロセス改善及び定期的な電話会議に加えて、協業出来る分野があることを確認した。

③ CMMI※8関連の最近の動き、AADLについては、以降に説明する。

（1） CMMI関連の最近の動き（CMMIとTSP※9）

SEIではSEP※10という所長直轄の組織において、CMMI、TSP及びメトリックスを担当している。

CMMI、TSPについては担当マネージャであるMike Konrad博士とシニア技術スタッフであるSteve Masters氏による説明を受けた。

SEIは、CMMIを“What to do”、TSPを“How to do”をカバーするものである、と位置付けており、とくにTSPに力を入れているようである。

CMMI と TSP 導入による「実際に出荷された製品品質」[CMU/SEI2003] と題されたスライドによると、表 1 に示す数値が報告されている。TSP がカバーしている CMMI の固有プラクティスを調査した報告 [CMU/SEI2004] によると、TSP は CMMI の Level 5 までの固有プラクティスの 75% をカバーしており、TSP がとくにパフォーマンスの向上を意識した方法であるために改善結果の違いを生じさせているものと推察される。

なお、マイクロソフト社や Adobe 社においては、CMMI の実施はしていないが、TSP は実施しているとのこと。マイクロソフト社においては、ソフトウェア開発では関連する組織がライフサイクル全体をカバーするというアプローチではなく、開発現場における個別のチームオペレーションに規律を導入することにより、高い生産性、品質の向上を図ってゆくというアプローチのようである。

SEI としては、CMMI と TSP を一緒に導入することにより、パフォーマンスはより早く向上出来ると考えているようである。

現行 CMMI は 1.2 であるが、年内には 1.3 がリリースされる予定とのことである。

表1　KLOC※11当たりの欠陥数

ソフトウェアプロセス		KLOC 当たりの欠陥数
CMMI	Level 1	7.5
	Level 2	6.24
	Level 3	4.73
	Level 4	2.28
	Level 5	1.05
TSP		0.06

SEI（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）の前にて（新谷）

また、CMMI がスマートグリッド関連開発プロジェクトの Smart Grid の成熟度として適用出来るかどうか検討されているようである。

（2） AADL について（アーキテクチャ、AADL）

SEI では RTSS※12 という所長直轄の組織において、アーキテクチャ及び Software Product Line を担当しており、アーキテクチャ全般についてはシニア技術スタッフの Felix Backmann 氏、AADL については同じくシニア技術スタッフの Peter Feiler 氏から説明を受けた。SEI の方針として、SEI 所長傘下のグループが協働するというものがあり、TSP を RTSS の活動の中に具体的展開の方策としているとの説明があった。SEC の今後の活動内容から、SEI との今後の協働は、これまでの SEP との連携に加え、RTSS を加えることになるであ

脚注

※1　**SEI**：Software Engineering Institute，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
※2　**MIT**：Massachusetts Institute of Technology，マサチューセッツ工科大学
※3　**NIST**：National Institute of Standards and Technology，国立標準技術研究所
※4　**AADL**：Architecture Analysis & Design Language，米 SAE が策定したアーキテクチャ記述言語
※5　**JAXA**：Japan Aerospace Exploration Agency，独立行政法人 宇宙航空研究開発機構
※6　**ITL**：Information Technology Laboratory，情報技術研究所
※7　**SSD**：Software and Systems Division，ソフトウェア・システム部
※8　**CMMI**：Capability Maturity Model Integration，能力成熟度モデル統合，CMMI は米国での登録商標
※9　**TSP**：Team Software Process，TSP は米国でのサービスマーク
※10　**SEP**：Software Engineering Process
※11　**KLOC**：Kilo Lines of Code
※12　**RTSS**：Research, Technology, and System Solutions

ろう。

Felix Backmann 氏による、ACE※13 at the SEI と題する説明においては、以下が強調された。

・ソフトウェアに依存するシステムの品質とそのライフはアーキテクチャによって決まる。
・最近の OSD※14（DOD※15 の機関）、NASA※16 等の調査や SEI の HP にある [CMU/SEI2009] によると、アーキテクチャがソフトウェア問題の組織的原因と特定されている。
・ACE とは、システムが実現するミッションの達成において、アーキテクチャを活用することを中心課題とする方法論。
・ACE の活動とは、以下を言う。
  – システムのビジネスケースを作る
  – 要求事項を理解する
  – 適切なアーキテクチャを決める
  – アーキテクチャを文書化し、周知する
  – アーキテクチャを評価する
  – アーキテクチャに対し、適切な試験及び測定をする
  – アーキテクチャに基づきシステムを実現する
  – システムがアーキテクチャに適合し実現されていることを確認する
  – アーキテクチャがビジネスとゴールに対応するように維持する
  – 組織、チーム及び個人がアーキテクチャを効果的に活用出来るようにする
・AADL について、Peter Feiler 氏から以下の説明を受けた
  – ACE におけるコア部分として、AADL が位置付けられる
  – AADL は、以下を記述する文字及び図を用いる言語であり、詳しくは AADL の HP を参照のこと
  – ソフトウェア、ハードウェア、物理システムを定義する
  – 連続する制御、イベント、応答処理
  – 運用モデル
  – 大規模なシステムのモデリング
  – 様々なシステム分析要求に応える
  – AADL 授 [AADL HP] は様々なツールとインターフェースを持っており、詳しくは上記 AADL の HP を参照のこと

