2014 年 3 月 31 日発行
第 10 巻 第 1 号（通巻 38 号）
ISSN 1349-8622

SEC journal 36

IPA 独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

SEC journal No.36
Vol.10 No.1 Mar. 2014

来たるべき IT 利活用社会の未来に「死角」はないか？

独立行政法人情報処理推進機構（IPA）
理事　技術本部長
立石　譲二

2013 年 7 月より独立行政法人情報処理推進機構（IPA）理事に就任しました立石譲二です。2012 年 5 月まで約 4 年間、SEC 副所長として情報システムの高信頼化に取り組んできましたが、約 1 年ぶりに当機構に戻り、セキュリティ、ソフトウェア高信頼化及び国際標準の推進に取り組んでまいりますので、よろしくお願いいたします。

試される日本発の IT 社会の安全・安心

昨年は 2020 年のオリンピック・パラリンピックの開催地が東京に決定した記念すべき年となりました。ちょうど半世紀前の東京オリンピック開催時、首都高、東名・名神などの高速道路や東海道新幹線といった社会インフラが整備され、世界の人々を出迎えたように、6 年後の大会開催に向けては、更に新たなインフラの整備が進展するものと期待されます。その際、スマートコミュニティや次世代 ITS のような高度な IT システムは従来のハードインフラに代わる新たな社会インフラの中核となることは間違いありません。この種の大規模システムは、自動車や家電製品などのコンシューマ・デバイスと電力網や通信網といった従来は独立していた様々なシステム同士が相互連携することにより一体として機能する、所謂 System of Systems（SoS）としての特徴を有しています。こうした高度なインフラを活用することにより、これまで困難だった交通渋滞の解消やリアルタイムな電力デマンド・レスポンスの実現が期待されている一方、この種の前人未踏の複雑系システムの安全・安心を確保するためには、セキュリティや高信頼化に向けた新たな課題にも挑戦していかなければなりません。特に、SoS においては、システムの構成要素は従来のように情報システムとそれを利用する人間という単純な構成ではなく、センサーと各機器が直接連携するマシン・トゥー・マシン（M2M）制御の構成比率が飛躍的に高まることから、個々の要素システムと全体システムとの間の安全性の関係を体系的かつ整合的に整理していく必要があります。日本の企業は個々の製品分野のものづくり力では世界をリードしてきましたが、こうした安全面でのシステマチックなアプローチは正直得意ではなく、ある意味で死角になっているといってもいいでしょう。例えば、スマートコミュニティにおけるシステムの安全評価に関する各メーカー間の責任分界点や階層的な安全管理モデルや標準規格の整備はまだこれからといった状況です。更に、様々な機器がインターネットにつながることから、サイバー攻撃のような外的リスクも増大します。ちなみに、2 年前のロンドン五輪開催時にはわずか 2 週間の開催期間中に 2 億 2 千万回以上のサイバー攻撃を受けたというデータもあります。こうした他国の教訓にも学びながら、6 年後のスポーツの祭典では、世界一安全で安心な IT インフラで世界のアスリートや観客を「おもてなし」したいと切に願う次第です。

技術が繰り出すリアリティに追いつけない利用者の意識

近年のスマホ利用者の急増とソーシャルメディアの発達に伴い、SNS への不適切な書き込みや悪ノリ画像の投稿など世間を騒がせる出来事が頻発しています。事件の当事者たちは、ごく限られた仲間うちで共有するつもりでアップしたのでしょうが、モバイル系を中心に高度に発達したインターネットにおいては、世界中の人々に公開したに等しいというのが今の現実です。別の例として、関東地方の大竜巻発生やロシアでの隕石落下などでは個人の投稿動画が多数提供され、興味深い映像とともに発生メカニズムや現象の解明に貴重なデータを提供してくれました。一昔前はトラック 1 台分の衛星中継車と専門スタッフを必要としていた実況中継装備を現代の私たちは一人一人が手のひらに収まるスマートデバイスとして毎日持ち歩いているのです。ネットの掲示板に書き込まれた友達の悪口を苦に自殺してしまった少女の心の中は、それを批判する私たち大人たちの感覚とはほど遠く、あたかも都心の往来のど真ん中で張り紙を貼られて立たされているのと同じような苦痛を味わっていたに違いありません。今や IT は仮想現実を通り越し、拡張現実（AR）など現実と仮想をシームレスに体感させるまでに至っています。IT 利活用社会の平和と幸福を保つためには、それを利用する私たちの意識やモラルの変革ももう一つの死角になってはいないでしょうか。

世界をリードするソフトウェア・エンジニア像とは？

国立情報学研究所（NII）副所長
東京大学大学院 教授
本位田　真一

×

SEC 所長
松本　隆明

IT が社会を支える昨今、ソフトウェアの安全性はますます重要な課題となってきた。IT の利用形態はますます多様化しており、ユーザー層も広がる一方である。パソコンだけでなく、スマートフォンやタブレットなどのモバイル端末、クラウドサービス、スマートシステムといった様々な製品やサービスが連携して、システムはより複雑化している。しかし、ソフトウェア業界は相変わらず中核となるエンジニアが不足しており、学生にも人気がない状態にある。そこで、次世代を担うソフトウェア・エンジニア像とその育成について、国立情報学研究所 副所長の本位田真一氏と考察した。

松本：ソフトウェアの重要性が高まってきた昨今、開発時の要求条件はより厳しくなってきました。優秀な開発者を揃えていくことは必要不可欠な課題ですが、エンジニアの数はなかなか増えません。そこで本日は、世界をリードするソフトウェア・エンジニア像とその育成について考えていきたいと思います。まずは先生が代表を務めていらっしゃる、研究者育成センターについて聞かせていただけますか。

本位田 真一（ほんいでん　しんいち）

1978 年早稲田大学大学院理工学研究科修士課程修了。（株）東芝を経て 2000 年より国立情報学研究所教授、2012 年より同研究所副所長を併任、現在に至る。
2001 年より東京大学大学院情報理工学系研究科教授を兼任、現在に至る。
現在、電気通信大学、英国 UCL などの客員教授を兼任。
2005 年度パリ第 6 大学招聘教授。
日本ソフトウェア科学会理事、情報処理学会理事、日本ソフトウェア科学会編集委員長、IEEE Computer Society Japan Chapter Chairman、ACM 日本支部会計幹事を歴任。
日本学術会議連携会員。
博士（工学）。

本位田：次世代の中核を担う技術者・研究者を育成する「GRACE センター（先端ソフトウェア工学・国際研究センター）」は、2008 年に設立された組織です。IT 技術者の教育拠点として、2004 年からご支援を受けていた科学技術振興調整費を活用して立ち上げました。ソフトウェアシステムの開発現場に最新の研究成果を導入できる、次世代リーダーになりうる人材「トップエスイー」（トップ SE）を育成しています。

当初は教育をメインに活動していましたが、教育だけでは不十分だと感じ、その後は研究・実践との連携を目指しました。例えば、研究成果を教育の現場に提供したり、教育で生まれた新たな課題を受講生と共に深堀りして共同研究に発展させたり、現場で実践した研究成果をまた教育の場にフィードバックしたりというように、教育・研究・実践を三位一体にするスパイラルを作ったのです。また、国内だけにとどまらず、世界各国の教育機関と連携しているのも特徴です。英国、ドイツ、フランスなど様々な国の大学と共に教育を進めています。

松本：教育を受けているのはどのような方々ですか。

本位田：大学院の学生も受講していますが、主な対象者は社会人です。当初は研究部門の方が多かったのですが、最近は開発現場の方のほうが増えています。修了後の進路は様々で、また自らの現場に戻って教育成果を活かそうとする社会人もいれば、学んだ課題を突破したいと社会人博士課程に進む修了生もいます。ただ、開発現場へ戻って学んだ成果を適用しようとすると、修了生は様々な課題に遭遇します。そこで、一年間の研修修了後も、それぞれの課題を持ち寄って共有したり解決策を議論できる場を提供することや、勉強会を開催するといったフォローもしています。

「トップエスイー」の受講生はこれまでで 230 名ほどが修了しており、現在は 45 名が受講中です。

松本：どのようなカリキュラムを組んでいるのですか？

本位田：設立当初は「アーキテクチャー」や「形式手法」に関する講義時間を多く行っていました。その後どのような教育が必要なのかを協賛企業にインタビューしたうえで「要求工学」や「クラウド」などを追加し、カリキュラムに反映してきました。また、来年度からは、新しく「プロジェクト・マネジメントコース」を設立します。これまでの「トップエスイー」は、プロジェクトの中心となる“スーパー・アーキテクト”を育成するプログラムでしたが、時代の流れを考えると、プロジェクト・マネージメントの視点が欠かせなくなっています。そこで、「勘と経験と度胸」のプロジェクト・マネージメントの世界に、いかにサイエンスの視点のマインドを持ち込むかという教育を推し進めようとしています。

松本：IPA でもプロジェクト・マネジメントやプロセス改善の方法論を開発し、普及・展開してきています。ところが、現場で活かされている様々な知見を、なかなかうまく体系化できません。生産性や品質のレベルアップを支えてきた暗黙知を継承していくためには、サイエンスな視点での整理・体系化が必要だと考えています。今は、人から人へ伝えていく徒弟制度の域を出ていないのが実情ではないでしょうか。

本位田：冒頭にあったエンジニアが不足気味という現状も踏まえると、やはり徒弟制度では限界がありますね。何らかの方法で形式化し、広めていかないといけません。

松本：先生も私も企業出身ですが、現状はやはりアーキテクチャーがわかる人間は限られています。全体のアーキテクチャーの設計に失敗すると、システムの開発は最後にどうしても破綻してしまう。機能だけでなく、非機能にかかわる部分をきちんと設計できるエンジニアがなかなかいないので、今後の育成が課題ですね。

本位田：これまでの多くのシステムは継続案件が多く、従来の開発経験が再利用できるものも少なくありませんでした。しかし、今後どんどん新しいサービスを展開することを考えれば、やはり新規開発課題が出てきます。そうした場面でも必要な技術を持ってきて開発プロセスを提示し、自分のプロジェクトに落とし込めるようにならなくてはなりません。

GRACE センターの新しい講義システム

松本：GRACE センターには、最先端のソフトウェア工学ツール開発や研究理論の構築ができる「トップ・リサーチャー」を育成するための「edubase」という教育システムがありますね。

本位田：はい。教室で充分理解できなかったり、欠席してしまった講義の録画を、パソコンやスマートフォンで復習できるシステムです。Java や C プログラミング入門といった初級の話は一切なく、非常に高度な内容ですが、訪問者は月 3 万人、ひと月で 26 万 PV とかなりのユーザーがついています。ソフトウェア工学という限られたフィールドでは、非常に誇れる数字だと思っています。

松本：IPA でも様々なセミナーを録画し、インターネットで閲覧できるようにしていますが、とてもそれほどの PV はありません。地方の皆さんには大変有効という意見ですが、いかがでしょうか。

本位田：インターネット配信で 19 科目を遠隔受講でき、4 名が実際に受講中です。例えば、ツールを使った演習では、ツールの使い方で戸惑う人も少なくありません。そんなとき、リモートでもその場で手をあげるような感覚で気軽に質問できる環境が整っています。今はまだテレビ会議の性能が追い付いておらず、ディスカッション中心の講義を遠隔で受講するのは難しいですが、ゆくゆくはその点も解消したいと思っています。

松本：インタラクティブなシステムで、非常に有意義ですね。今後は、どのような方向性で進めていく予定ですか？

本位田：これからどのよう

松本 隆明（まつもと　たかあき）

1978 年東京工業大学大学院修士課程修了。同年日本電信電話公社（現 NTT）に入社、オペレーティング・システムの研究開発、大規模公共システムへの導入 SE、キャリア共通調達仕様の開発・標準化、情報セキュリティ技術の研究開発に従事。2002 年に株式会社 NTT データに移り、2003 年より技術開発本部本部長。2007 年 NTT データ先端技術株式会社常務取締役。2012 年 7 月より独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）所長。博士（工学）。

な人材が必要になるかで、教育の方向性は変わっていきます。今は、発注のあったソフトウェアをただ作ればよいという時代ではありません。設計の段階でも、納品後の運用に関しても留意する必要があります。また、新しいサービスを展開しようとすれば、社会や法律との関わりも大切になってきます。特に、著作権などは、大きな問題となってきます。例えば米国で、テレビ局が配信した番組をある事業者がサーバーに置き、ユーザーがそのサーバーから閲覧を楽しんだという判例があります。こうしたサービスの提供は日本の感覚では完全に法律違反ですが、米国では個人使用なので問題ないという判決でした。このように、法律には各国で差があります。ですから、今後のエンジニアは法的な諸問題に対応するところまで、トータルに検討しなくてはなりません。諸外国において新たなサービスを提供する場合、そうしたスキルを身に着けながら国内外の実態をしっかりと把握し、実践できるようにしなければ、大きなビジネスチャンスを逃す可能性もあるでしょう。

GRACE センターの人材育成プロジェクト

松本：昨年 6 月に政府から発表された「世界最先端 IT 国家創造宣言」には、「IT 人材強靭化計画」として“国際的にも通用・リードする実践的な高度な IT 人材の育成”が謳われています。しかし、なかなか日本では国際的に突出した人材が出てきません。GRACE センターでは、通常の教育プログラムのほか、グローバル人材という意味ではどのような取り組みを行っていますか。

本位田：IT 人材にとってもステークホルダーがどんどん変化するなか、国際的なプロジェクトで活躍できる人材を育成するため、「UCL（ロンドン大学）」と連携して、相互に受講生を派遣し合う共同プロジェクトを実施しています。前回は、UCL から修士の学生が約 40 名、こちらからは 8 名ほどの社会人が参加しました。日本人 1 名につき、年下の外国人が 5 名という構図です。そこで、考え方やカルチャーが全く違うメンバーで 5 日間の集中開発演習を体験します。実は、言語の壁よりもカルチャーの違いが大きく、「日本人ならこれでわかってくれるのに……」といった悩みが尽きません。付き合う人間も多様化してくる時代で、よい経験となっているようです。そのほか、ビッグデータの専門家を招いたイブニングセミナーを開いて、データサイエンティストからソフトウェア技術者に向けての要望を汲み上げるなど、様々なプログラムを用意しています。また、要求工学の講義では、組込み系とエンタープライズ系のコミュニティの違いを肌で感じ、全く異なる要求定義の仕方をお互いに勉強できるのです。

さらにこれからは、継続的デリバリーとも呼ばれる大規模なシステムを 24 時間 365 日運用しながらソフトウェアシステムを適宜更新していくスキルが要求されます。その課題をクリアするために、従来のプロジェクトではプロジェクト・マネージャー、アーキテクト、設計者、プログラマー、運用者などと役割が分担されていましたが、設計者は運用を、運用者は設計を理解するなど、今後は一人で何役もこなさなくてはいけません。また、何役もできるメンバーでプロジェクトを構成することが求められます。

そこで、OSS により構築している Cloud 基盤演習として、GRACE センターの所有するクラウドシステムにわざと不具合を与え、実際に一人が何役もこなす問題解決の流れを体験してもらったりもしています。

松本：これまでの開発プロジェクトは納品したら終了で、以降の問題は見えてきませんでした。そもそも日本では運用開始時にかなりの高品質が求められているため、「以降はバグが出ないのが当然」という風潮が強い。そのため、余計に納品後のことが見えづらいところがありました。これからは、たとえバグが出てもきちんと対処して、現場にフィードバックする仕組みが必要ですね。最近いわれている DevOps などは、まさにこれですね。アジャイルなどの手法はいかがでしょうか？

本位田：近ごろは Scrum などもごく普通に見受けられますが、受講生には、実例や演習中心で体験してもらう必要があり、教科書中心の講義では難しいのがネックです。そのため、教育の現場できちんと教えられる人は多くありません。GRACE センターにはトレーナーの資格を持っている講師がいますので、Scrum で開発する際のスキルセットを身に付ける教育をしてもらっています。

松本：開発の現場では、受発注の枠組みの中でアジャイルを実施しようとすると、いかに発注側の人間を巻き込んでいけるかがキーになります。ところが現状は、受注側と発注側の溝がなかなか埋まらない。それが、日本でアジャイルがあまり広まっていない原因のひとつではないでしょうか。発注サイドにも、開発手法のわかる人がいればよいかもしれません。

本位田：まさにその通りです。欧米と違って、日本ではベンダー主体となっている弊害ですね。

日本の長所を活かせるエンジニアに

松本：では、現場で求められるIT人材像とその育成について話を進めたいと思います。これだけソフトウェアを取り巻く環境が多様で複雑になってくると、ビジネスの仕組みや、IT融合などという点では農業や医療のことも知る必要があります。このように多くの知識を得なければならない状況で、そうした人材を育成するには、どうすればよいのでしょうか。

本位田：全体設計をするためにエンジニアに求められる幅が広がってきた今、大学での人材育成には、教員でもある研究者の養成が必要です。現状では、学際的な分野を自分のメインテーマとしている研究者が少ないのも問題のひとつでしょう。国が学際領域により予算を投資して、よい研究成果を出せる環境を整えなければいけません。そのプロジェクトに、ドクターや修士の学生が入れるようにできれば理想的です。

松本：確かにそうです。ただ、一つの分野でとんがったものを追求していると、学際の分野で世界に手が届く成果が出せるようになるのは厳しいのではないでしょうか。

本位田:だからこそ“官”が道を作らなくてはなりません。きちんと予算を投入して、学際領域を育てていく必要があるのです。ソフトウェアエンジニアにも、学際領域に関する深い知見と同時に、自分のアイディアを実現できるスキルが不可欠となってきます。

松本：開発スキルだけの話であれば、中国などにも優秀な技術者は大勢いるため、外注で済んでしまいます。そうならないために、アプリケーションのことなども総合的に理解している人材を育てなくてはならないでしょう。しかし一方で、町工場に息づく手作業を磨くなど日本独特の“ものづくり文化”を伸ばすことが、日本の生きる道だとする意見もあります。

本位田：海外の方が来日すると、やはりサービスのきめ細かさや安全性、正確性がフィーチャーされます。“おもてなし”の意識は日本の誇るべきカルチャーです。そうした文化があるからには、ソフトウェアシステムやサービスも単に機能を提供するだけではなく、相手に合わせた細かいカスタマイズなどの工夫をすることが大切なのではないでしょうか。日本人にもとから備わっている感覚をうまく活かすことで、国際的な差別化が図れるはずです。

松本：確かに、日本の鉄道は1分以上遅れると監視システムで表示が出るよう運用されています。海外の方が見学に来ると「1分なんて誤差だ」と驚かれるそうです。正確性という文化を実現できる緻密な技術があるというのも、日本の長所。その安全で正確な鉄道技術が海外で認められ、輸出されているケースもあります。そういった日本ならではの技術のグローバル展開がこれから求められていくでしょう。