3 MIT

以下の諸項目は 4 時間に及ぶ Nancy Leveson 教授 [LEVESON] とのインタビューにおいて、彼女が強調したものである。当インタビューにおいては、① IPA の概要、② SEC の概要、③ SEC 及びその作業部会における活動トピックス、とくに③に多くの時間を割いてその活動にかかわるトピックスについて有意義な議論が出来た。以下に幾つかのキーワードとして、今後 SEC においてフォローすべきと考えられるものを列記する。

・Crisis of measurement

測れなければ管理出来ないが、人の能力及びプロジェクト管理というのは十分に測定出来るものではない。

・心理学

MCC※17 にて Belady 博士の下にソフトウェア開発の生産性及び品質向上の研究をしていた Bill Curtis[CURTIS-1] が行動及び認知心理学の観点からの考察が必要ということを IEEE[CURTIS-2] で発表している。安全性においても、「人の使用上の誤解に基づく」ことに起因するものがあり、心理学というものを考える必要がある。ちなみに Leveson 博士は心理学も専攻したとのこと。

・ソフトウェア工学からシステム工学へ

彼女の MIT での講義録は MIT オープンカリキュラム [MIT] の宇宙工学＃ 355 から最新のもではないがダウンロード可能であり、既にそこでシステム工学につ

いて説明。彼女はシステム工学を推進しているINCOSE[INCOSE]の理事でもある。

・より早いソフトウェア開発段階へのシフト

既にオックスフォード大学のTom Green[GREEN]講師により、「変化への対応」をいかに図るか、コミュニケーションのあり方、図視化が議論されており、彼女の活動はそれに影響を受けているとのこと。実行可能な要求仕様記述[要求仕様記述]というものがツールとして必要である。

・形式手法

Leveson教授はかつて形式手法の学会のプログラム委員等を務められたことがあるが、実際のシステムにおいてはステークホルダ間で「理解される記述」[MBSE]が必要であるとの彼女の主張は受け入れられず、彼女自身は形式手法のグループから一線を引いているとのことである。この用語は、「1つの言語というものではなく、ステークホルダのそれぞれが用いる言語の間でのマッピングがスムースに行われ、お互いがより良い理解をする手法」という説明をすると、より広く受け入れてもらえるのではないかとの指摘があり、これは、まさしくSEC形式手法人材育成WGが目指しているものであり、意を強くした。ちなみに、彼女もしばしば引用されているDavid Parnas教授も、SEC形式手法WGが主催するセミナーで講演していただいたときに同様な指摘をしている。Bメソッド※18、VDM※19、Z言語※20、等の形式手法言語を活用出来るようになるということと、形式手法が目的としているものを実現することとは同じではない。

次号のSEC journal No.23では、NISTの訪問を紹介する。

脚注

※13 **ACE**：Architecture-Centric Engineering
※14 **OSD**：Office of the Secretary of Defense
※15 **DOD**：Department of Defense，米国国防総省
※16 **NASA**：National Aeronautics and Space Administration，米国国立航空宇宙局
※17 **MCC**：Microelectronics and Computer Technology Corporation
※18 **Bメソッド**：B method，Jean-Raymond Abiralを中心に開発された形式手法の1つ。Z言語とも関連がある。
※19 **VDM**：Vienna Development Method，IBMのウィーン研究所で開発された形式手法の1つ。
※20 **Z言語**：Z notation，Jean-Raymond Abiralを中心にオックスフォード大学のプログラミング研究グループによって開発された形式手法の1つ。Z記法、Zメソッドとも言う。