本位田：技術力をベースとしたサービスの提供が重要で、ソフトウェア・エンジニアリングも、機能要求だけでなく非機能要求の取り扱いが大切ですね。

松本：今、自動車分野では「ISO26262」という機能安全規格が海外主導で定められています。一方で、性能のよい車というのは安全性だけがポイントではありません。燃費がよくて壊れにくい優れた日本車が生まれたのは、日本なりの細かな“すりあわせ”を続けてきた結果だと思います。最近は、IPAでもトヨタなどと共同ですりあわせのやり方も取り入れた開発方法論の海外提案を行い、標準となりつつあります。

本位田：ただ、すりあわせ文化はよくないという指摘もありますね。開発の速さだけからみれば、機能を追求し、コンポーネントを組み合わせて作ったほうがスムーズではありますが……やはりそれだけではありません。

松本：もちろん、ある程度はモジュール化して機能を分割していかなければなりませんが、やはり全体の細かな部分の課題は、すりあわせをしないと解決できないと思います。それができることを、日本のエンジニアの強み

にしていきたいですね。

本位田：そうですね。日本のある企業では、ずっと同じ部門で同じアプリケーションを作り続けているエンジニアもいます。ある意味では、これも強みです。ノウハウが形式化されていないといった弊害もありますが、こうした働き方ができるからこそ、すりあわせも可能になります。これまでのように米国の長所だけを追いかけて、日本の長所をつぶしていくようなやり方は、今後の発展に向いていません。

松本：まずは日本ならではの長所を活かして、ひとつずつ具体的な成果を出していくことが必要ですね。

得たスキルを現場で活用するためには？

松本：これからの日本に求められるエンジニア像が徐々に見えてきました。「トップエスイー」プロジェクトには企業に属している受講生が多いようですが、講義で得たノウハウや知見は修了後、どのようにそれぞれの職場で活かされるのでしょうか。

本位田：学んだことをそれぞれの現場でどう展開するかというのは、実は大きな課題です。現場に戻ってからギャップを感じ、深く悩んでしまう受講生もいます。そうした状況をサポートするため、カリキュラムの修了後も定期的な勉強会や、受講生とセンターが交流を保てるようなイベントを実施しています。

日本の企業には残念ながら、新技術の導入に否定的なカルチャーが存在します。「トップエスイー」が新しい技術を現場に持ち帰っても、なかなか受け入れられないケースがままあるのです。そのため、「トップエスイー」のカリキュラムには、どのようにプレゼンすれば社内でうまく展開できるかを検討する演習も用意しています。また、所属する現場の課題や開発プロセスのどこに、どの手法を導入すればいいのかを議論する講義もあります。この講義では、各受講生が持っているそれぞれの問題を取り上げて企画書を作り、ケーススタディを行うのです。

松本：具体的なケースを挙げると、企業文化の違いがあるのではないでしょうか。

本位田：もちろん様々な違いはあります。しかし、それがまた大きな刺激になるのです。また、国内外の文化の違いを考慮することも重要です。例えば、著作物の問題例では、O社が提供しているAPIをG社のシステムがそのまま利用したことで、O社がG社を訴えた裁判があります。しかし、APIを実現するコードが異なっていたため、裁判ではG社の主張が通りました。「トップエスイー」プロジェクトではこの実例に基づき、原告・被告・裁判官に別れて模擬裁判をする演習を実施しました。

松本：非常に興味深いですね。企業内でも大学でも難しい内容なので、そのような教育を行っているところは日本ではほかにないのではないでしょうか。

本位田：はい。おそらくないと思います。自分たちが優れたサービスを提供しても、輸出したとたんに訴えられる可能性もあるということを知る、よい演習です。

松本：しかし、エンジニアがそこまで担わなくてはいけないのは大変ですね。

本位田：法的な問題が起きたとき、日本の企業ではたいてい法務部門などが処理を担当しますが、ベンチャー企業にはそうした部門がない場合もあります。証人として、開発担当者が入廷しなければいけない事態もあるでしょう。

松本：日本車がアメリカで訴えられた事例があります。電子スロットル制御システムのソフトウェアが問題となったのですが、自動車メーカーはそれが正しいことをきちんと説明できませんでした。エンジニアは力を尽くしたようですが、最終的には第三者に解析を依頼したことで問題がないと証明でき、解決につながったようです。この事例を通じて、ソフトウェアに問題がないことを、第三者にきちんと説明できないといけない時代が来たことを痛感しました。

本位田：エンジニアの担う役割は、広がっていく一方ですね。

松本：IPAでも“未踏人材”を育成する活動を行っています。特定分野でとがっているだけでなく、幅広い視点を持ち、新たなイノベーションを考え出せる、突出した若手の人材です。ただ、そういった人材はうまく企業に貢献できないケースも見受けられます。国を挙げて、突出した人材に活躍の場を与える仕組みが必要なのかもしれません。

本位田：確かに、企業ではそのような突出した人材が評

価されにくい文化がありますね。それよりも、プレゼンテーションやネゴシエーション、マネジメントに秀でた人物に目がいってしまうのでしょう。

松本：「出る杭は打たれる」といいますが「出過ぎた杭は打たれない」という説もあるので、周囲が上手に伸ばしていけばよいのかもしれません。

本位田：突出した人材をうまくマネジメントできる上司も必要になりますね。知識やスキルの体積が同じだとすれば、それが横に広いか縦に深いかという問題。各個人に合わせた育成プログラムや、総合的な観点での評価が欠かせないはずです。

松本：並行して、基本的な IT リテラシーを持った層を底上げすることも重要な課題です。誰しもがスマートフォンを使える現代では、やはり使う側が気を付けなくてはならないこともあります。

本位田：「人材はピラミッドを構成している」といわれますが、この底上げを考えた場合、ピラミッド全体を引き上げるのは大変でも、最上位に位置する少人数のトップとなる人材を育成することは可能です。スキルの高い人材が全体を引っ張っていく仕組みを構築できれば、おのずとメンバーはついてくるでしょう。これが「トップエスイー」が目指している効果です。

松本：今、ソフトウェア業界では、2015 年問題が大きなテーマとなっています。マイナンバー法案が成立して実際に動き出すため、開発でソフトウェア・エンジニアの手が取られる。そのほかのシステムも更改期を迎えるタイミングで、人材が深刻に不足することが懸念されています。従来は大手のエンジニアが仕上げてきたインフラなどの仕組みを、人手が足りないために中小企業のエンジニアが担う場面もあるでしょう。コストや工期が限られる不景気の中、「仕上がれば OK」として開発を進めてきた中小企業のエンジニアは少なくありません。こうした人材をきちんとフォローして、スキルを磨いていかなければ、2015 年問題だけでなく、2020 年の東京オリンピックを乗り越えられないと指摘する声もあります。そういった人材の育成に IPA も積極的に貢献していきたいと思います。

本位田：よく、家作りとシステム作りを対比されますが、家作りでも専門性の高い企業が多く参画しています。システム開発でもエコシステムと呼ばれる、専門性の高い企業が役割を分担して連携する姿になってもよいと思います。特に中小企業においては、得意分野を磨いてアピールしていくことが必要になっていくのではないでしょうか。

松本：確かに、それぞれが得意分野をより明確にしたほうがよいですね。「リアルタイム性に優れたソフトを作るのが得意です」といったアピールは、あまり聞かれません。価格が主体の評価が行われている現状では、実現が難しいのかもしれませんが……。

“産”×“学”のハブとなる IPA と NII

松本：日本のソフトウェア工学の人材育成では大学の教育に期待したいところですが、産業界とのニーズには、まだギャップがあるように思います。最後に本位田先生から、SEC に対する期待についてお聞かせください。

本位田：ソフトウェア工学の教員は各大学で 1、2 名と大変少ないのですが、全国の大学の教員が連携すれば、大きな力になります。国立情報学研究所は大学共同利用機関なので、大学とは一線を画します。多くの大学の教員が連携するためにハブの役割を担う必要がありますし、そのような場を提供していきたいと考えています。

IPA は“産業界のハブ”、国立情報学研究所は“学のハブ”として連携すれば、産×学の距離が近づき、点と点のつながりではなく面と面のつながりが生まれ、国際競争力の強化にもなると思います。

松本：ぜひお互いに連携して、世界をリードする実践的な人材を育てていきたいと思います。本日はありがとうございました。

安全なソフトウェアシステムを実現するための新たなアプローチ

マサチューセッツ工科大学 教授
ナンシー・レブソン

×

SEC 所長
松本　隆明

IPA は 1 月 21 日、システムやソフトウェアの安全性に関する特別セミナーを開催した。登壇者の一人は、その分野の世界的な第一人者であり、「Safeware」や「Engineering a Safer World」などの著作で知られる、マサチューセッツ工科大学のナンシー・レブソン教授である。ソフトウェアに起因する障害やその分析、原因を導き出す方法といった安全性向上に向けた様々な対策を、先進的な手法と適用事例を含めて話していただいた。システムが複雑化している今、旧来の分析手法だけでは十分ではない。新しく、かつ安全なシステムを実現するためのアプローチについて、セミナーに先駆けてレブソン教授に伺った。

松本：本日はソフトウェアシステムの安全性について、世界でも第一人者のレブソン先生からお話を伺いたいと思います。安全性とは何か、システムの安全性を確保するにはどうしたらよいかを考察していきます。

レブソン：わかりました。まずは、このような機会をいただいたことを大変うれしく思っております。

ナンシー・レブソン

米国マサチューセッツ工科大学　航空宇宙工学部門及び工学システム部門教授。全米工学アカデミー会員。
ソフトウェア安全という新分野の創設者であり、現在も世界の第一人者。
これまでに 200 件以上の論文を執筆し、その研究結果や手法は、航空宇宙、交通運輸、科学プラント、原子力、医療機器など、安全に係わる多種多様な産業で応用されている。
代表的な著書に「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―」があり、最近の著書「Engineering a Safer World」(MIT Press) は 2012 年に出版された。

松本：こちらこそお越しいただき、ありがとうございます。では初めに、先生のご著書「Engineering a Safer World」の内容からお伺いします。本の中では、これまでのエンジニアリングの安全性に関する仮説と逆説がいくつか紹介されていますね。

安全性と信頼性の混同があるという仮説があります。この大きな違いというのはどこにあるとお思いですか。

レブソン：まず前提として、安全性や信頼性に関するこれまでのエンジニアリングが手法として確立されて以降、世界は変わりました。そもそも、FTA※1 や HAZOP※2、ETA※3 といったテクニックや確率論的なリスク分析が発明されたのは、コンピューターが広く普及する前です。その後、様々な場面でコンピューターが活用されるようになり、システムはより拡大・複雑化しました。これによって、ソフトウェアやコンピューターはそれまでのハードウェアとは異なり、エンジニアリングの方法を変えていったのです。

システムがシンプルだったときには、電子機械的な面で使用前に様々なテストを実施し、ほとんどの設計エラーを排除することが可能でした。しかし、コンピューターではあらゆるテストをし尽くすことは不可能です。どうしてもシステムの中には、設計のエラーが残ってしまいます。いま安全性に関する問題のほとんどは、そのエラーが原因になっているのです。安全性の問題は必ずしも、コンポーネントの故障に拠りません。

もうひとつの問題は、システムの複雑化によって、人間に拠る手動制御が難しくなったということです。システムを理解していない人間の起こすエラーは、ほとんど排除できない設計になっています。それにも関わらず、事故が起きたときには人間に責任があるといわれる。現在では、コンピューターが使われるようになる前に開発された安全・分析の古い技術は、もはや成り立っていないということです。

【脚注】

※1　FTA：フォールトツリー解析。ハザードの原因となりうる個々の障害の組み合わせをブール理論を用いて記述するトップダウンの検索方法

※2　HAZOP：設計で期待した運用から考えうる全ての逸脱と、その逸脱に関連するすべてのハザードを識別する手法

※3　ETA：イベントツリー解析。ディシジョンツリー形式で、何らかの故障とそれに続いて起こり得る一連の事象の結果を識別するために順方向検索を用いる手法

私が「Engineering a Safer World」を執筆したのは、これまでとは異なるタイプの問題が起き始めている現状を伝えるためでもあります。多くの複雑なソフトウェアから成るシステムで、新技術を活用し、どのように安全性を向上させていけばよいのかを説明したいと考えたのです。

松本：つまり、大きな問題は２つ。ひとつは、複雑で大規模なシステムではコンポーネント同士のインターフェースが非常に増えるため、コンポーネントの安全性のみに着目するだけでは不十分だという点と、今までは機械的なもので人間の関与がほとんどなかったのに、現在のシステムにはオペレーターやユーザーといった“人”の関与が多いという点でしょうか。

レブソン：そうですね。ただ、問題はコンポーネント間の相互作用あるいはインターフェースだけではありません。まずは、システム設計がソフトウェア開発の一部になってきていることに注目しなくてはならないのです。実は多くの事故が、特別な故障のないコンポーネントの中で起きています。確かにコンポーネント間の相互作用を制御するのはソフトウェアです。そのうえシステム設計や、従来ならオペレーターが担っていた作業も、ソフトウェアが行うようになってきました。

私はソフトウェアに関する数百件もの事故を調査・研究してきましたが、すべての事例でソフトウェア自体は要件を満たしており、実装も正確でした。しかし、もともとの要件が間違っていれば、コンポーネントが適切であっても事故は起こるのです。問題がソフトウェア・エンジニアリングではなくシステム・エンジニアリングの側にあることは、珍しくないのです。システム・エンジニアはソフトウェアを理解していません。事故をなくしていくためには、ソフトウェア・エンジニアとシステム・エンジニアが協力し、お互いに学び、関わり合うことが必要なのです。

松本：その点についてはまったく同感です。私はソフトウェア・エンジニアですが、システム・エンジニアの方と話すと、考え方に相違があると感じます。特に、ソフトウェアは物理的な法則に基づく装置や機械と違い、ロジカルに考えて作るため、発想の仕方から異なるように思います。ただ、システムの安全性を上げてゆくためには、当然ながら個々のコンポーネントの信頼性を上げていくことも必要ですよね。

レブソン：はい。ただし、どれだけ信頼性の高いコンポーネントだったとしても、システムの問題が設計部分にある場合は、安全性の向上にはつながりません。

松本：個々の信頼性を上げるだけでは十分でないということですね。最近のソフトウェア開発では、要求に合ったソフトウェアなのかをチェックしたり、上流工程での設計品質を上げたりすることを重要視します。そこで最も大切な問題は、きちんとした要求条件をどのように正しく策定するかではないでしょうか。

レブソン：はい。それが現在できる対策だと思います。新しい分析の手法によって可能になってきました。旧型のアプローチでは、ハザードや安全の分析を設計の中で取り入れていかなければなりません。しかし、システム理論に基づいた新しいアプローチを使えば、コンセプトを作る早い段階で分析が可能となるため、形式的手法により数学的に安全要件を導き出せます。ただ、そうした流れを作るには、従来のような信頼性の理論からシステム理論、システム・エンジニアリングの考え方に移行していく必要があるでしょう。

松本：安全性要求の考え方を、システム的な観点からきちんと決めていくことが重要ですね。

ルートコーズを求めるだけでなく全体像をとらえる

松本：それでは、著書にある２つ目の仮説の話に進みます。先生の考案した事故原因の分析モデルについてです。まず、今までイベントの連鎖として考えていた分析方法は、必ずしも正しくないのでしょうか？　私たちは事故の原因分析をする際、やはり「ルートコーズ（根本原因）」の発見を一番の目的にしてしまいがちです。

レブソン：イベントの連鎖がないということではありませんが、複数の原因が絡み合う相互作用や関係を考えていく際、イベントの連鎖を使うとどうしても限界が出てきます。直接の因果関係や原因だけでなく、間

松本 隆明（まつもと　たかあき）

1978 年東京工業大学大学院修士課程修了。同年日本電信電話公社（現 NTT）に入社、オペレーティング・システムの研究開発、大規模公共システムへの導入 SE、キャリア共通調達仕様の開発・標準化、情報セキュリティ技術の研究開発に従事。2002 年に株式会社 NTT データに移り、2003 年より技術開発本部本部長。2007 年 NTT データ先端技術株式会社常務取締役。2012 年 7 月より独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）所長。博士（工学）。

接的な関係に注目しなくてはならないのです。
東日本大震災の福島を例に考えてみましょう。原発に事故が起きた原因は、津波や外壁だけではありません。政府の諸機関と関連する会社の仕組みも、おそらく問題のひとつだったことでしょう。もし直接的なルートコーズだけに注目すると、福島の原発事故では津波や防潮壁の高さ、電源の設置場所にのみ焦点を当てることになります。原発の問題にとどまらず、全体的な社会の安全という視点で問題解決を図り、将来に備えていくのならば、本来は全体像をとらえる必要があります。つまり、原因になりえたすべての事象に目を向けるべきなのです。
人はシンプルな解答を求めるあまり、根本原因さえ直せばよいというシンプルな答えにたどりつく傾向があります。ひとつの原因を解消しさえすれば、問題はすべて解決できると思いたいのです。こうした根本原因の概念は、システム制御に関する幻想だといえるでしょう。マネージャーは自分でコントロールできると思いたいけれど、ひとたび大きな問題が起きれば圧倒され、コントロールが不可能な状態に陥ることもある。私たちが見ているのは、大きな問題の中にある一現象にすぎません。視野が狭いために、同じ根本から別の事故が発生してしまうこともあるのです。
米国で、沿岸警備隊が起こした複数のヘリコプター事故に関する調査が行われました。通常の手法で分析した際には複数の事故に類似性はなく、度重なる事故は偶然の一致であるように見えていました。そこに、私たちの新しい事故分析の手法を使ったのです。すると共通要因が見いだされ、的確な解決を果たすことができました。
松本：原因は必ずしもひとつに集約されるわけではなく、複数の要因が絡んで事故が起きるのですね。ただ、イベントの連鎖を遡って考えること自体は悪くないという認識でよろしいでしょうか。
レブソン：イベントの連鎖は間違った考え方ではないですが、十分ではありません。事故に影響した直接的な原因にとどまらず、因果関係のモデルを拡張して考えていかなければならないでしょう。
現代社会では、様々な新技術が使われています。ソフトウェアはもちろん、システムの複雑性も増しています。こうした時代に安全問題を考えるには、単なるイベントの連鎖を越えて、社会的システムとして思考を巡らせていかなくてはなりません。考え方を抜本的に変えていく必要もあるでしょう。新たな方法を展開していく場面で、人は現在の考え方を少しだけ変えようとしがちですが、それではうまく機能しません。ソフトウェアやシステムの現状を踏まえた、パラダイムシフトが必要になります。従来とはまったく異なるやり方で、システム・エンジニアリングを考えていかなければならないのです。
私たちが集めた多くの事例では、新旧様々な手法で分析しましたが、やはり新しい手法を使ったほうがより多くの原因や予防策を見つけられました。最近、米国の EPRI（米国電力研究所）では、原子力発電所の設計を複数の専門家に渡し、FTA、ETA や HAZOP などを含めた異なる手法で、それぞれ分析してもらう研究を実施しました。そこで私たちの手法「STPA」を使ったグループは、ほかよりも多くの問題を見つけることができました。今までその原子力発電所ではいくつもの事故が起こっており、それは分析者たちには事前に知らされていませんでしたが、私たちのグループのみが実際に起こったその事故を分析から発見できました。問題があったのが、コンポーネントではなくシステム設計だったからです。様々な業界で同種の例が多数あり、コンポーネントの信頼性が安全に関わるという古い考え方は、現在は十分ではないといえるでしょう。

適切な分析を実施するための取り組み

松本：分析をする際は、技術的な問題はもちろん、社会システムの問題や人間の精神的・感情的な問題といった様々な側面から原因を考えなければならないですね。しかし、そのためには幅広いスキルを持った人材が必要になります。その点はどのような解決法があるでしょうか。
レブソン：今後は、システム的な考え方のシステム・エンジニアリングの教育がより重要になってくるでしょう。また、分析の作業をグループ単位で行うことも必要です。ソフトウェア・エンジニアやシステム・エンジニア、アプリケーション・エンジニアたちが一丸となって問題に当たらない限り、本当の解決には至らないと思います。コンピューターに命令を与える機能しかないソフトウェアは、エンジニアたちにとっては胃潰瘍の原因になるかもしれませんが（笑）、それ自体が人に傷害を負わせたり、火災や爆発の原因になることはありません。つまり、ソフトウェア自体に安全が欠如しているというわけではないのです。しかし、安全性の欠如しているシステムの中にソフトウェアがある場合は、危険を生み出しかねない。システムは、火災や爆発といった事故につながってしまう可能性があるのです。ただ、コードを実装するだけのソフトウェア・エンジニアもいるでしょうから、全員がその責任を担わなければならないとは思いません。シス

テム・エンジニアと協業しなければいけないソフトウェア・エンジニアの数は限られるといえるでしょう。

松本：私たちは、各企業から過去の事例を収集して、お互いに共有する事業に取り組んでいます。

レブソン：事例の共有も有効ですね。ただ、残念ながら、事故が起こる原因はほとんど毎回異なるものです。

松本：本当にそうですね。現象を見ると、実は類似性のある事故は頻発しています。たとえば、日本では最近、社会を支えるインフラシステムに障害が起きました。あらかじめサーバーを二重化しておき、片方が壊れた際はもう片方に切り替えるのですが、その切り替えの失敗が多発しているのです。通信、証券、金融、流通など様々な業界で、切り替えがうまくいかず、システム全体がダウンしてしまう事態が起きています。そこには何か共通の原因があるのではないかと思い、現在、分析をしているところです。様々な側面から検討していますが、単に技術的な問題だけでなく、システムの裏には企業ごとの文化や管理方法などもあるため、同じ原因が存在するといえるのか、判断が難しいと考えています。その点をしっかり把握しないと、事例を教訓として共有する意味が薄れ、汎用性がなくなってしまう。そこで私たちは、できるだけバックグラウンドを整理し、事象が起きた状況をコンテキストにまとめるようにしています。そうした活動についてのご意見をお聞かせください。

レブソン：冗長性やコンポーネントの再利用は、ハードウェアの信頼性に対して主たる解決策ですが、ソフトウェアには使えません。冗長性を持つことは、同じエラーを繰り返してしまうことだともいえます。再利用についても同様ですね。過去 10 年間で起きたすべての宇宙船事故の事例では、ソフトウェアが再利用されていました。同じコードが別の宇宙船で使われ、事故が発生していたのです。

松本：冗長性の問題は、ソフトウェアではなかなか解決できません。しかし、事故が多発している現状を踏まえ、ソフトウェアが直接的な原因でないとしても、実態を明らかにする必要性があるのではないでしょうか。

レブソン：共通の要因を探すのは、確かによいことです。ただ、その作業にばかりとらわれて、違う要因だけれどもお互いに結びついているかもしれない部分をなおざりにしないよう、気を付けなければなりません。

たとえばある日本車メーカーの負けた訴訟では、ソフトウェア・エンジニアリングが適切に行われていなかったことが問題となりました。実はその訴訟の中で、“ソフトウェア・エンジニアリングが適切に行われていなかった”ことと“意図しない加速”の因果関係は証明されていません。それにも関わらず、裁判には負けてしまった。この例からも、航空や軍事の分野では長く行われてきたハザード分析やセーフティー・エンジニアリングを、自動車業界にも取り入れていくべきだということが見えてくるでしょう。これまでどおりコンポーネントの信頼性にばかりフォーカスしていては、新しくて複雑な車には機能しません。自動車業界の方は、様々な機能間の相互作用が大きな問題のひとつだとおっしゃっていました。新たな機能を搭載したとき、機能間の意図しない相互作用があるそうです。たとえば、ある状態のときにヘッドライトとワイパーの相互作用で問題が生まれるなど、以前のシンプルなシステムでは予想もできなかった事態が実際に起きています。先ほどの訴訟となったケースも、もし問題があったなら、システム的な見地から見れば発見できたことでしょう。ソフトウェア・エンジニアリングだけでなく、システム・エンジニアリングもやり方を変えていかなければならないのです。

松本：航空業界や軍需業界で行われていた手法を、自動車業界では取り入れていなかったということですね。米国には、そうした問題を解消するための、業界をまたいだ情報共有の仕組みはないのでしょうか？

レブソン：残念ながらありません。このケースでは、従来のやり方に従っていたことが問題だったと思います。航空や軍需の分野は、複雑なシステムと長く向き合ってきたため、手法が開発されました。しかし、これまでの自動車業界では、車全体は複雑なものでもコンポーネントに分割できたので、コンポーネント間の相互作用は限られていると考えられてきたのです。現在私たちは数社の大手自動車企業と協業し、このような問題に対処するために必要なシステム・エンジニアリングの変更を提案しています。

松本：カーナビゲーションシステムのソフトウェアに不具合があり、ブレーキがおかしくなったという話を聞いたことがあります。今までは独立していると思っていた部分がそれぞれ密接に関連して動いていたり、自動運転の登場も間近になっていますから、今後の自動車メーカーはますます車を複雑なひとつのシステムとして認識しなくてはなりませんね。

レブソン：おっしゃるとおりです。また、様々な相互作用を考えるには、車内のコンポーネントだけでは足りません。車を取り囲む環境や道路の設計からドライバーのソーシャルな部分まで、今まで考慮されていなかったことにも目を向ける必要があります。

エンジニアはほとんどのシステムにおいて“人”の部分を無視してきてしまいました。物理的なパーツのみを設計し、オペレーターが適切に使用してくれることを勝手に想定していたのです――たとえば、ブレーキに問題が起こっても、運転手が適切に対応してくれるというように。しかし、システムがあまりにも複雑になってしまった今、そうした想定は成り立ちません。システムが複雑になったことで人に責任を転嫁するのではなく、人はユーザーとしてだけでなく、システムを構成する重要な一部分だと考えなくてはならないのです。

IT のセキュリティも、ひとつの例です。ユーザーが通常できないようなことを、当たり前にさせる想定でセキュリティが組まれています。そうなると、問題があったときに責任をユーザーに押し付けることになる。たとえば IT 管理者が、とても難しいインストールの設定を適切にできなかったり、複雑なパスワードを覚えられなかったりした場合、人が悪いことになるのです。長くて意味のない数字や文字が並んだパスワードを、書き留めずに何百も覚えられる人がいるでしょうか？　これは、想定条件が悪いと言わざるを得ません。

松本：パスワードは頻繁に変更することが求められますが、正直覚えきれないのではないでしょうか。

レブソン：これは、セキュリティの進歩が足りない点だと思います。本来ならユーザーをシステムの一部として扱わなければならないのに、敵として扱ってしまっているのでしょう。

発注側と受注側の距離を埋める

松本：次の話題ですが、最近のシステムは、ソフトウェアがほとんどをコントロールしているといっても過言ではありません。その安全性を確保する困難さについて、お話を伺います。まず、ソフトウェア開発においては要求条件を考える人と実際に作る人が異なる場合が多いため、お互いにうまくコミュニケーションをとることがとても大切です。日本は米国に比べ、発注側と受注側の距離が離れている傾向が強い。要求条件を決める人が自らソフトウェアを作る“内製”が多い米国では、日本よりコミュニケーションの問題が少ないのではないかと思いますが、いかがでしょうか。

レブソン：比較できるほど日本について詳しくはありませんが、コミュニケーションが安全の大きな要求事項だというのはそのとおりだと思います。ビル・カーティスの調査では、100 以上の大規模なソフトウェア・プロジェクトを対象に、要因分析を行いました。ほとんどが失敗したプロジェクトだったのですが、成功例と失敗例の違いを調べたのです。大きな違いとして、成功したプロジェクトには、ソフトウェアが使われるシステム全体を理解している人が常に携わっていたことがわかりました。

解決策や進め方には 2 つあると思います。まずひとつは、最近、システム分析がおろそかになってきており、それを元の状態に戻してシステム分析を充分かつ綿密に行うこと。米国でも、医療のウェブサイトで「オバマケア」の大きな問題がありました。通常の原因は、開発者がソフトウェアの要求を正しく理解しないまま、業務に当たっていることです。ソフトウェア・エンジニアとプログラマーがやる仕事はそれぞれ違うのに、あまり職務の区別がされていないのも問題でしょう。プログラマーはソフトウェア開発全体での設計・実装の詳細を理解している人ですが、ソフトウェア・エンジニアは多くの企業で“コードを作る人”が昇進して大きな肩書きとしてソフトウェア・エンジニアと称しつつ今までと同じ仕事をしている場合がありますが、必要なスキルも経験も異なります。プログラマーとソフトウェア・エンジニアとでは実際やる仕事も、必要とされるスキルも異なります。先ほどお話したカーティスの調査でも、成功したプロジェクトでシステム全体を理解し責任を持っていた人間の多くはコード作りに関しては優秀でなかったという結果がみられました。

日本のように発注者と受注者が異なる場合は特に、両社間できちんとコミュニケーションがとれるインターフェースを持たなくてはなりません。たとえば軍需の分野では、プライムコントラクターが全体の責任を持ちます。もちろん大きなプロジェクトでは多数の会社に委託しますが、それらのインターフェースをつなぐのはプライムコントラクターの役目です。それが、オバマケアのウェブサイト制作では欠けていました。問題が起きた後、委託会社はそれぞれ自社に非がないことを主張していました。実際に各コンポーネントは問題なく機能しており、単体でのテストも注意深く実施されていたのです。

的確に要求仕様をまとめるためには？

レブソン：2 つ目の解決策は、要求の仕様固めです。多くの現場で、リクワイアメント・エンジニアリングがなおざりにされてしまっているのが問題です。デザインの仕様ととらえられ、機能要求として考えられていないケースも多々あります。私は本領域における 20 年ほど

の経験を活かし、要求の仕様固めについて著作でも論じてきました。現在の要求のまとめ方は、決して十分ではありません。その要求が仕様になった根拠や理由は記述されていない場合がほとんどです。使用時の様々な前提条件や、なぜそのような使い方になるのかという理由・想定についても書かれていません。ユーザーに必要なトレーニングやアクションなどは、想定条件にすら入っていないのが現状です。

松本：わかりやすい仕様書を作ることも大きな課題です。文章はどうしても表現が曖昧で誤解を生じやすくなってしまいます。先生のおっしゃるように仕様の理由を書いたとしても、なかなか伝わりにくいのではないでしょうか。

レブソン：確かに、それは大きな問題ですね。形式手法・数学的手法が解決策としてよく挙げられますが、それだけでは難しいでしょう。ソフトウェア・エンジニアには理解しかね、エラーの特定につなげられないのです。

私は 1990 年、米国政府の依頼によって「TCAS」という航空機衝突回避システムを認証しました。非常に複雑なシステムで、曖昧ではない仕様書を作るのが困難でした。使われているロジックも大変難しく、政府も認定に難色を示していたほどです。そこで私は、スペシフィケーション言語を作りました。数学的な基礎に基づいた言語なので、曖昧ではありません。誰もが 10 分で学ぶことができ、エラーを見つけられます。なかには意図を示す仕様やトレーサビリティ、機能要求も入っているため、「インテント・スペシフィケーション」とも呼んでいます。JAXA でも有用だとして、使われているものです。形式手法のコミュニティでは十分な数学的根拠がないと考えられているようですが、根底では数学的モデルを用いているため、数学的分析をすることも可能です。たとえば一貫性や完全性に欠けているものは、このモデルで導き出せます。

「Engineering a Safer World」の後半でも触れていますが、それぞれが商用ツールとしてシステムを提案していることが問題なのかもしれません。ツール開発者は自分のツールを使ってほしい気持ちが強いため、なかなか新しいアイディアが導入されないのです。ただ、私のインテント・スペシフィケーションは、有用性が実証されています。今使われているひどいツールを乗り越えていかなければ、問題は解決できません。

新しい因果関係のモデル「STAMP」

松本：お聞きしたいことがたくさんあるのですが、時間が迫ってきています。最後に、先生が提唱されている STAMP についてご紹介いただけますか。

レブソン：はい。STAMP は新しい理論的因果モデルです。因果関係に対する考え方なので、ツールでもテクニックでもありません。STAMP 以前の古いモデルはイベントの連鎖に基づいたもので、200 年も前から存在しています。現在の様々なツールは、こうした古いモデル上に構築されました。STAMP はそれをさらに人・環境にまで広げていくという考え方で、ソフトウェアが果たした役割まで考えを広く巡らせます。

松本：STAMP の特徴は、安全性を信頼性の面ではなく、コントロールとコンストレイント（制約）としてとらえることだと思っているのですが、その理解でよろしいでしょうか。

レブソン：そうです。故障の予防という従来の考え方から、ユーザーの行動を制約し実行させるというコンセプトに変えていくべきだと思っています。たとえば人間や政府の行動、組織の文化・風土、管理方法といった要素が、制約に対して違反する可能性をはらんでいます。そうした部分を変え、システムやコンポーネントの挙動に対して安全に関する制約を設けていくのです。

また、セキュリティについても攻撃や脆弱性を考えるだけでなく、制約を実行していかなければなりません。これは新たな提案なので、まだ書籍でも触れていない内容です。STAMP の考え方を土台に新たなツールを作っていけば、よりパワフルなツールが生まれます。STAMP は旧来の考え方を含んだうえに、さらに広げたものだからです。

松本：ソフトウェアシステムの安全性に関して大変貴重な示唆を伺うことができました。今後の IPA/SEC の取組みにもぜひ活かしていきたいと思います。本日はどうもありがとうございました。

SEC 特別セミナー「Engineering a Safer World」について

独立行政法人情報処理推進機構（IPA）
技術本部 ソフトウェア高信頼化センター（SEC）ソフトウェアグループ リーダー
中村　雄三

1. はじめに

近年、システムの複雑・大規模化に伴い、システムの安全性を確保することがより重要になっている。そこで独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）では、2014 年 1 月 21 日に、マサチューセッツ工科大学のナンシー・レブソン教授と有人宇宙システム株式会社の星野伸行主幹技師を招いて特別セミナー「Engineering a Safer World ―安全なシステムを実現するための新たなアプローチ（手法と事例）―」※1 を開催した。

ナンシー・レブソン教授（以下「教授」と略す）は、安全なシステムとソフトウェアを目指した研究の第一人者であり、国内では「セーフウェア」の著者としても著名な方である。今回のセミナータイトルは、最新の著作「Engineering a Safer World」※2 からいただいたものであり、講演では著書の中で新たに提唱されている「システム理論に基づく事故モデル（STAMP※3）」、「STAMP に基づく安全解析手法（STPA※4）」、及び「STAMP に基づく不具合分析手法（CAST※5）」を中心に事例を交えてご講演いただいた。また星野主幹技師には、STAMP/STPA の概要を補足していただくとともに、宇宙機における実際の適用事例に関してご紹介いただいた。

2. Engineering a Safer World（ナンシー教授）

ここでは教授の講演の主なポイントを概説する。

「安全」と「信頼性」は異なる

システムを構成する個々のコンポーネントの信頼性が高くても事故は起こりうる、逆に個々のコンポーネントの故障があっても事故につながるとは限らない。最近の事故は、コンポーネント間の相互作用によるものが多く、システム全体の安全性とそれを構成する個々のコンポーネントの信頼性は別物、つまり個々のコンポーネントの信頼性を上げたからといっても、システム全体が安全とはいえず、システム全体にわたる安全性の考慮が重要である。

例えば、ソフトウェアが要件通りに極めて高い信頼性で開発されたとしても、システム安全に関する要求の欠如、或いは特定の動作条件に関して要件で規定されていない場合には、安全とは言えない。

従来の「安全工学」、「信頼性工学」の問題

従来の安全工学、信頼性工学は、コンピュータが普及するはるかに前に考案されたものであり、比較的単純な電子機器システムを対象にしていた。しかし近年は、あらゆるところにコンピュータ制御が入り、システムが大規模・複雑化してきており、従来の手法では安全に対する対応ができなくなっている。例えば、操作者の役割も、装置を直接制御する事から、装置を制御するコンピュータを監督する、というように変わって来ている。また、システムが単純な頃はほぼすべての試験が可能であったが、システムが複雑化したことによりすべてを試験することが難しくなっている。

このような中で、個々のコンポーネントの不具合ではなく、システムの上位レベルの設計上の誤り等により事故が発生する事、システム全体が複雑化し、人間が全体を把握し制御することが難しくなっているにもかかわらず、何かあるとシステム設計上の問題ではなく操作者に責任を負わせようとする傾向がある、といった状況になったため、これに対処できる新たな方法論が必要となった。

従来の「事故モデル」の問題

事故モデルには 2 種類が考えられる。①単一、または複数のコンポーネントの不具合から生じるもの、②コンポーネント間の相互作用により生じるもの。近年では、コンピュータやソフトウェアが導入され、コンポーネント間の密接な結合や複雑な相互作用により、②に起因する事故が増えている。このような事故に関しては、従来の「複数の原因の連鎖による事故モデル（ドミノモデル）」

【脚注】
※ 1　http://sec.ipa.go.jp/seminar/20140121.html
※ 2　http://mitpress.mit.edu/books/engineering-safer-world
※ 3　STAMP：System-Theoretic Accident Model and Processes
※ 4　STPA：System-Theoretic Process Analysis
※ 5　CAST：Causal Analysis using STAMP

で捉えようとすると限定的になるため、全体を見て原因になりうるすべての要因に焦点を当てるべきである。

システム理論に基づく新たな事故モデル（STAMP）

事故は、複雑で動的なプロセスにかかわるため、従来のようなシステム内での単純な不具合イベントの連鎖ではなく、システム、操作者、環境も含めた相互作用の中での動的な制御の問題として捉える。システム全体に対して安全を確保するためには、安全のための一連の制約を実施する必要がある。事故は、システムコンポーネント間の相互作用が、この制約を乱した時に発生するものと考える。安全でない制御動作は以下の 4 種類に分類される。①安全のために必要な制御コマンドが送出されない、②安全でない制御コマンドが送出された、③基本的に安全なコマンドの送出が早すぎた／遅すぎた、④制御の停止が早すぎた／長すぎた。このように安全性の確保とは、信頼性の問題ではなく制御の問題である。従って、安全に関する従来との違いは、「不具合を防止する」のではなく、「システムの振る舞いの中で安全のための制約（safety constraint）を確実に実行する」ということになる。教授はこの点に関して、講演では幾つかの事例を挙げて説明された。