参考文献

[AADL HP] http://www.aadl.info と https://wiki.sei.cmu.edu/aadl
[CMU/SEI2003] CMU/SEI-2003-TR-014：http://www.sei.cmu.edu/library/abstracts/reports/03tr014.cfm
[CMU/SEI2004] CMU/SEI-2004-TR-014：http://www.sei.cmu.edu/library/abstracts/reports/04tr014.cfm
[CMU/SEI2009] CMU/SEI-2009-SR-007：http://www.sei.cmu.edu/library/abstracts/reports/09sr007.cfm
[CURTIS-1] Bill Curtisを紹介するホームページ：http://www.linkedin.com/pub/bill-curtis/5/998/bab
[CURTIS-2] Curtis, B. (1981, Ed.)：Human Factors in Software Development, Washington, DC：IEEE Computer Society, 2nd Ed., 1985
[GREEN] http://www.brookes.ac.uk/schools/artsandhumanities/publishing/staff/green.html!
[INCOSE] http://www.incose.org/
[LEVESON] Nancy Levesonの教授のMITホームページ：http://sunnyday.mit.edu/, より詳細な履歴：http://sunnyday.mit.edu/bio-serious.html
[MBSE] http://sunnyday.mit.edu/csrl.html における Model-Based System Engineering
[MIT] http://search.mit.edu/search?site=ocw&client=mit&getfields=*&output=xml_no_dtd&proxystylesheet=http%3A%2F%2Focw.mit.edu%2Fsearch%2Fgoogle-ocw.xsl&proxyreload=1&as_dt=i&oe=utf-8&departmentName=web&filter=0&courseName=&q=Nancy+Leveson&btnG.x=6&btnG.y=11
[要求仕様記述]http://sunnyday.mit.edu/csrl.html

ETIひろしま Embedded software Technology Initiative

（ちゅうごく地域ロボットテクノロジー（高度組込みソフトウェア）産業活性化人材養成等事業）
（平成20年度～22年度　経済産業省:地域企業立地促進等事業費補助金　採択）

http://www.eti-h.jp/

ちゅうごく地域ロボットテクノロジー（高度組込みソフトウェア）産業活性化人材養成等事業　事務局　プロジェクトリーダー
岡本　勝幸

中国地域の産業は、鉄鋼、造船、輸送機械製造に代表される産業を中心とした瀬戸内工業地帯の歴史的背景から、経済の中心を製造産業に置く産業構造となっており、域内総生産（GRP）で約7.5兆円の製造能力を持つ。この地域における製造業ニーズは、GRPで約6.3兆円と推計されており、約1.2兆円の余剰生産を誇っている。

しかし、同地域におけるサービス産業は、物流関係･医療関係分野を中心としており、約6.8兆円の規模で、製造業を0.7兆円余り下回っている。この地域におけるサービス業ニーズは、GRPで約8.0兆円と推計されており、約1.2兆円の域内生産が不足しているのが実態である。これらのサービスとしては、情報サービス、調査サービス、金融等のビジネス支援サービスがある。

この地域の経済を総合的に見ると、製造業における約1.2兆円の域内余剰生産分を、域内で不足しているサービスの域外からの獲得に投入し、均衡させている実態が明確になる。このことから、地域内産業構造の製造業への依存度の高さと、サービス業の域内供給が不足しており、サービス産業が未成熟であると言える。

先進国における経済は、急速にサービス化しており、GDP及び就業人口の両方で、全体の70%以上を占めていることが報告されている[大場2009]。我が国においても、経済産業省の調査によれば、GDPの約70%をサービス産業が占めている[通商白書2010]。そのような視点からこの地域の経済を見ると、製造業への依存度が極めて高く、長期的な産業構造に大きなリスクを内包している。

また、中国地域の製造業も、大きな構造変革のさなかにある。近年の著しい情報技術の発展によって、製造業において生産される製品に情報技術を応用した部品を組み込み、システムとしての製品の製造コストを大幅に削減することが可能になる。また、情報技術をしなければ不可能な知的な制御も可能となり、製品の機能と価値を高める例もある。

ここでは、以上を踏まえ、SECの方々にもご協力をいただいている「中国地域における高度組込みソフトウェアに関わる人材育成事業」について紹介する。

1　事業の目的

中国地域内の製造業が高度化し、国際競争力を維持するためには、世界的に著しい速さで進行している「部品のエレクトロニクス化※1」に対応出来る、以下に挙げる企業を支える人材の育成が急務である。

① 企業内で製品・部品開発を担当する技術者（ハードウェア分野・ソフトウェア分野）
② 製品開発にかかわる技術者を管理・統率する第一線の管理者
③ 上記人材の育成を担当し、企業における製品・部品の開発プロジェクトを成功させるために必要な知識を活用し企業経営を支援するコンサルティング等を担当する高度な専門知識を持つ人材（とくにソフトウェア分野）