さらに、従来の観点では、操作者のミスに起因した事故に対して、対策として①さらに自動化を進める、②ルールや手続きで厳しく規制する、との考え方であった。一方、新たな考え方では、操作者のミスは原因ではなく、全体システムの設計を見直すべき兆候（symptom）と捉えている。

STAMP に基づく安全解析手法（STPA）、及び STAMP に基づく不具合分析手法 (CAST)

ここでは誌面の関係で、STPA、CAST の手法の内容に関しては説明を省略する。教授は毎年、STAMP ワークショップを開催されており、これらの手法の適用事例が数多く集まっているとのことであった。実際に、航空機、宇宙機、航空管制、自動車、化学プラント、原発等の分野に適用されているとのこと。それらの経験から STPA に関しては、従来の FTA、FMEA 等の手法と同等以上のハザードを、しかも従来よりも少ないコストで実施できた、と報告されていた。

教授の講演に対する主な質疑応答として、一つは従来の手法から新たな手法に切り替える判断をどうしたらよいかの質問に対しては、比較的小規模からの試行適用が良いのではないか、もう一つは現時点での手法適用が専任の操作者を前提とした化学プラント等の産業界での分析が中心になっているが、一般利用者を対象とした場合どのような違いがあるかに関しては、現在、自動車メーカー等と検討を進めている、との事であった。

3. 宇宙機での事例紹介（星野主幹技師）

続いて、有人宇宙システムの星野主幹技師による STPA 適用事例の講演があった。今回の事例では、システムの IV&V（独立検証及び有効性確認）で検証対象を絞るため、ハザード解析技術として STPA を FTA 等と併用、不具合分析技術として CAST を他の技術と併用されているとのことであった。星野氏はまず STPA に関して実際の適用の流れを説明されるとともに、宇宙機 HTV(コウノトリ) での適用事例に関して紹介された。

特徴的であったのは、STPA により従来手法である FTA では識別されなかった幾つかの要因が検出された点であった。さらに、最近の取組みとして、操作者に関してヒューマンメンタルモデルを考慮した分析手法の紹介もしていただき、非常に興味深いものであった。

4. おわりに

教授の日程調整等のため、特別セミナー開催の周知が年末になってしまったにもかかわらず 200 名近い方に参加いただき、活発な質疑応答も行うことができた。両氏の講演は、今後の大規模・複雑化する製品・システムの安全を考える上で、極めて参考になるものと考える。また STAMP/STPA,CAST に関しては、国内ではまだ適用事例が多くないと考えられるが、今後、試行評価等が進められ、安全のために寄与する事が期待される。

最後に、このセミナーのために来日いただいたナンシー教授、及び多忙な中で講演いただいた星野主幹技師に感謝したい。

ピアレビュー有効時間比率計測による ピアレビュー会議の改善と品質改善の効果




久野 倫義[†1]　中島 毅[†1]　松下 誠[†2]　井上 克郎[†2]

本論文は，ピアレビュー会議の改善に関するものである．ピアレビュー会議はソフトウェア開発における品質向上の重要な活動であり，ピアレビューの中心的な活動である．ピアレビュー会議は品質向上の活動として定着しているが，ピアレビュー会議を完了したソフトウェアにおいても，欠陥が残存している場合も多い．そこでピアレビュー会議測定ツールを用いて，ピアレビュー会議の課題を定量的に明確化した．課題を解決するため，ピアレビュー有効時間比率という指標を用い，ピアレビュー会議が欠陥抽出を中心とした活動になるように改善を行った．その結果ピアレビュー有効時間比率の平均が 9.64%から 26.4%に向上し，単位時間当たりの指摘件数が 1.7 件から 2.1 件へ改善した．さらにテスト段階へ流出する欠陥数が減少し，製品品質を向上できた．

An Improvement of Peer–Review-meetings Using the Peer-Review-Effectiveness Ratio and Effect of Quality Improvement

Noriyoshi Kuno[†1], Tsuyoshi Nakajima[†1], Makoto Matsushita[†2], Katsuro Inoue[†2]

Abstract

In this paper, we show an improvement for peer-review-meetings which are center of the peer-reviews which have an important role for software quality improvement. In software developments, the peer-review-meetings are common activities. After peer-review-meetings, there are some bugs which should be removed in peers-review-meetings. So we made peer-review-meetings' problems clear quantitatively with a measurement tool. And we improved peer-review-meetings which focus on remove bugs by using a new metrics named peer-review-effectiveness ratio. As a result, peer-review-effectiveness ratio rise up from 9.64% to 26.4% and the number of bugs which removed in a peer-review meeting per unit time are increased from 1.7 to 2.1. Moreover, we show the number of bugs which removed in test is decreased.

1. はじめに

ソフトウェアの欠陥が引き起こすシステム障害の社会的な影響が増大する中，高品質なソフトウェアを開発するための検証手法及びそれらを使った品質管理方法を確立することが求められている．高品質なソフトウェアを最終検査工程だけで達成することは困難であり，開発各工程で確実に欠陥を除去していき後工程に流出させないことが必要である [Kan2002][中島 2008].

要求分析からコーディングに至る上流工程では，設計文書やプログラムを対象としたピアレビューが主たる検証手段である [Gilb1993] [織田 2006].

【脚注】

†1　三菱電機株式会社　設計システム技術センター
†2　大阪大学　大学院情報科学研究科

ピアレビューは人手により実施するため参加者個人の技量に大きく依存し，その効果にバラツキが現れやすい [森崎 2009]．検証手法として，このバラツキを軽減することを目的に，観点やチェックリストを用いる方法 [野中 2004] やプロセスを重視し組織力を活用する方法 [細川 2009] などが提案・評価されている．

Gilb はソフトウェアインスペクションを体系化し，その中心的な活動として欠陥抽出を行うピアレビュー会議（ソフトウェアインスペクションではロギングミーティングと呼ぶ）を定義した [Gilb1993]．

ピアレビュー会議に対する問題点として，ピアレビュー会議時間を短縮し，ピアレビュー会議におけるアイドル時間を削減する必要があることや，ピアレビュー会議がプロジェクト遅れを引き起こしており，ピアレビュー会議は不要であると主張している [Johnson1998] [Glass1999]．

一方，レビュー会議の質を上げるため，レビュー会議の参加人数の適正化とファシリテーションの有効性を評価する取り組み，ピアレビュー速度／指摘密度／レビュー効率という指標でピアレビューを分析する取り組みなどがある [小室 2005] [中野 2006]．

上記のいずれの研究においても，ピアレビュー会議において具体的にどのような活動を行っているかを定量的に評価していない．開発現場では，一言でピアレビュー会議と言っても千差万別であり，その中身を把握しなければ，その改善は困難である．そこでピアレビュー会議における読上げ，指摘，議論などの活動の内訳を定量化し，それらの活動の偏りを正し、企業活動で重要である，限られた時間内で効率と品質を改善することが可能となる．

本論文では，まずピアレビューの中心的活動であるピアレビュー会議を定量的に評価し，ピアレビュー会議の実施方法を改善することで，ピアレビュー会議をその目的である欠陥抽出活動に変更し製品品質を改善できることを示す．2 節において，ピアレビュー会議の問題点を明確化する．次に，3 節においては，レビュー会議を定量的に評価する手法を提案し，4 節では提案手法の実プロジェクトへの適用結果を示しその有効性を示す．5 節ではピアレビュー会議改善による製品品質向上の効果を示す．6 節では関連研究について述べ，従来研究と本提案技法との関係を明確にする．

2. 従来研究と解決すべき課題

2.1. ピアレビュー会議

ソフトウェアインスペクションは，図 1 に示すように，計画策定, キックオフ, 個人チェック, ピアレビュー会議（ソフトウェアインスペクションでは，ロギングミーティングと呼ぶ), 編集及びフォローアップで構成される [Gilb1993]．

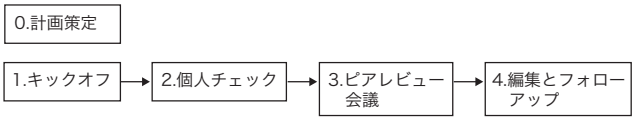


図 1　ソフトウェアインスペクションの流れ

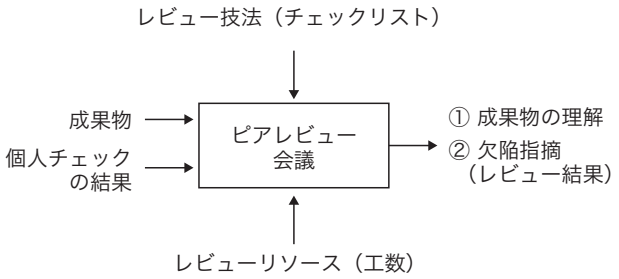


図 2　ピアレビュー会議プロセス

インスペクションは品質向上の重要な活動と位置づけられ，多くの研究対象となっている . その中でも，中心的役割を担うピアレビュー会議に対する従来研究を述べ，本研究で解決すべき課題を明確にする．

Gilb は，ピアレビュー会議の目的を個人チェックにおいて検出した欠陥を報告し，会議の場で新たに発見した欠陥と共に記録することであると定義した [Gilb1993]．その目的を達成する為，ピアレビュー会議においては，資料内容の読上げや修正の提案，個人で検出した欠陥に対する議論を行わず，欠陥の抽出に特化することで，ピアレビューを効果的な活動にできると報告している．図 2 は，ピアレビュー会議の課題を明確化する為，Gilb の報告をプロセスとして整理したものである．ピアレビュー会議では，成果物と個人チェックの結果をインプットとし，レビュー技法とリソースを使い，成果物を理解し指摘を行う．

2.2. ピアレビュー会議の課題

2.1 で示したようにピアレビュー会議は，欠陥の記録と会議で新たな欠陥を発見することが目的であり，その他の活動を行わないように制御する必要がある．ピアレビュー会議において，単に個人チェックにおいて検出された欠陥のみを報告し，会議時に追加欠陥が発見されなければ，ピアレビュー会議を行う必要はない．

Johnson は，インスペクションをソフトウェア品質改善のための，ただ 1 つの重要な手法と述べている．しかし，企業におけるソフトウェアインスペクションの採用率は低く，その原因の 1 つがピアレビュー会議に工数が多くかかること，アイドル時間があることを挙げている [Johnson1998]．

Glass は，複数人による個人チェックが品質向上のためには十分で有り，ピアレビュー会議は不要であると主張している．さらにピアレビュー会議開催がプロジェクトの進捗

を遅らせていると報告している [Glass1999].

上記報告ではピアレビュー会議は不要であると主張しているが，個人の視点で抽出した欠陥から他者が新たな欠陥を抽出できるピアレビュー会議は欠陥を流出させないという点で最も重要な活動であり，開発の現場では必ず実施すべき活動である．

2.3. 本研究が扱う課題

ソフトウェア開発の現場において，ピアレビュー会議が品質向上の中心的活動として定着してきている．ソフトウェア開発の各フェーズにおいてピアレビュー会議を行い，欠陥抽出を行っている [森崎 2009]．しかし，ピアレビュー会議を完了しても，ピアレビュー不足やピアレビューで検出すべき欠陥がテスト段階に流出する欠陥が残存する場合がある．

2.2 節で示した課題や上記の開発現場における課題など様々な問題点は報告されているが，種々な要因がからみあっており，ピアレビューの欠陥抽出件数／ピアレビュー全体時間などの評価指標では直接ピアレビュー会議の問題を把握することはできない．そこで実際にピアレビュー会議において何が行われているかを把握し，それをどのように改善し，結果をどのように評価するかを明確化することが必要である．

3. ピアレビュー会議の改善手法

前節で述べたようにピアレビューの問題点を把握する為，ピアレビュー会議で実際に何が行われているかを測定する手法を明確化する．

3.1. ピアレビュー会議の定義

一般には，ピアレビュー会議がどのように行われたかを測定することは難しい．レビュー会議時間であれば，会議開始時間と終了時間から算出できるが，レビュー会議では，読上げや単なる質問に要する時間もあり，単純な会議時間とレビューに要した時間は異なる． そこで，まず表 1 のようにピアレビュー会議で行われる活動を整理した。

表 1　ピアレビュー会議で行われる活動

発言内容分類	内容
開始宣言	目的や欠陥指摘件数目標の説明
内容読上げ	ピアレビュー対象の作業成果物の説明
指摘	作業成果物に対する欠陥指摘
議論	指摘に対する議論や成果物以外の議論
修正案	指摘に対する修正案の検討
意図の質問	指摘に対する，その意図の確認
無発言	発言のない時間

表 1 の定義は，Gilb がピアレビュー会議で想定した活動（「開始宣言」「指摘」「意図の質問」）と実際にソフトウェア開発の現場で行われると想定される発言から導いた．

測定者はピアレビュー会議に出席し，図 3 に示すピアレビュー会議測定ツールを用いて測定を行う．本ツールは，発言内容を測定するためのボタンと発言者ごとの発言時間を測定するボタンで構成される． 測定者がピアレビュー会議に出席し，出席者の発言内容を確認し表 1 の活動に該当するボタンを押下する．その際に発言者に該当するボタンも押下することでピアレビュー出席者ごとの発言時間を記録する．

3.2. 調査結果と課題

ピアレビュー会議測定ツールを用いて，12 種の製品を開発する 12 部門における 31 回のピアレビュー会議を定量的に測定した結果一覧を表 2 に示す．今回の調査対象は，ドキュメントに対するレビューを対象とし，コードレビューは対象としていない．

表 2 ピアレビュー会議時間測定結果

会議 No	開始宣言	内容読上	指摘 (*TF*※)	議論	修正案	意図の質問	無発言	部門名
1	1.4%	22.4%	6.7%	56.2%	7.0%	4.3%	2.1%	A
2	0.3%	15.4%	9.5%	46.1%	16.9%	11.6%	0.3%	A
3	0.5%	11.5%	5.6%	69.8%	7.3%	0.6%	4.7%	A
4	2.9%	25.6%	10.2%	36.4%	14.2%	1.3%	9.5%	A
5	0.0%	86.1%	8.3%	1.5%	0.0%	0.0%	4.1%	B
6	0.1%	75.2%	12.3%	10.3%	0.8%	0.0%	1.4%	B
7	1.0%	34.0%	5.0%	47.0%	0.0%	6.0%	7.0%	C
8	0.2%	22.5%	13.9%	43.5%	0.4%	0.0%	19.7%	C
9	0.4%	20.9%	25.4%	49.9%	0.2%	1.8%	1.4%	C
10	0.1%	20.6%	15.3%	21.7%	35.4%	4.0%	2.8%	C
11	0.1%	10.3%	15.4%	63.0%	6.2%	4.4%	0.5%	C
12	1.0%	74.0%	5.0%	13.0%	2.0%	0.0%	5.0%	D
13	1.0%	62.0%	7.0%	24.0%	4.0%	0.0%	2.0%	D
14	0.4%	61.9%	9.0%	11.7%	12.7%	0.4%	4.0%	E
15	0.5%	68.4%	2.4%	0.6%	0.0%	4.7%	23.4%	E
16	0.0%	14.0%	14.0%	48.0%	4.0%	0.0%	20.0%	F
17	2.0%	37.7%	17.7%	23.3%	3.9%	9.7%	5.8%	G
18	0.0%	38.2%	14.3%	31.6%	1.6%	2.6%	11.7%	G
19	0.0%	9.9%	27.8%	52.3%	2.3%	4.0%	3.6%	G
20	0.8%	40.7%	15.9%	37.4%	3.6%	1.0%	0.7%	H
21	0.1%	41.2%	7.8%	24.9%	1.8%	8.1%	16.0%	H
22	0.5%	23.2%	26.3%	37.9%	6.5%	2.2%	3.3%	H
23	2.3%	13.4%	22.9%	39.3%	8.5%	7.6%	6.2%	I
24	1.8%	35.3%	0.7%	39.2%	6.6%	11.5%	4.8%	J
25	4.4%	59.5%	1.1%	18.8%	2.8%	13.1%	0.4%	J
26	1.0%	8.3%	20.0%	47.4%	0.0%	22.3%	1.0%	J
27	0.8%	21.1%	8.2%	44.2%	0.1%	2.0%	23.7%	K
28	0.0%	25.4%	4.7%	39.4%	2.1%	2.6%	25.7%	K
29	0.2%	7.9%	8.3%	28.5%	6.6%	9.1%	39.5%	L
30	0.2%	55.0%	6.7%	18.3%	3.4%	7.6%	8.8%	L
31	0.1%	29.3%	5.8%	24.9%	11.8%	5.6%	22.5%	L

※ *TF* については，3.3 項に述べる．

発言内容　C の発言時間です。

B：開始宣言, C：読上げ, D：指摘, E：指摘に対する議論, F：修正案, G：意図の質問, H：無発言時間

開始宣言	内容読上げ	指摘	議論	修正案	意図の質問	無発言

発言内容のボタンを押すと、その前に押されたボタンの発言内容時間が終了します。

会議完了

発言者　A さんの発言中です。

A	B	C	D	E	F	G	H	I	J	K	L	M	N	発言なし

発言者のボタンを押すと、その前に押されたボタンの人の発言時間が終了します。

図３　ピアレビュー会議測定ツール

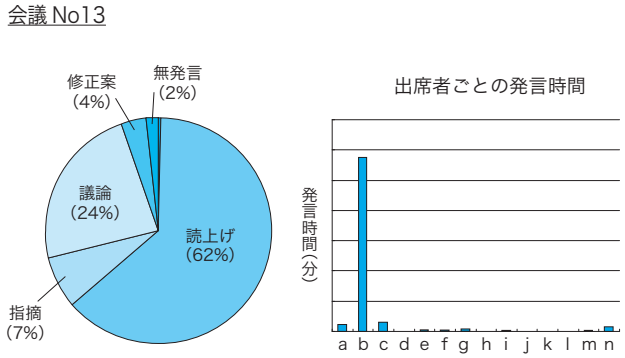


図４　仕様説明を中心としたピアレビュー会議結果

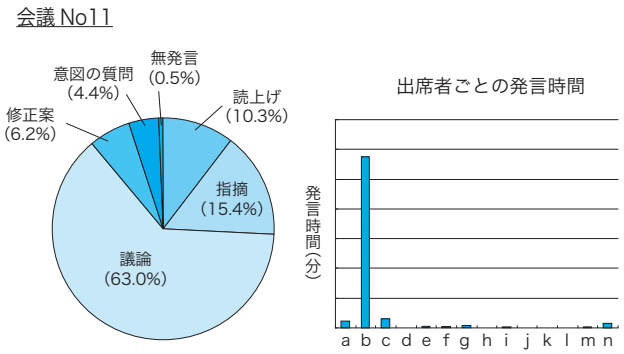


図５　設計活動を中心としたピアレビュー会議結果

表２より，部門ごとに差はあるが，以下の課題がある．

ｉ）ピアレビュー会議という同一の名称であっても内容は様々である．

ii）成果物を読上げる時間が 30% を超えるものが半数近くある．

iii）無発言時間が 10% を超えるものが３割ある．

iv）指摘時間が 10% 未満のものが半数を超える．

図４から図６は，測定結果をグラフ化したものであり，円グラフは表１の発言内容，棒グラフは出席者ごとの発言時間である．図４は，表２の会議 No.13 の測定結果である．図４の円グラフからはピアレビュー対象の作業成果物を説明する読上げ時間が，会議の 60%以上を占めていること，棒グラフからは発言者が１名に集中していることが分かる．それぞれのピアレビュー会議の出席者が 14 名であり，そのうち１名のみが資料を読上げていることから，本ピアレビュー会議が実質的には仕様説明会であることを示している．

図５は，表２の会議 No.11 の測定結果である．図５の円グラフからは，欠陥指摘に対する議論や修正案の検討時間が長く，欠陥指摘時間は 10%程度であり欠陥指摘活動というより設計自体を行っている．出席者の発言時間については，特定の出席者に偏っていないことから出席者の選定には問題がないことが判る．

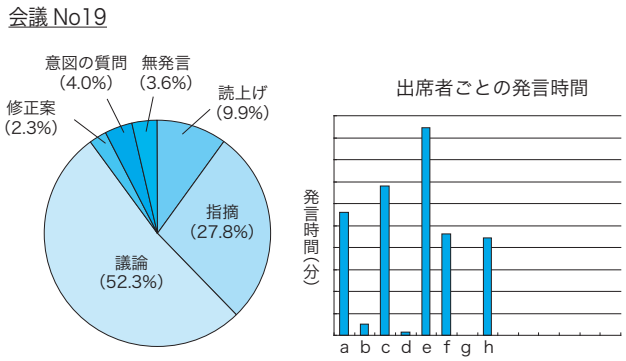


図６　欠陥抽出を中心としたピアレビュー会議結果

図６は，表２の会議 No.19 の測定結果である．図６の円グラフからは，議論の時間も長いが欠陥抽出時間比率が 25% を超えており欠陥抽出を中心としたピアレビュー会議であることが判る．出席者の発言時間については，特定の出席者に偏っていないことから出席者の選定には問題がないが，議論時間が長く欠陥抽出を十分できていない可能性がある．

3.3. ピアレビュー会議プロセス定義と有効指摘率を用いたピアレビュー会議改善

3.2 項で示したように，本来は欠陥抽出を意図しているピアレビュー会議が説明会や設計活動になっており，ピアレビュー会議を本来の欠陥抽出に特化した活動とするために

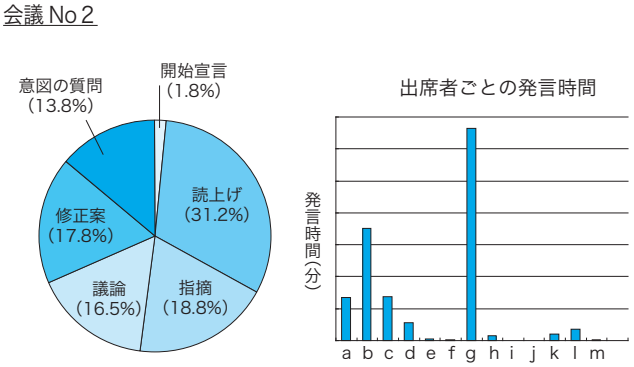


図 7　部門 A のピアレビュー会議測定結果（改善後）

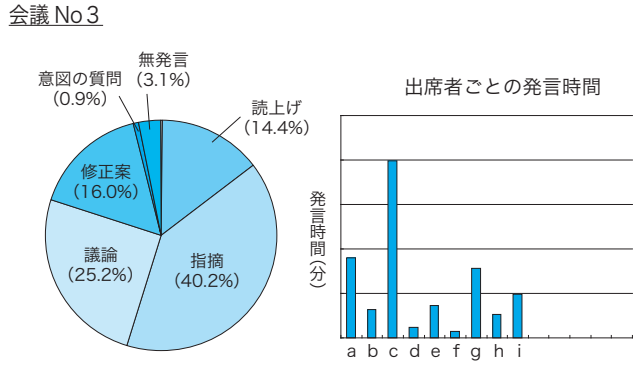


図 8　部門 H のピアレビュー会議測定結果（改善後）

は，内容の読上げ，修正案の検討，無発言を少なくする必要がある．まず，図 4 などの定量的に把握したピアレビュー会議の現状を開発者に説明を実施し，目指すべき欠陥抽出を中心とした活動とすべきことを合意した．その後，Gilb が定めたピアレビュー会議プロセスをベースにソフトウェア開発の現場に適合するピアレビュー会議プロセスを以下の通り定義した．

(1) モデレータ，書記，読上げ者（できれば作成者以外）を決める．

(2) レビュー会議の目的を文書化し，全員が見える場所に示す．

(3) 目的に合致した参加者を決定する．

(4) 事前査読時間と指摘件数を全員が報告し，事前査読が不十分なら延期を検討する．

(5) 会議内容を記録（指摘内容だけではなく，発言で気になる点を記録する）．

(6) 全員が目的を意識し，不要な議論（前提を基にした議論等）を排除する．

(7) 全員が指摘をするように順番に指摘を促す．

(8) モデレータは，発言内容，動作を観察し，納得していないようなら発言を促す．以下に注意する．

① 資料自体の読上げをしていないか．

② 指摘に対しその場で回答をしようとして議論になっていないか．

③「以前に説明したように」という発言がある場合，今までも口頭で説明し記録されていないと判断し，発言を文書化するように促す．

④ 必要であれば議論の内容を仕様書に記載することを促す．

⑤ 用語の解釈が出席者間で異なっていないか．

(9) 目的が達成できたかを，ピアレビュー会議終了時に確認する．

(10) 指摘をピアレビュー会議後に確認し，記録誤り，抜けがないことを確認する．

上記で定義したプロセスに基づくピアレビュー会議を実施し，ピアレビュー会議における発言内容ごとの時間を測定した．ピアレビュー会議が欠陥抽出を中心とした活動となることを推進する為，ピアレビュー有効時間比率（以降 *TF* と記述）と呼ぶ指標を導入した．*TF* は，ピアレビュー会議測定ツールの「指摘ボタン」を押していた時間の割合であり，以下の式で表わされる．

$$TF = \Sigma\ (Ti) / T \qquad (式\ 1)$$

ここで，

Ti: レビュー会議参加者 i が指摘を行った時間

i: レビュー会議参加者，T : 総レビュー会議時間

式 1 で示すように，*TF* は，各レビュー参加者がレビュー会議中に指摘を行った時間の総和を総レビュー時間で割ったものである．*TF* を高めることで，ピアレビュー会議を欠陥抽出中心とした活動とすることができる．なお，今回測定したピアレビュー会議時間は，過去のピアレビュー会議のデータから 1 頁にかけた平均時間により決定しているため，*TF* を向上することは，目標ピアレビュー会議時間内で欠陥抽出の効果を最大化することである．ただし，目標ピアレビュー会議時間の設定がピアレビューの効果に影響を与える可能性があり，本来は流出欠陥が基準を越えたケースをはずれ値として除き，統計的に目標を決定することが望ましい．

4. 適用と評価

4.1. 適用

3.3 節で示した改善プロセスを適用し，その適用結果を測定できた 4 部門における 8 回の会議結果を表 3 に示す．表 3 から改善前のように *TF* が 1 桁のピアレビュー会議はなくなっておりピアレビューが欠陥抽出活動に変化したことが分かる．

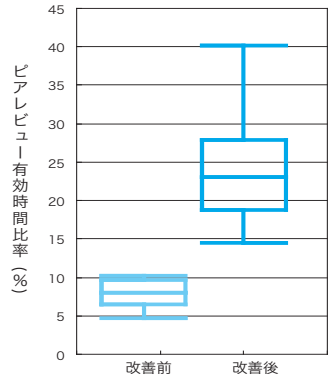


図 9　改善前後の *TF* の箱ひげ

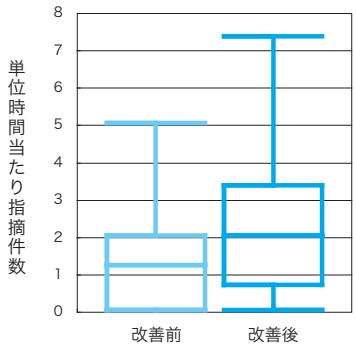


図 10　改善前後の単位時間単位指摘件数の箱ひげ

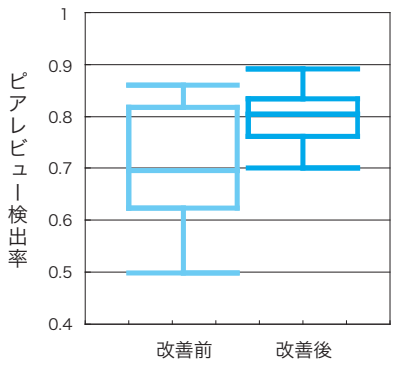


図 11　全検出欠陥に占めるレビュー指摘欠陥割合（式 2）の変化

表 3　ピアレビュー会議測定結果（プロセス定義後）

会議 No	開始宣言	内容読上	指摘（*TF*）	議論	修正案	意図の質問	無発言	部門名
1	1.5%	53.6%	23.5%	9.6%	6.9%	1.5%	3.4%	A
2	1.8%	31.2%	18.8%	16.5%	17.8%	13.8%	0.0%	A
3	0.2%	14.4%	40.2%	25.2%	16.0%	0.9%	3.1%	H
4	0.1%	9.9%	22.7%	47.1%	4.1%	3.4%	12.8%	H
5	0.8%	27.8%	18.5%	45.0%	4.1%	0.0%	3.8%	H
6	1.0%	2.0%	49.0%	35.0%	4.0%	7.0%	2.0%	K
7	1.0%	12.3%	23.7%	29.0%	17.9%	3.2%	12.8%	L
8	0.6%	40.4%	14.5%	15.5%	12.1%	6.0%	10.9%	L

改善後の測定結果において，*TF* が比較的低い会議と *TF* が比較的高い会議の測定結果を示す．図 7 は，表 3 の会議 No.2 の測定結果である．*TF* は 18.8% であり，改善前に比べ向上したが，読上げ，議論，修正案，意図の質問に対する時間比率も多く，13 名の参加者を集めた説明会，欠陥抽出活動，設計の混在した会議であった．

図 8 は，表 3 の会議 No.3 の測定結果である．9 名の参加者でピアレビュー会議を実施し，同一製品における改善前の比率と比較して，指摘時間比率の向上，読上げ時間比率の減少など改善効果があった．

これらの結果から，プロセス定義に基づきピアレビュー会議を実施し，その結果を見える化し開発者と課題を共有することで，ピアレビューを Gilb の示したピアレビュー会議のあるべき姿に近づけることができると言える．

4.2. 評価結果

4.1 項の適用結果からピアレビュー会議における *TF* は平均 9.64% から 26.36% に改善した．本項では，改善前と改善後の *TF* の変化について評価し，その有効性を判定する．まず改善前後の 2 群の分散が同一であることを帰無仮説として分散比を検定した．表 4 から P 値は 0.022 であり，有意水準 5% で分散が同一であるという仮説は却下された．そこで，分散が異なるとして，2 つの平均に差がないことを帰無仮説として検定を行った．表 5 から P 値が 0.003 と 0.005 であり，また t 境界値よりも算出された t の絶対値が大きく，有意水準 5% で平均に差がないという仮説は却下され，平均値は統計的に有意な差を持つと結論できた．

表 4　改善前後の分散比の検定

	平均	分散	観測数	自由度	観測された分散比	P(F<=f) 片側	F 境界値 片側
改善前	9.64	36.1	12	11	3.927	0.022	3.012
改善後	26.36	141.77	8	7			

表 5　改善前後の平均値の差の検定

	平均	分散	観測数	自由度	t	P(T<=t) 片側	t 境界値 片側	P(T<=t) 両側	t 境界値 両側
改善前	9.64	36.1	12	9	-3.673	0.003	1.833	0.005	2.262
改善後	26.36	141.77	8						

図 9 は，改善前と改善後の *TF* の測定結果を箱ひげ図を用いて示したものであり，改善前後の *TF* に差異があることが判る．なお，ひげの両端は最大最小値を示している．

5. レビュー会議改善による品質改善に対する評価

5.1. 単位時間当たりの指摘数の評価

これまで示した改善により，ピアレビュー会議を欠陥抽出を中心とした会議に変更できたことを示した．さらに，ピアレビュー会議における単位時間当たりの指摘件数は，平均 1.7 件から 2.1 件に向上した．この違いを評価する為，まず 2 つの分散が同一であることを帰無仮説として分散比を検定した．表 6 は，改善前後の単位時間当たりの指摘件数の分散比の検定結果である．表 6 から，P 値は 0.0001 となり有意水準 5% で分散が同一であるという仮説は却下された．そこで，分散が異なるとして，2 つの平均に差がないことを帰無仮説として検定を行った．表 7 から P 値が 0.00026 と 0.0005 であり，有意水準 5% で平均に差がないという仮説は却下され，改善前後の平均値が統計的に有意な差を持つと結論できた．なお，指摘件数に関しては欠陥

種別により分類し，上位仕様書に要求事項として記載されていない改善事項（例えば保守性に関する指摘）や体裁に関する欠陥などはカウントしていない．

表 6　改善前後の単位時間当たりの指摘件数分散比の検定

	平均	分散	観測数	自由度	観測された分散比	P(F<=f) 片側	F 境界値 片側
改善前	1.666	2.419	213	212	1.548	0.0001	1.213
改善後	2.136	3.744	557	556			

表 7　改善前後の単位時間当たりの指摘件数平均値の差の検定

	平均	分散	観測数	自由度	t	P(T<=t) 片側	t 境界値 片側	P(T<=t) 両側
改善前	1.666	2.419	213	474	3.496	0.00026	1.648075	0.000517
改善後	2.136	3.744	557					

図 10 は，改善前と改善後の単位時間当たりの指摘件数の測定結果を箱ひげ図を用いて示したものであり，改善前後の単位時間当たりの指摘件数に差異があることが判る．

これまで説明してきたように , ピアレビュー会議プロセスを測定 , プロセスの問題を定量的に把握し , 目標となる指標を決定した上で改善活動とその効果を測定することでプロセスの問題を解決できることを示した．

開発の現場においては，ピアレビュー工数とピアレビュー指摘件数に目標値を設けピアレビュープロセスの統一を図っている．今回の改善により，目標ピアレビュー工数におけるピアレビュー指摘件数を向上することができ，テストフェーズに流出する欠陥数を抑制することができた．

5.2. ピアレビュー改善による製品品質向上の評価

ソフトウェア開発において，上流工程で品質を作り込むことが重要である．そのためには，設計書様式を決定し欠陥混入を防止すると共に，ピアレビューにより欠陥を検出する．さらにピアレビューで検出できない欠陥をテストで検出する．しかしテスト段階で全ての欠陥を検出することは困難であり，設計段階で欠陥混入を防止することが製品品質を向上する方法である．図 11 は，ピアレビュー改善を実施した複数のプロジェクトにおけるソフトウェアライフサイクル全体で検出する欠陥数のうち，ピアレビューの欠陥数の比率を示したものである．

ピアレビュー検出率 ＝
　ピアレビュー検出数／ライフサイクルの欠陥数（式 2）

改善前後において，ピアレビューで検出した欠陥数が増加しており，テストで検出する欠陥比率が減少していることが判る．

上記改善前後のピアレビュー検出率についても，検定を行った．まず 2 群の分散が同一であることを帰無仮説として分散比を検定した．表 8 から P 値は 0.032 であり，有意水準 5% で分散が同一であるという仮説は却下された．そこで，分散が異なるとして，2 つの平均に差がないことを帰無仮説として検定を行った．表 9 から P 値が 0.012 と 0.025 であり，有意水準 5% で平均に差がないという仮説は却下され，改善前後のピアレビュー検出率の平均値は統計的に有意な差を持つと結論できた．

表 8　改善前後のピアレビュー検出率の分散比の検定

	平均	分散	観測数	自由度	観測された分散比	P(F<=f) 片側	F 境界値 片側
改善前	0.703	0.0141	14	13	2.911	0.032	2.577
改善後	0.791	0.0048	14	13			

表 9　改善前後のピアレビュー検出率平均値の差の検定

	平均	分散	観測数	自由度	t	P(T<=t) 片側	t 境界値 片側	P(T<=t) 両側
改善前	0.703	0.0141	14	21	-2.41	0.01272	1.720743	0.02543
改善後	0.791	0.0048	14					

5.3. ピアレビュー参加者による主観的評価及び改善活動の展開

ピアレビュー改善を実施した開発現場から，以下の意見がありピアレビュー改善が有効に機能したと評価できた．

(1)　ピアレビューの問題点が定量的にはっきりし改善が進んだ．

(2)　ピアレビューを実施することが目的となりがちであったが，ピアレビューが本来の目的である欠陥抽出に有効な活動となった．

一方，以下のような課題点も上げられた．

(3)　定着化が課題である．改善に関わったメンバ以外を巻き込むには教育が必要．

(4)　なぜ，議論をしてはいけないのか．有識者が集まれる時間は少ない．

上記のような課題に対しては，開発現場の実態を考慮し 3 段階で改善を進める方法や欠陥抽出と修正案検討の場を分離するなどの提案を行った [久野 2009]．その結果をガイドラインとしてまとめ , ソフトウェア開発のベストプラクティスとして全社ソフトウェア開発を行う事業所から利用できるようにした．またソフトウェア開発のプロジェクトリーダに対する教育において今回の改善の講義を行い，ピアレビュー改善を全社的に展開している．

6. 関連研究

ピアレビューの結果から品質を評価するために定量データを用いて評価する手法として，以下のような手法が提案されている．

中野らは，500 プロジェクトのコードレビューデータから，レビュー効率（ライン数／レビュー工数）に対してレビュー指摘密度（指摘数／ライン数）が大きい場合は，テスト段階で欠陥が多く検出される傾向を示した [13]．しかし，レビュープロセスについては，全プロジェクトで統一されており，プロジェクトごとにプロセスの変化がないことを前提としており，具体的にレビュー会議においてどのような活動を実施したかは把握していない．

「定量的品質予測のススメ」では，レビュー工数を式 3 で定義した [SEC2008]．

レビュー工数　＝
Σ　各レビューアのレビュー実施時間　　（式 3）

その際に，有識者以外（育成等を目的とした要員）のレビューアの工数は，レビュー工数から除外するなど，有識者以外のレビュー参加者の工数を適切な係数で補正することが望ましいとあるが，具体的な係数については言及されていない．またレビュープロセスの評価フローとして，レビュー工数密度の評価と適切さを評価することになっているが，無発言時間などを含むレビュー工数全体を用いて評価するため，その適切さ自体に誤りが入る可能性がある．

野中はインスペクションの定量的管理に用いる指標として欠陥指摘工数密度を定義した [野中 2009]．

欠陥指摘工数密度 ＝　Σ NFi ／　Σ Ti　　（式 4）
NF i は欠陥数，T i: は総レビュー時間であり，
i: はレビュー参加者を示す．

式 4 における各値の測定方法が一貫していること，すなわち開発プロセスが標準化され安定していることが必要であるとしている．

効率的なレビューやレビュープロセス改善手段としては，レビュー会議では欠陥の抽出に集中することや，ピアレビュー会議の工数をレビュープロセスの評価メトリクスとして用いることを提案している [飯山 2008] [安達 2006]．

また Robbins は , レビューアがドキュメントやレビュー用シナリオを読む時間などの相対時間を分析し , ドキュメントを読む時間が全体の 29.64% であることを報告しているが，29.64％であることの有効性や課題については言及していない [Robbins2009]．

上記の各研究においては，各種指標の測定方法を一貫性のあるものにすることや，開発プロセスを標準化することの重要性は述べているが，具体的な方法については示されていない．本文で示した方法により，ピアレビュー会議を欠陥抽出中心の活動とすることで，ピアレビュー工数の精度を向上でき，ピアレビュー評価手法を改善できると考える．

7. おわりに

本論文では，ピアレビューの中心的活動であるピアレビュー会議を定量的に評価し，ピアレビュー会議の実施方法を改善することで，ピアレビュー会議をその目的である欠陥抽出活動に改善できることを示した．

実際にピアレビュー会議の改善を行うのは，そのプロセスを定義するだけではなく，実際にどのような活動を行っているかを定量的に測定し，開発者へフィードバックし，制御することが重要であることが分かった．さらにピアレビュー会議が欠陥抽出活動ではなく単なる説明会であったり，設計活動であったりした場合，ピアレビュー会議時間をピアレビュー工数として品質評価や品質予測を適切に行うことはできないことを示した．

今後は，ピアレビュー有効時間比率とテスト段階における欠陥数のデータを蓄積し，両値の相関を分析することで，ピアレビュー有効時間比率を品質判断の基準値として用いる上での適値範囲を決定していく予定である．また今回の改善はピアレビュー会議に焦点を絞った活動であるが，本手法は様々な会議の改善に用いることが可能である．

【参考文献】

[Kan2002] S. H. Kan: Metrics and Models in Software Quality Engineering, Addison-Wesley, 2002.