本事業においては、企業の内外を問わず、上記②及び③に該当する高度な人材を育成することを目的とし、以下の項目を実践する。

① ソフトウェア分野において、特定製品・部品開発を支援する専門教育及びコンサルティング業務に携わる、高度に専門的な知識と、実践経験を備えた高度かつ高付加価値を持った企業内外の人材を育成することを主たる目的とした教育プログラムを開発、実証する
② 実証に基づいてより効果的な教育プログラムに改善してゆく

本事業の教育プログラムを通じて、以下の内容を具現化することを目標とする。

① 学習者の高度な専門知識の獲得と演習等による体験学習を通した実践的なノウハウの獲得

② 高度かつ高付加価値な組込みソフトウェア開発に関するコンサルティングを提供出来る人材の育成
③ 上記の人材を核として、広島県内を拠点とする地域内外製造産業の高度化を支援する環境を、長期的な視点から整備する

2 研修の目的・構成及び単位・修了認定について

(1) 研修の目的

ソフトウェア開発のリスクとプロジェクト管理の理論を学び、ソフトウェア管理者として最小限知っておくべき基礎を理解する。

・ソフトウェア工学の重要な分野であるソフトウェア・プロセスに関する基礎を学び、その現実のプロジェクトの課題を理解する
・ソフトウェア・マネジメントや品質マネジメント等の基礎であるソフトウェア・プロセスの改善、プロセスの成熟度評価等に関する理論を学び、現場での実践における要点を理解する

(2) 研修の学習目標

① ソフトウェアのリスクと管理に関連した規格や話題に関する基礎知識を習得する
・ISO 9126、ISO 9000、ISO 15504、ISO 12207 等の標準規格の概容を理解する
・IEEE 830 や CMM※2、CMMI※3 等の実績のある企画や手法等の概容を理解する
② 他のソフトウェア技術やプロセスに関する講座において基礎となる知識を獲得する
・ソフトウェアの技術的レビューやテストの目的と方法に関する基礎的知識を習得する
・ソフトウェア・プロジェクトの管理の基礎となる理論や方法論に関する基礎的知識を習得する

(3) 育成すべき人材像

本研修は、近い将来において地域内の製造業における製品のエレクトロニクス化を根底で支える組込みソフトウェアの開発を適確にマネジメント出来る人材、組込みソフトウェアの開発に携わる人材に対して教育・指導を担当出来る人材、またそのような適切なマネジメントを社外から支援出来るコンサルタントに、近い将来成長出来る人材を育成する。

(4) カリキュラムの構成

本研修のカリキュラムの構成を図1に示す。本研修のカリキュラムは、2つの基礎科目（必修）の上に構成される以下の4つの専門分野から成り、それぞれ、1つ以上の専門科目の単位修得を前提とする。

脚注

※1 **部品のエレクトロニクス化**：様々に存在する部品の内、従来より機械的及び一部電気的な仕組みにより動作していた機能を、ソフトウェアとの組み合わせにより更に高度化した機能として動作実現させること。半導体素子と併せて導入されることが多い。
※2 **CMM**：Capability Maturity Model
※3 **CMMI**：Capability Maturity Model Integration