[中島 2008] 中島毅，東基衛 : ソフトウェア開発における品質プロセスのコスト最適化のためのモデルとシミュレーションツール , 電子情報通信学会論文誌 , Vol.J91-D, No.5, pp.1216-1230, 2008.

[織田 2006] 織田巌 : ソフトウェア・レビュー技術，ソフトウェア・リサーチ・センター , 2006.

[Gilb1993] T. Gilb and D. Graham: Software Inspection, Addison-Wesley, 1993.

[森崎 2009] 森崎修司 : ソフトウェアインスペクションの動向 , 情報処理 , Vol.50, No.5, pp.377-380, 2009.

[野中 2004] 野中誠 : 設計・ソースコードを対象とした個人レビュー手法の比較実験，情報処理学会研究報告 , SE-140-4, Vol.2004, No.118, pp.25-31, 2004.

[細川 2009] 細川宣啓 : 第三者インスペクションによる品質検査と欠陥予測 , 情報処理 , Vol.50, No.5, pp.405-411, 2009.

[Johnson1998] P. M. Johnson: Reengineering Inspection, Communications of the ACM, Vol.41, No.2, pp.49-52, 1998.

[Glass1999] R. L. Glass: Inspection – Some Surprising Findings, Communications of the ACM, Vol.42, No.4, pp.17-19, 1999.

[小室 2005] 小室睦他 : 開発現場の実態に基づいたピアレビュー手法の改善と改善効果の定量的分析 , SEC journal, Vol.1, No.4, pp.6-15, 2005.

[中野 2006] 中野裕也 , 水野修 , 菊野亨 , 阿南佳之 , 田中又治 : コードレビューの密度と効率がコード品質に与える影響 , SEC journal, Vol.2, No.4, pp.10-17, 2006.

[久野 2009] 久野倫義 , 丹羽友光 , 前川隆昭 : デザインレビューの効果的実施及び評価方法 , 三菱電機技法 , Vol.83, No.5, pp.18, 2009.

[SEC2008] 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 : 定量的品質予測のススメ , オーム社 , 2008.

[野中 2009] 野中誠 : ソフトウェアインスペクションの効果と効率 , 情報処理 , Vol.50, No.5, pp.385-390, 2009.

[飯山 2008] 飯山俊介 : 設計レビュー指標値の算出 , 第 28 回ソフトウェア品質シンポジウム , 2008.

[安達 2006] 安達賢二 : レビュープロセスの現実的な改善手段の提案 , ソフトウェアテストシンポジウム , 2006.

[Robbins2009] B. Robbins: Cognitive Factors in Perspective-Based Reading (PBR): A Protocol Analysis Study, Third International Symposium on Empirical Software Engineering and Measurement,pp.145-155, 2009.

ソフトウェア品質の第三者評価における探索的データ解析ツールの利用とその効果：OSSデータを対象とした検証実験


大平 雅雄[†1]


伊原 彰紀[†2]


中野 大輔[†2]


松本 健一[†2]

本論文では，ソフトウェア品質の第三者評価における探索的データ解析ツールの利用とその効果について報告する．探索的データ解析は，明確な分析目的を持たない状態で，与えられたデータに潜むモデルや傾向を多角的に分析する手法である．ソフトウェア品質の評価をおこなうためには，まず，ソフトウェアとその品質が実現される過程を理解するところから始めなければならない場合も多い．データの特徴，傾向，規則性，特異点などを「仮説」として生成し，ソフトウェア品質との因果関係を明らかにしていく過程は，探索的データ解析そのものと言える．OSS プロジェクトデータ（Eclipse Platform プロジェクトの課題データ）を対象に，探索的データ解析ツール HCE (Hierarchical Clustering Explore) [Seo2002] を用いた仮説生成実験をおこなった結果，ドメイン知識のない被験者であっても，ソフトウェア工学の研究分野で検証済みの重要な法則につながる仮説を生成できることが分かった．

Exploratory Data Analysis in Independent Assessment of Software Quality: A Case Study of OSS Project Data

Masao Ohira[†1], Akinori Ihara[†2], Daisuke Nakano[†2], Kenichi Matsumoto[†2]

Abstract

This paper reports the use of exploratory data analysis and its effect in independent assessment of software quality. Exploratory data analysis (EDA) is a method to analyze models or trends hidden in datasets from various perspectives while there is no explicit goal for the analysis. Assessing software quality often requires an understanding of the process of developing software and achieving the quality. In a sense, the early process of the analysis in independent assessment of software quality can be regarded as the process of EDA, since without any specific goals, extracting characteristics, trends, regularity, singularity and so forth from datasets need to uncover causal relationships hidden in the datasets. We have conducted a case study of an EDA tool called HCE (Hierarchical Clustering Explore) [Seo2002] with issue dataset collected from the Eclipse Platform project. As a result, we found that the tool can support subjects to create hypotheses which would result in finding important rules verified in the literature of software engineering, even if our subjects do not have sufficient knowledge on the target domain.

【脚注】

† 1　和歌山大学 Wakayama University

† 2　奈良先端科学技術大学院大学 Nara Institute of Science and Technology

1. はじめに

現在，利用者からは中身が見えないソフトウェアの安全性，信頼性にかかわる品質について，専門的知見に基づいた中立的な第三者の立場で評価し，専門知識を持たない利用者にも理解できる形で提示するための仕組み「ソフトウェア品質監査制度」の整備が急務となっている [IPA]．ソフトウェアシステムの高度化・複雑化とともに，ソフトウェア品質を担保するための考え方が国際的に変化してきており，我が国の産業界の品質説明力を強化する必要性に迫られているためである．

このような背景を受け，我々の研究グループでは，ソフトウェアやその品質が実現される過程を様々な観点から解析・可視化するための表現手法「ソフトウェアプロジェクトトモグラフィ (Software Project Tomography)」を構築し，ソフトウェア品質の第三者評価を支援するための要素技術を構築した [IPA2012]．ソフトウェアトモグラフィとは，医療におけるコンピュータ断層撮影，所謂，CT (Computed Tomography) から着想を得た手法である．ソフトウェア開発プロジェクトを体に見立て，プロジェクトの開始から終了までの幾つかの時点において，プロジェクトの状態を定量的に表すスナップショットとして，プロジェクトを断面画像のように様々な視点で可視化し，その品質が実現される過程を表現することを目的としている．

本論文では，構築した要素技術の内，「仮説の生成・検証」を支援するための探索型解析技術の効果について，既存の探索的データ解析ツール HCE (Hierarchical Clustering Explore) [Seo2002] を用いた被験者実験の結果をもとに報告する．

2. 探索的データ解析

2.1. 第三者評価における探索的データ解析

探索的データ解析は，明確な分析目的を持たない状態で，与えられたデータに潜むモデルや傾向を多角的に分析する手法である [Tukey]．ソフトウェア品質の第三者評価，特にその初期段階では，探索的データ解析が必要になると考えられる．ソフトウェアの品質を評価する第三者は，対象プロジェクトの全容についてあらかじめ詳細に把握している訳ではないため，ソフトウェアやその品質が実現される過程を理解するところから始めなければならない．

そのため，まず，構成管理システム，工程管理システム，不具合管理システム等に蓄積されたソフトウェアプロジェクトのデータから，その特徴，傾向，規則性，特異点などを「仮説」として生成し，ソフトウェア品質との因果関係を明らかにしていく必要がある．さらに，生成した仮説の真偽や適用範囲の確認をおこなうことが，客観的にソフトウェア品質を検証するための「検証型解析」における解釈モデルや判断基準，分析モデル等の構築に必要となる．

2.2. 第三者評価のスコープと探索型解析技術

ソフトウェア品質の第三者評価における探索型解析技術の主たる目的は，文献 [IPA] に示されている第三者評価（品質監査）に求められる 4 つのスコープの内，①プロセス実施の妥当性（プロセスが規定通りに実施されているか，プロセスを構成するタスクやアクティビティが確実に実施されているか）と，②採用規格・技術の妥当性（採用した規格や技術は適切か，採用した技術は適切に利用されているか）を，検証型解析技術を用いて客観的に検証するのに必要なモデルや基準を提供することである．

一般的に，探索的データ解析では，散布図やヒストグラム，箱髭図等の簡易な統計ツールが利用される．しかし，ソフトウェアプロジェクトデータ（あるいは，ソフトウェアプロジェクトトモグラフィにおいて利用されるスナップショットデータ）は，基本的に多変量・多次元データから構成される．データの特徴や傾向を把握するために任意の 2 変数間の関係を調べる，といった作業だけでも多大な労力が必要となることが多いため，簡易な統計ツールのみでは不十分な場合が多い．

近年では，多変量・多次元データに対する探索的データ分析を支援するソフトウェアが多数登場しており，その多くはデータや分析結果をグラフィカルに表示し，ユーザがデータを探索（試行錯誤）しながらデータを理解したりモデルを構築したりするのを助ける．本研究では，ソフトウェア品質を評価する第三者は，必ずしも

- 対象プロジェクトの全容をあらかじめ詳細に把握している訳ではない
- 多変量・多次元データに対する統計手法に精通している訳ではない

ことを前提（ペルソナ）として，第三者が効率的に探索的データ解析を行えるようにするための探索型解析技術の選定をおこない，rank-by-feature framework [Seo205] に基づく探索型可視化方式を採用した．また，rank-by-feature framework を実装した探索的データ解析ツール HCE (Hierarchical Clustering Explore)[Seo2002]

[Seo2007] を用いた被験者実験をおこない，第三者評価における探索型解析技術の有用性を検証した．

3. 探索的データ解析ツール

3.1. 技術選定方針

探索型解析技術として，rank-by-feature framework にもとづく探索型可視化方式を採用するに至った技術選定方針を以下に述べる．

前述したように，本研究では，ソフトウェア品質を評価する第三者は，対象プロジェクトの詳細及び統計手法に十分な知識がない場合があることを想定している．加えて，ソフトウェアプロジェクトのデータ（本研究におけるスナップショットデータ）は，多次元・多変量データの集合体であり，情報可視化における従来の原則「Overview first, zoom and filter, then details on demand」は必ずしもあてはまらないことに着目した．そこで本研究では，多数の変数間の関係を網羅的に表し，かつ，第三者（分析者）の認知的負荷の軽減にもつながる，できるだけシンプルな提示方法が必要なことから，Seo らが提案するフレームワーク rank-by-feature framework[Seo2005] が有用であると考えた．

Rank-by-feature framework では，探索型可視化を行うユーザが，多変量・多次元データから興味深い仮説を導き出せるよう，1 次元（ヒストグラムや箱髭図等），あるいは，2 次元（散布図等）でデータが表現される．また，複数の視点からデータの全体像を素早く把握し，系統立った可視化が可能となるよう，データへの順位付け (rank) が行われる．順位付けには，データ分布の正規性や均一性，外れ値や特異値など，データの特徴 (feature) を表す統計量が利用される．その一方で，3 次元表現など，ユーザの認知的負荷が大きく，解釈やデータ操作がしばしば困難となる表現形式は使用されない．

本研究では，スナップショットの探索型可視化において，ソフトウェアプロジェクトトモグラフィを構成する 5 つの観点（要求，作業，組織，プロダクト，課題）を特徴（feature）と考え，それらを表す統計量によってスナップショットの構成要素やその解析結果の順位付けを行う．例えば，「プロダクト」の観点であれば，モジュールの複雑さ，変更量，クローン値などにもとづいてスナップショットの構成要素やその解析結果を順位付けすることになる．特徴として用いる 5 つの観点は，ソフトウェア開発者・管理者にとってなじみ深いものである．従って，可視化結果の直感的な理解や解釈を可能にし，仮説の真偽や適用範囲の確認も容易になると考えた．

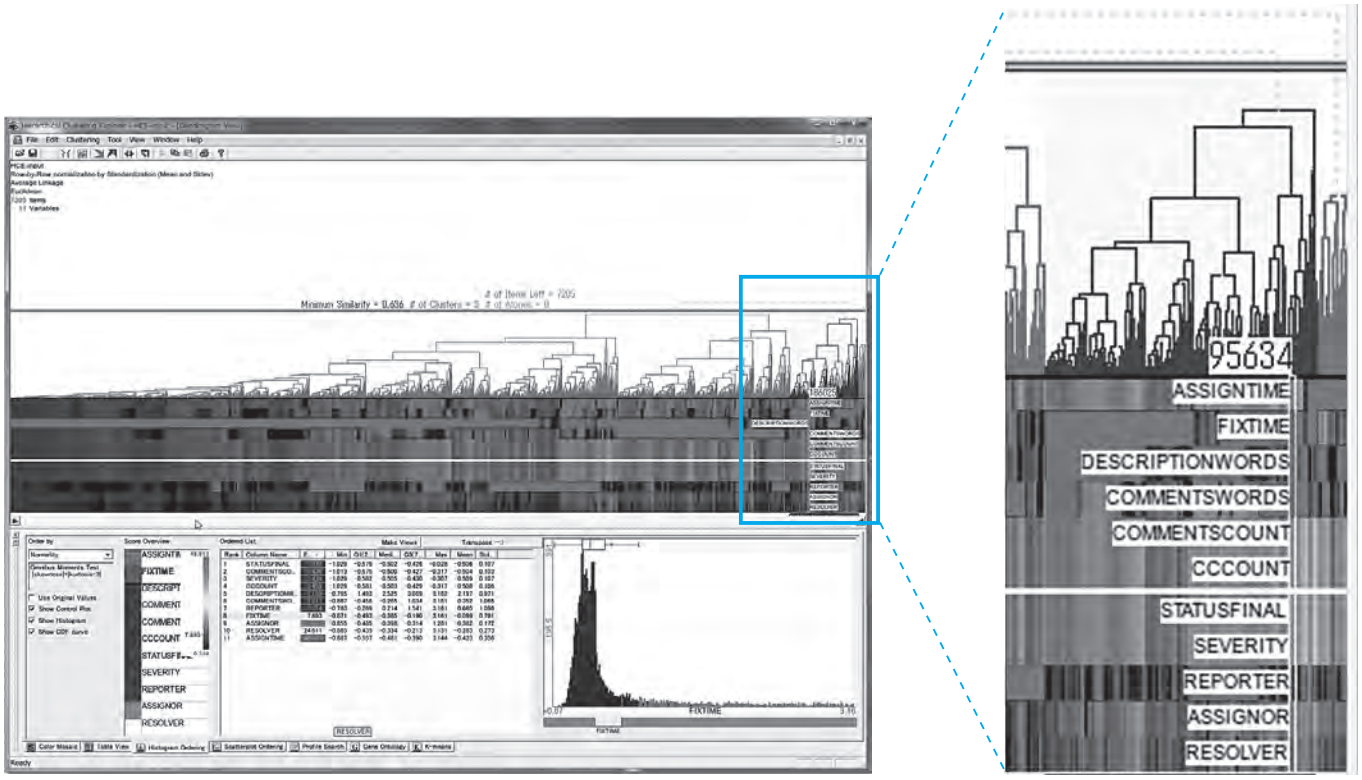
3.2. HCE (Hierarchical Clustering Explore)

本研究では，rank-by-feature framework[Seo2002] を実装した探索的データ解析ツール HCE (Hierarchical Clustering Explore)[Seo2002][Seo2007] を用いた被験者実験をおこない，第三者評価における探索型解析技術の有用性を検証する．以下では，HCE ツール（図 1）の概要と利用用法について述べる．

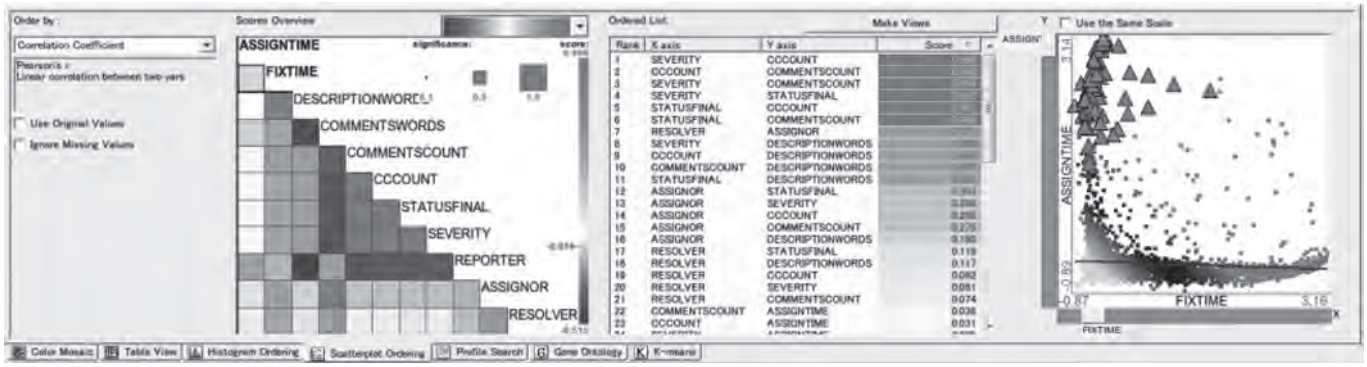
HCE ツールは，入力されたスナップショットデータから，類似するデータ同士をグルーピングする階層的クラスタリング機能（図 1-(a) の画面上部），クラスタ内のデータの分布を把握するヒストグラム作成機能（図 1-(a) の画面下部 , Histogram Ordering），分析者が注目したクラスタの任意の 2 変数間の関係を把握するための散布図作成機能（図 1-(b), Scatterplot Ordering），クラスタ内のすべての変数間の関係を把握するための平行座標プロット機能，クラスタ内のデータの分布を把握するヒストグラム作成機能（図 1-(c), Profile Search）などから構成されている．これら豊富な探索型可視化分析機能により，HCE ツールは，生物学分野における遺伝子解析やプロテオーム解析（タンパク質の機能・構造の大規模解析）など [Cluzet][Tsai] で数多くの利用実績がある．

分析者はまず，階層的クラスタリング機能を用いていくつのクラスタを作成するかを指定する．分析対象データは通常，多変数データであるが，変数間の類似性が階層図の下側に色の濃淡で表現されている（図 1-(a) の画面上部）ため，容易にデータのまとまりを識別できるようになっている．注目するクラスタを選択（図 1-(a) の画面上部の 3 つのクラスタの内，真ん中のクラスタ）すると，選択したクラスタに対応するデータの散布図やヒストグラムが図 1-(a) の画面下部に自動的に表示されるため，一画面の中で探索的にデータを分析することができる．特に，相関関係の強い（または弱い）2 変数をランク付けする（例えば，図 1-(b) の色の濃淡による相関係数の表示方法）ことで，分析対象に精通していない，あるいは，分析のエキスパートでない分析者にも重点的に分析すべき対象が容易に選択できるようにしている点に大きな特徴がある．

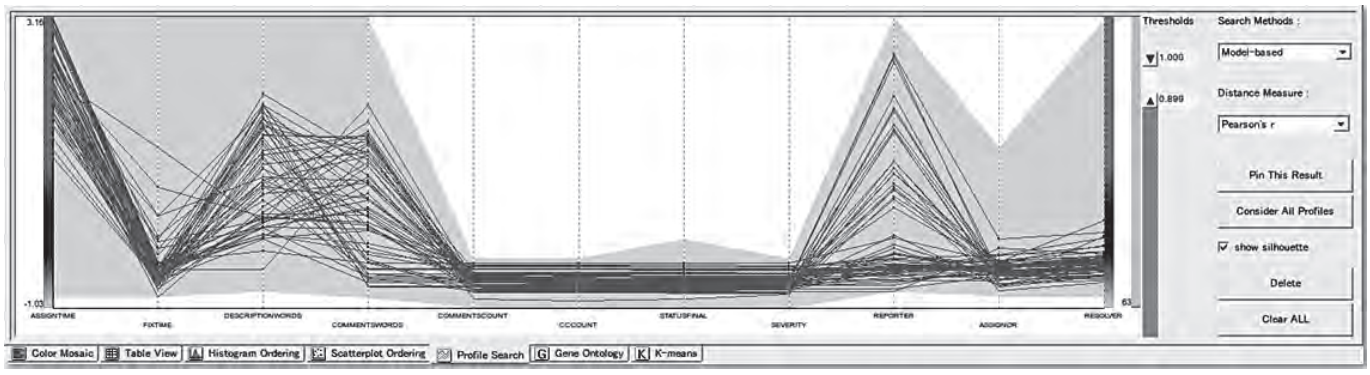
また，HCE ツールでは，ユーザがデータセットを準備するのを助けるために，タブ区切り，コンマ区切りのテキストファイルの他，MS Excel ファイルを入力ファイル



(a) ツールの概観と選択中のクラスタ（拡大図）　（画面下部は Histogram Ordering タブ選択時の表示例）



(b) Scatterplot Ordering タブ



(c) Profile Search タブ

図 1　HCE ツール [Seo2002][Seo2005]

として利用できる．一行目に変数名，二行目にデータの型を記述し，三行目以降にデータを配置した入力ファイルを作成すれば分析を始めることができるため，スナップショットデータとの親和性も非常に高い．

4. 検証実験

4.1. 実験概要と目的

探索型解析技術として，rank-by-feature framework にもとづく探索型可視化方式用いることが，ソフトウェア品質の第三者評価，特にその初期段階における探索的データ解析に効果があるかどうかを調べるための検証実験をおこなった．

検証実験では，2 種類の実験をおこなった．まず，OSS 開発を熟知しているグループとそうでないグループの 2 群に分け，Eclipse Platform プロジェクトの課題（不具合）データを HCE ツールで分析させた（**検証実験 I**）．さらに，知識の有無だけではなく，MS Excel 等の一般的な表計算アプリケーションを用いた場合との効果の違いを明らかにするために，検証実験 I と同様の実験をおこなった（**検証実験 II**）．

実験の目的は，Rank-by feature framework に基づく探索型可視化分析が仮説生成を支援できるかどうかを確認することである．具体的には，(1) 分析対象のドメイン知識の有無が生成される仮説数にどれくらいの影響を与えるのか，(2) 被験者が生成した仮説が国際会議等ですでに発表された知見に含まれるか（重要な知見につながる仮説を導けたか）どうかを，被験者実験により検証した．

表 1　実験に用いた課題データに含まれる変数（項目）

項目（変数）	詳細
AssignTime	修正者が決定されるまでの時間
Fixtime	修正されるまでの時間
Resolution	課題の最終状態（RESOLVED, VERIFIED など）
Severity	重要度
CCCount	メーリングリストの登録者数
Comments	コメントの件数
CommentWords	コメントの総単語数
DescriptionWords	記述情報の単語数
Reporter	報告者の名前
Triager	管理者の名前
Fixer	修正者の名前
Component	不具合が発生されたコンポーネント名

表 2　仮説生成実験 I（HCE ツール使用）の結果

被験者群	被験者	仮説生成数	仮説生成数（平均）	重要な仮説の数	重要な仮説の数（合計）
グループ A	A-1	5	6	0	4
	A-2	6		1	
	A-3	7		3	
グループ B	B-1	2	2.7	0	2
	B-2	3		1	
	B-3	3		1	

表 3　仮説生成実験 II（MS Excel 使用）の結果

被験者群	被験者	仮説生成数	仮説生成数（平均）	重要な仮説の数	重要な仮説の数（合計）
グループ C	C-1	3	4.3	0	1
	C-2	3		0	
	C-3	7		1	
グループ D	D-1	2	4.7	0	0
	D-2	4		0	
	D-3	8		0	

4.2. データセットと変数

実験では，Eclipse Platform プロジェクトの課題（不具合）データから，Eclipse 3.2 に関連する約 7,200 件をデータセットとして収集した．本データセットに含まれる 11 の変数を表 1 に示す．

4.3. 被験者

検証実験 I では，OSS 及びデータマイニングに関連する研究を行っている大学院生 3 名をグループ A とし，それらに精通していない大学生 3 名をグループ B として被験者を構成した．

検証実験 II においても同様に，OSS 及びデータマイニングに関連する研究を行っている大学院生 3 名をグループ C とし，それらに精通していない大学生 3 名をグループ D として被験者を構成した．

いずれの実験においても，被験者の重複はない（合計 12 名）．なお，被験者の大学院生は，本実験で対象とする OSS プロジェクトの課題データで用いられる変数の意味について，（個人差はあるものの）同程度の知識があると見なすことができる．一方，両実験の大学生は，情報科学分野で 3 年以上在籍し，履修科目の 1 つとしてソフトウェア工学を学んだ経験があるものの，OSS 及びデータマイニングに関してはほとんど知識のない被験者である．その意味で，実験課題に対しては同程度の知識量であったと見なすことができる．

4.4. 実験手順とタスク

実験に先立ち，被験者には HCE ツールの利用方法と使用する変数（メトリクス）について説明した後，食品に含まれる栄養素をまとめたサンプルデータセット（本実験には無関係のドメインにおけるデータ）を与え，HCE

ツールを約 1 時間試用してもらった．HCE ツールの利用に大きな支障がないことを確認した後，前述したデータセットを与えて実験を開始した．各被験者は隔離され，グループ内の被験者同士で相談等ができない環境でタスクに取り組んだ．実験時間は 1 時間とした．

被験者には，データセットに含まれる興味深い関係や傾向を事実としてまず抽出し，抽出した事実からプロセス改善や品質改善につながる案を仮説として生成するよう指示した．また，仮説を思いついた時点でメモとして記録しておくよう依頼した．被験者の様子は実験者が記録するとともに，画面キャプチャツールを用いて被験者の画面操作を記録した．

4.5. 実験結果

- 分析対象のドメイン知識の有無が生成される仮説数にどれくらいの影響を与えるか？

表 2 から，HCE ツールを用いた場合では，ドメイン知識の有無で生成できる仮説の数に違いがあることが見て取れる．一方，表 3 から，MS Excel を用いた場合では，グループ内での個人差が見られるものの，多くの仮説を生成できる被験者はグループ C, グループ D のどちらにも 1 名存在しており，ドメイン知識の有無によって生成できる仮説の数に違いがないことが分かる． また，ドメイン知識のないグループ B とグループ D が生成した仮説数は，MS Excel を用いたグループ D の方が多いことが見て取れる．

- 被験者が生成した仮説が国際会議等ですでに発表された知見に含まれるか？

被験者 A-3 が 3 件の重要な仮説を生成しているため，表 2 の結果のみで結論付ける事は困難であるが，表 2 の重要な仮説の合計から，HCE ツールを用いた場合では，ドメイン知識の有無が生成できる重要な仮説の数に違いを生むことが確認できる．ただし，ドメイン知識のないグループ B は，生成できた仮説数はすべてのグループの中で最も少ないものの，重要な仮説を 2 件生成できている．表 3 から，MS Excel を用いた場合では，生成できた仮説数はグループ C,D ともに比較的多いものの，重要な仮説はグループ C が 1 件生成できたのみであった．

検証実験 I 及び II の結果から，HCE ツールを用いた場合と，MS Excel を用いた場合とで，生成できる仮説数自体には大きな違いはなかったが，仮説の質の点では大きな違いがあることを確認できた．また，HCE を用いたグループ間にはドメイン知識の差が仮説生成の数と質の差として現れることが確認できた．一方，MS Excel を用いた場合では，ドメイン知識と仮説生成数とに関連は見当たらなかった．次章では，実験中の被験者の様子から本実験結果について考察する．

5. 考察

5.1. HCE ツールの利用

検証実験 I の結果から，ドメイン知識を有した被験者が数多くの仮説を生成できることが分かった．ただし，ドメイン知識のない被験者であっても，一人平均 2,7 件の仮説を生成でき，さらに，『「課題が報告されてから担当者に割り当てられるまでの時間」と「課題が割り当てられてから解決されるまでの時間」は反比例する.』といった趣旨の，すでにソフトウェア工学の研究分野で検証済みの重要な法則を仮説として生成できることが分かった．

HCE ツールを用いた被験者はまず，幾つかのクラスタを生成するところから分析を開始する．例えば，本論文の実験で用いた Eclipse Platform の不具合データを可視化している図 1-(a) では，3 つのクラスタが生成されていることを示している．生成された個々のクラスタは，データの特徴量が類似するデータの集合を意味する．したがって，被験者が着目（選択）したクラスタを分析する時点で，すでに幾つかの変数の間には何らかの強い関係が存在することになる. 図 1-(a) で選択中のクラスタ（真ん中のクラスタ）と，左右の 2 つのクラスタとの明らかな違いは，AssignTime と FixTime のスコア（値）の大きさから見て取れる．例えば，現在選択中のクラスタとその右側のクラスタとでは，AssignTime と FixTime のスコアの取り方に真逆の関係（一方が大きな値で一方が小さな値）があることを見て取ることができる．すなわち，AssignTime と FixTime には負の相関が存在する事を示しており，『「課題が報告されてから担当者に割り当てられるまでの時間」と「課題が割り当てられてから解決されるまでの時間」は反比例する.』という関係 [Ohira] を導きだす事ができる．

HCE ツールが提供する（画面下部で利用する）各種分析機能は，注目するクラスタ内の変数間の関係を詳細に調べる過程を支援するため，MS Excel を用いた被験者と比べ，仮説生成の質に違いが現れたものと考えられる．

表 4 に検証実験 I で被験者が生成した仮説の内，重要な知見につながる仮説を示す．HCE ツールを用いた被験者は，特定のクラスタやクラスタ内での変数間の関係に着目して分析していることが見て取れる．これらの仮説

は，ソフトウェア工学の研究分野においても重要性がすでに確認されており，支援手法についても多数提案されている [Anvik][Breu][Jeong][Ohira][Zimmerman] ものである．これらの結果からは，rank-by-feature framework に基づく探索型可視化方式は，探索的データ解析においてドメイン知識の不足する分析者であっても，課題対応プロセスの実施・適用技術の妥当性評価につながるものと考えることができる．

5.2. MS Excel の利用

HCE ツールを用いた被験者が重要な仮説を合計 6 件生成できたのに対して，MS Excel を用いた被験者は，生成できた仮説数自体は多いものの，重要な仮説を 1 件しか生成できなかった．

被験者の画面操作をキャプチャしたデータから，被験者は主に，SUM, AVERAGE, AVERAGEIF, COUNTIF などの関数，相関係数を求めるアドオン機能を使用し分析を行っていたことが分かった．また，興味のある 1 つの変数に着目し，着目した変数と他の変数との関係を 1 つ 1 つ調べるといった方法で分析を行っていることも分かった．HCE ツールを用いた分析では，選択したクラスタが「なぜクラスタとしてまとまりを持つのか？」を考えるために，すべての変数間の関係を俯瞰的に調べる機能（図 1-(b),(c) など）が利用できるのに対して，MS Excel を使った分析では，まず興味のある変数に着目し，その変数に関係する変数を見つけるのが分析の主眼となっていることが分かった．例えば，多くの被験者が課題の重要度 (Severity) に着目していたが，重要度そのものがデータセット全体の中で重要な変数ではない（課題の多くは重要度が適切に付与されていない）ため，結果的に重要な仮説を生成するのまでには至らなかったものと思われる．

特に，注目した変数と他の変数との関係を機械的に調べる傾向が強く，表 5 に挙げた被験者 C1 の仮説 C1-1 及び C1-2 のように，被験者自身が矛盾する仮説に気付いていないケースが多々見られた．このような傾向は，MS Excel を用いた検証実験 II だけではなく，検証実験 I においても共通して見られた傾向であった．したがって，このような傾向は，探索的データ分析をおこなう際の共通課題であるとも考えられる．探索的データ分析ツールを用いる際に，手本となる仮説生成までの手順をいくつか用意しておくことにより，今回見られた傾向を大きく

表 4　生成された重要な仮説

被験者群	事実	仮説
グループ A	AssignTime か？短いときは FixTime にばらつきがあるが，AssignTime が長ければ FixTime が短くなっている．	慎重に不具合を割り当てることが，プロダクトの品質改善につながる [Anvik].
	不具合が修正者に割り当てられるまでの時間 (AssignTime) は，管理者 (Triager) によって異なる.	修正依頼までに要する時間を短縮しようとしているプロジェクトとって，管理者の選出は非常に重要な要素となる [Ohira].
	修正依頼時間 (AssignTime) と修正時間 (FixTime) がともに長いクラスタに着目すると，descriptions の単語数 (DescriptionWords) が多い傾向にある.	不具合をより正確に報告することは，不具合の修正時間を改善するのに役立つ [Breu].
	コメント数 (Comments) と descriptions の単語数 (DescriptionWords) の散布図から，ほとんどコメントがなく単語数も少ないクラスタに分布する不具合は，コメント回数／単語ともに多い不具合に比べて，修正されるのが遅い (FixTime).	不具合報告の情報は，不具合修正プロセスを改善するために重要である [Breu].
グループ B	多くのコメントが付けられた (Comments) 不具合報告は，Resolution が RESOLVED や VERIFIED となっている．	不具合を確実に修正してもらうためには，開発者がコメントの内容を理解できるよう明瞭に不具合の内容を記述する必要がある [Zimmerman].
	管理者 (Triager) のヒストグラムを見ると，少数の管理者に不具合を多数割り当てていることが分かる．また，それらは修正されるのが遅くなる (FixTime) 傾向にある．	こうした事態を防ぐためには，適度な負荷分散をおこない，不具合修正プロセスを改善する必要がある [Jeong].
グループ C	コメント数が 10 件以上ある案件は AssignTime が長い．コメント数が 10 件以上ある案件は FixTime が短い.	複数人がとりかかる必要のある問題は修正を依頼する人を決定するのに時間がかかるが，活発な議論が繰り広げられる環境では早く問題が解決する [Ohira].

表 5　被験者 C1 が生成した仮説

仮説番号	事実	仮説
C1-1	不具合の深刻度 (Severity) が上がるほど，バグレポート当たりの CmmentsCount が増える	開発者は，バグの深刻度が高いレポートに関心を持つ
C1-2	不具合の深刻度が上がるほど，バグレポート当たりの Description words が増える	説明文の長いバグレポートは，深刻度の高いバグを報告している
C1-3	具合の深刻度と CommentsCount の相関係数は -0.11 であり，ほぼ相関がない.	開発者がバグレポートに興味を持つ要因は，バグの深刻度ではない.