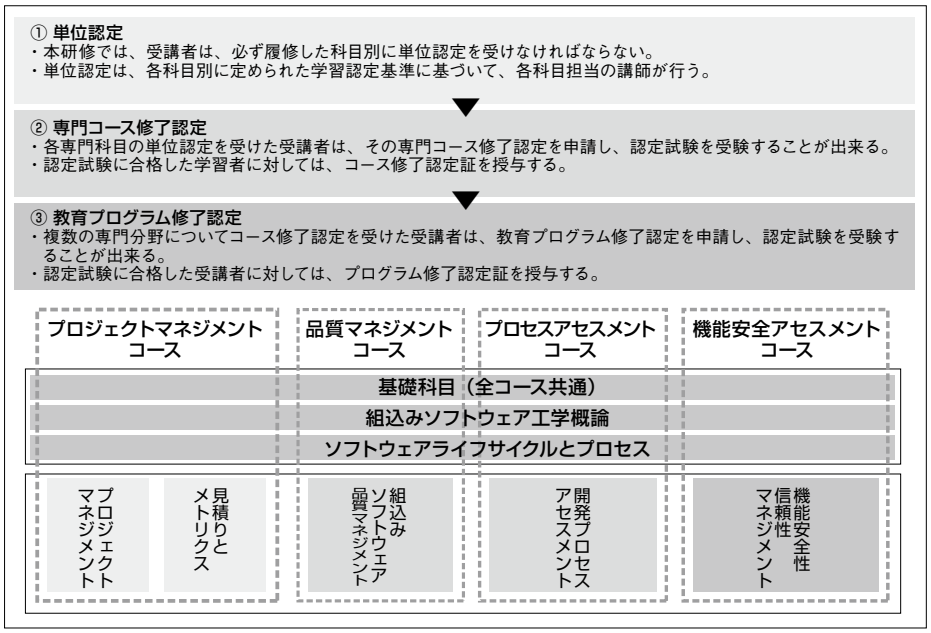


図1 カリキュラムの構成

3 事業実施計画

図 2 に平成 20 年度からの事業実施計画を示す。

4 研修項目及び概要

表 1 に研修項目及び概要を示す。

5 研修担当講師

研修を担当する講師とその担当内容は以下の通り。

1. 大場　充（広島市立大学情報科学部情報数理学科 ソフトウェア工学講座　教授）
   担当：「組込み系ソフトウェア開発技術者に求められるスキル・ソフトウェア開発技術の基礎」
2. 新谷　勝利（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　調査役）
   担当：「ソフトウェア・プロセスと共通フレームの基礎および演習（ISO/IEC 12207）」
3. 堀田　勝美（株式会社コンピータジャパン　取締役社長　チーフ・コンサルタント　CMM/CMMI/ISO 15504 リードアセッサ）
   担当：「開発プロセス評価ガイドの基礎および演習（ISO/IEC 15504）」
4. 神庭　弘年（日本アイ･ビー･エム株式会社　シニア・エグゼクティブ・プロジェクトマネジャ　PMI 日本支部　会長）
   担当：「プロジェクトマネジメントの基礎および演習」
5. 松本　健一（奈良先端科学技術大学院大学　情報科学研究科ソフトウェア工学講座　教授）
   担当：「ソフトウェア工数見積の基礎と実習」
6. 奈良　隆正（NARA コンサルティング　代表）
   担当：「品質マネジメントシステムとソフトウェア品質保証の基礎および演習」

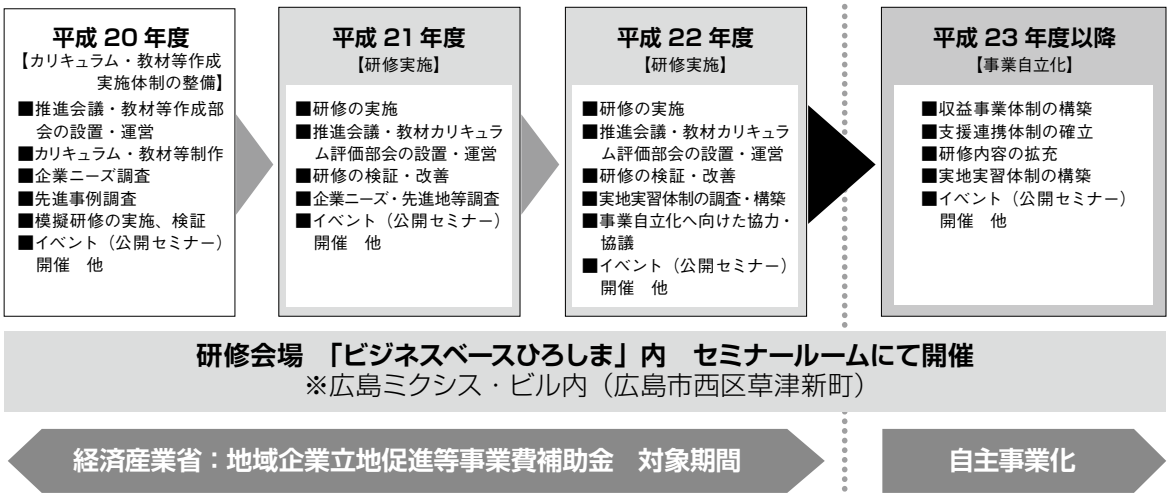


図2　実施計画

表1　研修項目と概要

	項　目	概　要	講義回数（1.5h/ 回）
1	組込み系ソフトウェア開発技術者に求められるスキル・ソフトウェア開発技術の基礎	ソフトウェア開発技術における基礎的な解説及び組込み系ソフトウェア開発技術者が習得すべき技能や能力等に関する解説	20 回（30h）
2	ソフトウェアライフサイクルとプロセスの管理（ISO/IEC 12207）	ISO/IEC 12207 におけるソフトウェアの開発工程に共通するフレームワーク等の基礎的な学習及び演習による理解度の確認	10 回（15h）
3	開発プロセス評価ガイドの基礎および演習（ISO/IEC 15504）	ISO/IEC 15504 におけるソフトウェア開発プロセス評価、改善手法等に関する基礎的な学習及び演習による理解度の確認	20 回（30h）
4	プロジェクトマネジメントの基礎および演習	プロジェクトマネジメントの基本的な構成要素となる、プロジェクト管理や PMP、PMBOK、P2M の基礎を学び、演習により理解を深める	15 回（22.5h）
5	ソフトウェア工数見積の基礎と実習	ソフトウェア開発における工数の算出及び見積に関する基礎的な学習及び演習による理解度の確認	15 回（22.5h）
6	品質マネジメントシステムとソフトウェア品質保証の基礎および演習	ソフトウェアにおける品質保証及びマネジメントに関する体系的な基礎学習と演習による理解度の確認	15 回（22.5h）
7	機能安全と安全性設計技術の基礎および演習（IEC 61508）	IEC 61508 におけるソフトウェア設計及び開発に関わる安全性に関する基礎的な学習及び演習による理解度の確認	15 回（22.5h）

7. 土肥　正（広島大学大学院工学研究科情報工学専攻 ソフトウェア信頼性工学講座　教授）
担当：「機能安全と安全性設計技術の基礎および演習（IEC 61508）」

6 研修の実績について

① 平成 21 年度：（修了）
・研修実施期間：平成 21 年 7 月 28 日（火）～平成 22 年 2 月 14 日（日）
・受講者数　　：25 名　　・研修修了者数：23 名
・単位認定者数：18 名
・専門コース修了認定者数：延べ 11 名
（プロジェクトマネジメントコース：3 名、品質マネジメントコース：3 名、プロセスアセスメントコース：3 名、機能安全アセスメントコース：2 名）
② 受講生及び派遣元企業の声（アンケート調査より一部抜粋）
・仕事上の現実的な課題に対して具体的な課題の整理が出来、かつ対応策の立案・実施と進むことが出来た
・研修の受講を通じて仕事を遂行する上で定量的な状況判断が出来るようになり課題整理や解決に役立った
・ソフトウェア工学に関して基本の考え方と我々の実開発で適用する上での課題を探ることが出来るレベルまで習得することが出来た
・ソフトウェア品質の計測において新たな手法の導入を検討開始出来た
・ソフトウェアに関する知識を体系的に仕立ててシステム開発に必要となる技術として習得・整理することが出来た
・実務経験が豊富な講師の具体例を交えた話や経験談、事例紹介やエピソードが面白く講義内容にも興味が持て、学術的な内容や裏づけも分かりやすかった
・ディスカッションや議論を交えた講義が大変良かった。とくに他社の受講生の考え方や意見等を直に聞けて様々な意見交換も出来、新鮮であったと同時に勉強になった
・参加した社員からの評価も高く仕事上で役立つ内容も多いと考える。来年度も積極的に参加して受講させたい
③ 平成 22 年度：（実施中）
・研修実施期間：平成 22 年 6 月 14 日（火）～ 11 月 27 日（土）
・受講者数：研修受講→ 25 名
広域連携受講者（DVD 配布による受講、平成 22 年 10 月実施予定）→ 26 名（広島、山口、岡山、島根各県より参加）

7 今後の展望：地域戦略及び目標

地域としての人材育成・技術開発戦略を形成することが重要かつ可及的課題である。
・産学官が合意・連携した高度人材育成戦略・取組みが必須
・高等教育・実務経験を通した人材育成が不可欠
・高度人材が支える「高度ものづくり支援技術開発」が必要
・グローバルを意識した人材育成、人材確保が必要

・平成 22 年度より広島市立大学大学院情報科学研究科システム工学専攻において開設された「組込みソフトウェア関連専門コース」との単位相互認定等を柱とした協定を締結する予定

2025 年に世界レベルの新産業地域を実現することを目標とする。
・2015 年までにより「具体的な」高度人材の育成を目指す
→「専門コース修了認定者」：50 名
「教育プログラム修了認定者」：10 名

■問い合わせ先

・株式会社広島ソフトウェアセンター
担当：浜田隆文
電話：082-278-8877　Fax：082-278-8878
E-mail：hsc@h-sc.co.jp

・ビジネスベースひろしま
担当：岡本勝幸
電話：082-501-5004　Fax：082-501-5014
E-mail：info@b-base.jp

参考文献

[大場 2009] 大場 充：組込みソフトウェアの高度化とものづくり 組込みソフトウェア産業の育成と地域戦略，中国総研，No.47，社団法人中国地方総合研究センター，2009 年 7 月
[通商白書 2010] 経済産業省：通商白書 2010

ソフト業界の構造変化への対応を急ごう

IPA 顧問　学校法人・専門学校 HAL 東京 校長
鶴保 征城（つるほ　せいしろ）

ソフトウェア開発の上流工程に関する勉強会を、実践的ソフトウェア教育コンソーシアム（P-sec）の主催で行いました。IPA、JISA（社団法人 情報サービス産業協会）等幾つかの団体のご支援のおかげで、35 度を超える猛暑（8 月 23 日）にもかかわらず 200 人以上の方々に参加していただき熱心な討論が行われました。以下は、関係各位にご案内した趣意書の要約です。

『ソフト業界への逆風はおさまる様子もなく、むしろ現状が常態と考えざるを得ない状況になっています。周知のように、大手 SIer の戦略転換は明らかであり、多重下請け構造での仕事＝工程分業が可能な仕事については、中国やインドへのオフショアを急速に進めています。国内に残った仕事も利益が出なくなりました。

一方、ユーザ企業では、単なる業務効率化のためのシステム化は一巡しました。現在の課題は、業務の高度化や企業変革、あるいは新たな利益を生むビジネスモデルの創出であり、そこに企業の生き残りをかけています。このためにはそれを支えるシステムが不可欠であり、ユーザと一体となってソフト開発を完遂してくれるベンダーを求めています。つまり、ユーザにとっては、今やビジネスモデル創出とソフト開発は、表裏一体なのです。

「日本のソフト会社は今後 5 年で半減する」「仕事は 30％減少する」「ソフト業界の従業員数は 87 万人、そのうちソフト技術者は 60 万人程度。そして SE と言える人は、多く見て 1 割ぐらいではないか」などと言われています。つまり、技術者の多くを占めるプログラマーや管理要員については、アウトソーシングが進むと需要が激減する可能性があります。

業界全体はこのようなトレンドにあるとしても、個々の企業は勝ち抜き、生き残らなければなりません。一つの方法として、ますます高度化するユーザの期待に直接応える方法があります。昔から言われている下請けから元請けへの転換ですが、ユーザからの直接受注は、下手すると大赤字の原因になります。小生も含めて、手痛い経験をされた（している）方が多いと思います。この結果、ソフト会社は、「羹（あつもの）に懲（こ）りて膾（なます）を吹く」状態で、リスクの少ない下請けや派遣業務に安住していますが、時代は変わり、これではソフト会社の将来はありません。

この状態から脱するために、ベンダーがマスターすべき最も重要な技術に、上流工程の要求分析があります。「そんなことはわかっている。すでにやっているよ」とおっしゃる方も多いと思いますが、なかなか思うような結果が出ていないのではないでしょうか。

上流工程のポイントは次の三つです。

① 概念モデルの明確化。多くのユーザ企業は現在、ビジネスモデルの抜本的な変革を迫られています。ベンダーとしては、ユーザの事業の本質・方向性を指し示す概念モデルの作成に協力しなけなければなりません。

② 概念モデルが出来ても、仕様を自然語で記述しているのではダメです。あいまいさを排除するために、モデリング技術を使って概念モデルをヌケモレなく実装可能な記述に落とし込まなければなりません。

③ 全社レベルのデータ整備を行うこと。データ整備とは各種マスターやデータ項目の一元化を指しますが、上記①、②を実施するための大前提です。』

多くの方々から、随分過激だなと言われましたが、趣旨については大方の賛同が得られたと思います。パネル討論は、司会 田口潤氏（株式会社インプレスビジネスメディア 取締役）、パネラー 佃均氏（一般社団法人 IT 記者会 代表幹事）、神沼靖子氏（P-sec 副会長）及び講演者にお願いし、活発な議論が行われました。パネル討論の詳細は、講演資料、アンケート結果と共に、P-sec の HP（http://www.p-sec.jp/）に掲載しておりますので、ぜひご覧ください。

ソフト会社の救いは、長年の蓄積によって財務状況が良い会社が多いことです。単年度の収益は気になるところですが、本稿のような上流工程シフトへの人材育成や企業連携・買収等に向かって、先達が蓄積した資産を有効に活用すべき時期に来ていると思います。

「伝える力」を育成する努力をしなければならない

「話す」「書く」「聞く」能力が仕事を変える！
伝える力

池上 彰　著
ISBN：978-4-569-69081-0
PHP 研究所刊
新書判・205 頁
定価 840 円（税込）
2007 年 5 月刊

この本は、第 64 刷（2010 年 8 月 20 日）で 90 万部以上売れたということである。SEC journal No.21（2010 年 6 月 30 日発行）の当欄で山本修一郎氏の著作「CMC で変わる組織コミュニケーション」の書評を書いた。同じコミュニケーションにかかわる書評であるが、SEC journal 読者の日常コミュニケーションでは、論文や技術書に基づくものだけではなく、むしろビジネス要件をシステム及びソフトウェアで、いかに実現するかにかかわるものであろうと考え、あえて今回は、柔らかいハウツー本で 3 年前に発行された本の紹介をしたい。

第 8 章の「上質のインプットをする」には、計 70 のいわゆるベストプラクティスが説明されている。

社団法人 日本情報システム・ユーザー協会（JUAS）の調査においてもソフトウェア開発におけるエンジニアの文章力が問題であるということも判明している。ソフトウェアそのものは確かに形式化の進んだプログラム言語で表現されるが、その入力となるビジネスの記述は日常使用する日本語であり、その記述が「伝える力」を有していないと、結果として作成されるソフトウェアはビジネスの要件を反映していないものになる可能性が大となる。私たちは、日常の生活において、「伝える力」を育成する努力をしなければならないと考え、あえて、本書の紹介をする次第である。

（新谷勝利）

人材育成担当者こそ技術者以上に学習すべき

人材開発マネジメントブック
学習が企業を強くする

福澤 英弘　著
ISBN：978-4-532-49046-1
日本経済新聞出版社刊
A5 判・312 頁
定価 2,730 円（税込）
2009 年 2 月刊

一般的な教育学、学習理論、更には経験に基づく独自の教育方法に関しては多く書籍が存在する。しかし、技術者が人材育成を考える立場になったときに、勉強の方法が無い状況である。本書は世の中に数多く存在する人材育成に関する知識を体系的に整理・解説している。学習理論から始まり、研修設計だけでなく OJT や現場での学びまで多岐にわたっているため、概要を認識し、人材育成に取り組む際の参考になる教科書である。

企業の技術教育担当は、元技術者が担当することが多く見受けられる。技術に関しては経験やスキルを保有しているが、人材育成に関する知識やスキルは持ち合わせていない場合もあるだろう。では、その組織が長年やってきた人材育成の方法を継承し、取り組むだけで良いのであろうか。技術と同様、時代と共に人材育成の方法論や、育てる対象となる技術、そして人材のスキル、モチベーションも大きく変わってきている。人材育成のセオリーを理解し、その組織が置かれている状況に合わせ適用出来るスキルが求められる。

本書は、文献情報として各章ごとに参照している文献が掲載されている。更に各領域ごとに 3 冊のお勧め文献も掲載されている。教育学や学習理論等を深堀したいときには、この情報を基に探求することが可能である。

企業は人が命だと言うのであれば、知識と知恵、そしてスキルを有する人材育成担当者が求められる。本書はそんな人材育成担当者の入門書籍としては最適な一冊である。

（渡辺登）

編集後記

SEC journal No.22をお届けします。このNo.22から若干スリム化を図りました。いかがでしょうか。

お伝え出来る内容の密度を上げ、更なる充実に努めてまいりますので、よろしくお願い申し上げます。

今号では、SEC journal No.21の記事『特別セミナー「組込みソフトウェアの信頼性を考える」より』と同様、組込みソフトウェアについての信頼性を考えることを主題とし、その2回目として「組込みソフトウェアによる信頼性及び安全性」を取り上げました。これらはシリーズとして、SEC journal No.23に3回目を企画しており、ここで完結する予定です。また、アングルでは、形式手法を取り上げています。これも前号からのシリーズとして今後も連載を予定しています。ぜひ継続してご覧ください。

(Y.I)

編集部より

この「SEC journal」は、読者の皆様方からのご意見を頂戴したくアンケートをご用意いたしました。今後も同封させていただきますが、ご一読ご協力お願いいたします。

古都を染める紅葉

（撮影：金沢成恭）

SEC journal® 第6巻第3号（通巻24号） 2010年10月25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 松田 晃一
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

IPA（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。

応募様式は、下記のURLをご覧ください。
http://sec.ipa.go.jp/secjournal/papers.html

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

応募要項

投稿締切り

年4回、3ヵ月毎に締切り、締切り後に到着した論文は自動的に次号審査に繰り越されます。
（応募締切：1月・4月・7月・11月各月末日）
締切り後、査読結果は1ヶ月後に通知
詳細スケジュールについては、投稿者に別途ご連絡いたします。

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。

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2009年10月29日IPAフォーラム2009）。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）。

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

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.17

No.18

ESxR特集号

No.19

No.20

No.21

SEC journal No.22
第6巻第3号（通巻24号）
2010年10月25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 Tel.03-5978-7543 Fax.03-5978-7517
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 URL：http：//www.ipa.go.jp/
所長 松田 晃一

IPA®
独立行政法人 **情報処理推進機構**

ISSN 1349-8622