改善できる可能性がある．

また，MS Excel を用いた被験者が重要な仮説を導きだせなかったその他の理由として，データの特徴量の類似度を基にデータのまとまりを構成するクラスタリング分析とは異なり，MS Excel を用いて SUM や AVERAGE 関数を用いて分析を行っても，対象プロジェクトにおける問題や課題を抽出するのが本質的に困難であった可能性がある．プロジェクトにおける問題や課題は，プロジェクトに所属する開発者（第一者）さえも認識しづらいものと考えられる（プロジェクト全体で認識された課題や問題であれば，すでに克服されている可能性が高い）ためである．したがって，SUM や AVERAGE 関数を用いてデータセット全体の傾向から分かる課題や問題は，プロジェクトにとって本質的なものではない場合が多く，MS Excel を用いた被験者のほとんどが重要な仮説につながる事実を抽出することができなかったものと思われる．

6. まとめ

本論文では，ソフトウェア品質の第三者評価における探索的データ解析ツールの利用とその効果について報告した．第三者が効率的に探索的データ解析を行えるようにするための探索型解析技術の選定をおこない，rank-by-feature framework [Seo205] に基づく探索型可視化方式を採用した．また，rank-by-feature framework を実装した探索的データ解析ツール HCE (Hierarchical Clustering Explore)[Seo2002][Seo2007] を用いた被験者実験をおこない，第三者評価における探索型解析技術の有用性を検証した．検証実験の結果から得られた知見は以下の通りである．

- rank-by-feature framework に基づく探索的可視化方式は，ソフトウェア品質の第三者評価，特に初期段階での探索的データ解析を支援することができる
- rank-by-feature framework に基づく探索的可視化方式によるデータ分析は，一般的な表計算アプリケーションでおこなうデータ分析に比べ，プロセスの実施・適用技術の妥当性評価につながる重要な仮説を導きだすのに有用である

本論文における検証実験は，オープンソースプロジェクトの課題データを用いておこなった．今後は，ソフトウェアプロジェクトトモグラフィにおけるスナップショットが提供する 5 つのペイン（構成画面）のデータすべてを用いて，プロジェクト及びプロダクトの異常検出や品質評価が行えるかどうかを，ソフトウェア開発企業のデータを用いておこなっていく必要がある．特に，第三者評価の観点から探索型可視化技術の有用性を検証するために，大学の研究者等が生成した仮説と，実際に開発に携わった管理者・開発者などの実務者が生成した仮説がどの程度一致するかを確認することが求められる．

謝辞

本研究は，独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター（SEC: Software Reliability Enhancement Center）が実施した「2012 年度ソフトウェア工学分野の先導的研究支援事業」の支援を受けたものです．本研究の被験者実験には，奈良先端科学技術大学院大学情報科学研究科の皆様，和歌山大学システム工学部の皆様に御協力頂いた．

【参考文献】

[Anvik] J. Anvik, L. Hiew, G.C. Murphy, “Who should fix this bug?,” Proceedings of the 28th international conference on Software engineering (ICSE’06), pp. 361–370, 2006.

[Breu] S. Breu, R. Premraj, J. Sillito, T. Zimmermann, “information needs in bug reports: improving cooperation between developers and users,” Proceedings of the 13th ACM conference on Computer supported cooperative work (CSCW’10), pp. 301–310, 2010.

[Cluzet] S. Cluzet, C. Torregrosa, C. Jacquet, C. Lafitte, J. Fournier, L. Mercier, S. Salamagne, X. Briand, M. T. Esquerre-Tugaye, and B. Dumas, “Gene expression profiling and protection of Medicago truncatula against a fungal infection in response to an elicitor from green algae Ulva spp,” Plant, Cell and Environment, Vol.27, pp.917-928, 2004.

[IPA] 情報処理推進機構 ,“ ソフトウェアの品質説明 力強化のための制度フレームワークに関する提案 (中 間報告)” (平 23-9).

[IPA2012] 2012 年度ソフトウェア工学分野の先導 的研究支援事業「ソフトウェア品質の第三者評価の ための基盤技術 - ソフトウェアプロジェクトトモグラフィの開発 -」成果報告書 ,

http://www.ipa.go.jp/files/000026806.pdf

[Jeong] G. Jeong, S. Kim, T. Zimmermann, “Improving bug triage with bug tossing graphs,” Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering (ESEC/FSE’09), pp. 111–120, 2009.

[Ohira] M. Ohira, A.E. Hassan, N. Osawa, K. Matsumoto, “The Impact on Bug Management Patterns on Bug Fixing: A Case Study of Eclipse Projects,” Proceedings of the 28th International Conference on Software Maintenance (ICSM’12), pp. 264–273, 2012.

[Seo2002] J. Seo, B. Shneiderman, “Interactively Exploring Hierarchical Clustering Results,” IEEE Computer, Volume 35, Number 7, pp. 80-86, 2002.

[Seo2005] J. Seo, B. Shneiderman, “A rank-by-feature framework for interactive exploration of multidimensional data,” Information Visualization, Vol.4, No.2, pp.96-113, 2005.

[Seo2007] J. Seo and H. Gordish-Dressman, “Exploratory Data Analysis with Categorical Variables: An Improved Rank-by-Feature Framework and a Case Study,” International Journal of Human-Computer Interaction, Vol.23, No.3, pp.287-314, 2007.

[Tsai] J.-M. Tsai, H.-C. Wang, J.-H. Leu, H.-H. Hsiao, A. H. J. Wang, G.-H. Kou, C.-F. Lo, “Genomic and proteomic analysis of Thirty-nine structural proteins of shrimp white spot syndrome virus,” Journal of Virology, Vol.78, pp.11360-11370, 2004.

[Tukey] J.W. Tukey, “Exploratory Data Analysis,” Addison-Wesley, 1977.

[Zimmerman] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schroter, C. Weiss, “What Makes a Good Bug Report?,” IEEE Transactions on Software Engineering (TSE), Vol. 36, No. 5, pp. 618–643, 2010.

情報システムの事故データ

情報システムの障害状況 2013年後半データ

IPA 顧問
松田　晃一

SEC 研究員
目黒　達生

SEC 調査役
大髙　浩

2013 年 7 月から 12 月までに報道された情報システムの障害状況を報告する。この間に報道された情報システムの障害は合計 9 件、月平均 1.5 件／月であった。これは、2009 年、2010 年とほぼ同じ水準で、比較的低い水準であった。また、事故の影響範囲や重大性といった質的な面からみても、幸いにして比較的軽微なものが多かったと言える。

1．はじめに

本稿では、2013 年 7 月から 12 月までの 2013 年後半の半年間に報道された情報システムの障害状況をとりまとめて報告する。また、これらの事例の中から、保守作業の手順確認とハードウェア故障をカバーするソフトウェアの限界について若干の考察を試みる。これらは、事例に限った特殊な問題ではなく、他のシステムにも共通の問題であり、今後の事故防止の参考となると思われる。

2．2013 年後半の概況

2013 年 7 月から 12 月までの半年間で報道された情報システムの障害は合計 9 件となった。その全体は表 1 に示す通りであり、障害発生件数を月平均にすると 1.5 件／月となる。半年間のデータだけでみると比較的低い水準であるが、2013 年前半の件数 21 件 [松田 3 2013] を加えて年間通して平均すると、2.5 件／月となる。これは 2011 年の平均値 2.25 件／月 [松田 2 2012] と同水準であるが、2010 年の平均値 1.47 件／月 [松田 2011]、2009 年の平均値 1.3 件／月 [経産省 2009] に比べるとまだ高い水準にある。2013 年後半の月別発生件数と 2007 年以降の月平均件数の推移を図 1 に併せて示す。

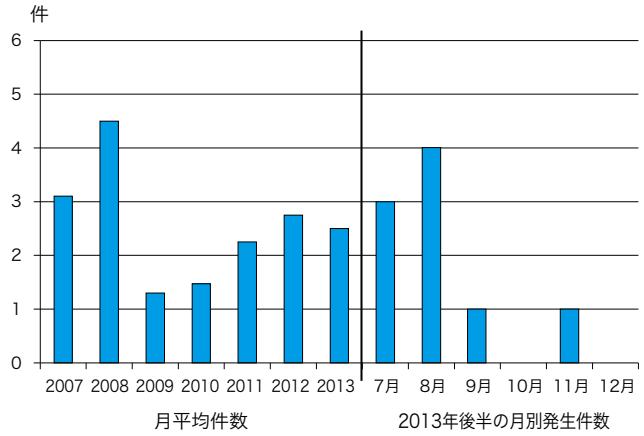


図 1　情報システムの障害発生件数の推移

2013 年前半には、通信事業者の携帯電話サービスの障害が繰り返し発生し、広範なユーザに影響を与えたが、幸い後半にはそのような大規模な事故は報道されていない。しかし、列車運行システム（事例 1324）※1 や空港の出入国管理システム（事例 1322）の故障によって多くの旅客が影響を受けたり、病院の医療情報システム（事例 1330）の故障によって病院の機能が大幅に低下するなどの事例が発生した。

3．保守手順の事前確認の限界

コンピュータシステムにおいては、ハードウェアの定期的な保守点検やソフトウェアのバージョンアップなどの保守作業は避けられない。通常の定型的な運用作業と異なって非定型的な作業となることが多いため、保守作業が切掛けとなって障害が発生する事例も多い。

このため、保守作業にあたっては事前に周到に準備をして実施することが求められる。事例 1324 の列車運行

【脚注】

※1　事例に付与されている番号の前半 2 桁は事例が発生した西暦年の下 2 桁、後半 2 桁はその年に発生した事例の通し番号であり、本連載を通して一意の番号となっている。

表１　2013 年前半の情報システム障害データ（報道に基づき SEC が整理）

No.	システム名	発生日時（上段） 回復日時（下段）				影響	現象と原因	直接原因	情報源
		年	月	日	時				
1322	成田国際空港　手荷物搬送システム	2013	7	11	7 時 20 分	成田空港の第 2 ターミナルの南側出国審査場において手荷物搬送システムの障害が発生。 出発客の手荷物をベルトコンベヤーで自動搬送する装置と、出国審査場 1 カ所のコンピューターシステムに相次いで障害が起きて停止した。出発 8 機に 30 ～ 50 分の遅れが生じた。	成田空港第 2 ターミナル内の手荷物搬送システム（BHS）の不具合は、同日 7 時 20 分に爆発物検査装置（EDS）のサーバー障害により手荷物搬送システム（BHS）が停止したもの。 不具合発生後の調査の結果、爆発物検査装置（EDS）のサーバー通信不良が判明したため、部品交換を実施し、同日 10 時 14 分に復旧。	機器障害	・日本経済新聞（2013.7.11） ・毎日新聞（2013.7.11） ・朝日新聞（2013.7.11）
		2013	7	11	10 時 14 分				
1323	愛知県犬山市の投票システム	2013	7	15	10 時 30 分	投票しようとした有権者の氏名と選挙人名簿を照合できなくなった。 約 30 人が投票所を訪れ、9 人は電話で名簿と確認して投票できたが、約 20 人は本庁舎の投票所に回ってもらうなどの措置をとった。	同市羽黒の南部公民館の期日前投票所で本庁舎と結ぶシステムが通信不能となった。	不明	・毎日新聞（2013.7.16）
		2013	7	15	11 時 20 分				
1324	JR 九州列車運行システム	2013	7	18	4 時 00 分	列車運行システムの障害により 16 路線、385 本が運休、約 11 万 1 千人に影響。 鹿児島線の門司港－鹿児島中央間、若松線の若松－折尾間、福北ゆたか線の折尾－博多間で、列車の運行が一時確認できなくなったため、鹿児島線、若松線、福北ゆたか線の 3 区間で運行を止めた。 このことにより、この 3 線と接続している路線でダイヤが乱れた。運休または遅延が発生したのは、3 路線のほか香椎線、日田彦山線、後藤寺線、長崎本線、佐世保線、日豊本線、久大本線、豊肥本線、肥薩線、吉都線、三角線。	システム導入 (1997 年) 以降、初めて行った信号機を自動制御する機器「自動列車進路制御装置（PRC）」にある基板部品の交換時に不具合が発生。この外部記憶装置は 2010 年 7 月、システムを構築したメーカが定期交換で、それまで記録用に使っていたハードディスクを SSD に交換して、取り付けていた。PRC では内部に取り付けた記憶装置の再立ち上げが 0.2 秒以内に完了しなければならない設定になっていた。メーカが 2010 年に取り付けていた記憶装置は 0.3 秒かかるものだったため設定が異なるこの記憶装置への情報のやりとりができなくなり、PRC が立ち上がらず、システム全体にトラブルが波及した。	設定値誤り	・ITPro (2013.7.18) ・日本経済新聞 (2013.7.23) ・朝日新聞 (2013.7.23) ・毎日新聞 (2013.7.18) ・日経エレクトロニクス (2013.8.19) ・日経コンピュータ (2013.9.19)
		2013	7	18	6 時 54 分				
1325	気象庁緊急地震速報	2013	8	8	16 時 56 分	気象庁は、「奈良県を震源、マグニチュード（M）最大 7.8」とする緊急地震速報を 34 都府県に発表。 緊急地震速報に伴い、東海道・山陽新幹線は小田原－新岩国間で午後 5 時 15 分まで運転を見合わせた。近畿の JR 在来線や私鉄を含め 40 万人超に影響した。 実際に同時刻ごろに起きた地震の震源地は和歌山県北部で、M 2.3 だった。	和歌山県北部で実際に起きたマグニチュード (M)2.3 の地震と三重県南東沖に設置している海底地震計のノイズ異常が重なり、データを処理する静岡県御前崎市の陸上中継局で誤った処理をおこなったのが原因。 問題の地震計は 8 日午前 3 時ごろから数回と、午後 4 時 43 分ごろから 10 回程度、信号が途切れる不具合が続いていた。 信号の途切れだけでは地震発生と判定されないが、和歌山県の地震が偶然重なったことで誤報につながった。 通常、地震計は地震の揺れによらないわずかな地面の動きを常にとらえ、地震とは違う「ノイズ」と見なす処理をしている。だが、今回、三重県沖ではノイズが 2 秒程度途切れたため、地震計がその後のノイズを揺れと誤認した。	機器障害 ソフトウェア障害	・日本経済新聞 (2013.8.8) ・朝日新聞 (2013.8.8) ・気象庁報道発表（2013.8.21）
1326	ゆうちょ銀　Web サイト	2013	8	20	11 時 00 分頃	ゆうちょ銀行のウェブサイトが表示できない、あるいは表示までに時間がかかるという事象が発生。 ネットバンキングなど一部のインターネットサービスも接続しづらい状況が発生。	不明	不明	・日本経済新聞（2013.8.20） ・朝日新聞 (2013.8.20) ・ゆうちょ銀行報道発表 (2013.8.21)
		2013	8	20	22 時 30 分頃				
1327	みなと銀行	2013	8	31	9 時 00 分	兵庫県と大阪府にある同銀行の支店、市役所やコンビニエンスストアなど店舗外を含め 2 府県の全てのＡＴＭ（１６４カ所）が使用不能になり、現金の引き出しをはじめ全取引がストップ。	同行の Web サイトによると、オンラインプログラムの不具合によるもの。	ソフトウェア障害	・日本経済新聞 (2013.9.1) ・みなと銀行報道発表 (2013.9.2)
		2013	8	31	16 時 00 分				
1328	ヤフー公金支払システム	2013	8			川崎市は 9 日、インターネットを通じて公共料金を支払えるサービス「Yahoo! 公金支払い」で、水道利用者 1 件分の水道料金を市外に住む別人から徴収していたと発表した。同市に 8 月、京都府の女性から使った覚えのない水道料金（4 カ月分 6508 円）が引き落とされたと問い合わせがあり発覚した。本来の水道利用者がカードの番号と有効期限を誤って入力し、それがこの女性と偶然一致したため、引き落とし登録がされた。	ヤフーによると、このサービスを利用してクレジットカードで支払う場合、カードの番号と有効期限、セキュリティーコード（SC ＝裏面記載の 3 桁数字）を入力して本人照会する。しかし、水道料金についてはシステムミスで SC による照会がされていなかった。	ソフトウェア障害	・毎日新聞（2013.9.9） ・日本経済新聞 (2013.9.10) ・日経コンピュータ (2013.10.17)
		2013	9						

1329	ソフトバンク　信用情報送信システム	2009	10	8		ソフトバンクモバイルは 2013 年 10 月 1 日、一部のユーザーに対して、支払い済みの携帯端末の分割代金を誤って「未入金」扱いで信用情報機関に登録してしまったと発表した。誤って登録してしまった情報は 6 万 3133 件。そのうち信用情報機関に対して照会がかけられた情報は 1 万 6827 件。（期間は 2009 年 10 月 8 日～ 2013 年 8 月 6 日）。そのため信用情報機関の取引時に悪影響を受けた可能性がある。ソフトバンクモバイル広報によると、現時点で同社に「実際に信用情報を使った取引に問題が出た」と申告してきたユーザーの数は 12 件。	誤登録の原因は、あるシステムのプログラムの設定ミス。「ソフトバンクモバイルがユーザーの入金状況などを管理している課金システムから、信用情報機関に対して情報を送るシステムがある。この情報を送るシステムのプログラムの設定や確認の作業に、人為的なミスがあった」（同社広報）。	プログラム設定ミス	・日本経済新聞（2013.10.2） ・日本経済新聞（2013.10.3） ・日経コンピュータ（2013.11.14）
		2013	9	末					
1330	淡路医療センター　電子カルテシステム	2013	11	6	8 時 30 分頃	6 日午前 8 時半ごろ「電子カルテシステム」が正常に動かなくなり、医師がカルテを検索できなくなったほか、手術患者の脈拍や呼吸などを計測する管理モニターも停止し手術ができなくなった。障害発生から約 6 時間半後の 6 日午後 3 時ごろに一部の手術室でモニターが復旧したため、この日予定された 11 件の手術のうち 6 件を夕方から深夜にかけて実施。残りは翌日以降に延期した。また、ネットワークでつながる外来患者の診察券受け付け機器も使用不能となり、受け付け業務がストップ。職員が急きょ手作業で受け付けをおこなった。同センターは普段 1 日平均約 800 人の外来患者を診察するが、急を要しない患者に後日の受診や他の医療機関での診察を呼びかけ、この日は約 200 人にとどまった。	業者の調査で機器の不具合が原因と判明。	機器障害	・朝日新聞（2013.11.8）
		2013	11	7	4 時 00 分				
別枠 1301	ニフティメールシステム	2013	9	30	13 時 38 分	@nifty メール（セカンドメール、セカンドメール PRO などを含む）の一部のユーザにおいて、以下の機能が利用できなくなった。 （1）メールソフトを使っての受信 （2）WEB メールへのアクセス（PC 版、フィーチャーフォン版、スマートフォン版） なお、メールの消失および第三者による閲覧の事実は確認されなかった。	不明。	不明。	・日本経済新聞（2013.10.1） ・ニフティ Web サイト（2013.10.1）
		2013	10	1	4 時 47 分				
別枠 1302	LINE	2013	11	19	11 時 05 分	午前 11 時 5 分、全世界の iPhone ／ Android 版ユーザーの一部で通信障害が発生。ホーム、タイムライン、LINE GAME などの付加機能が午前 11 時 35 分に、通話・トーク機能は午後 12 時 2 分に復旧したという。影響を受けたユーザーの数は明らかにしていない。 不具合のあった時間帯、Twitter では「メッセージが送れない」といった声が多数ツイートされた。	運営元の LINE は「内部システムの一時的な負荷の増加によるエラー」が原因だったことを明らかにした。	不明	・ITMedia ニュース（2013.11.19） ・ITMedia ニュース（2013.11.20）
		2013	11	19	12 時 02 分				

システムのケースにおいても、事前に手順を確認するなどの準備をして作業を実施したにもかかわらず、障害が発生してしまった。この事例では、信号機を制御する自動列車制御装置の LAN ボードを交換する保守作業を実施するため、あらかじめ検証用の設備を使って部品交換の手順を確認しておいた。確認済の手順で本番系の部品交換を実施し、システムを再起動したが記憶装置の立上げがタイムアウトとなり再起動に失敗し、事故に繋がったと報道されている。手順の確認を行ったのは検証系であり、そこで使われていた記憶装置は HDD であったのに対し、本番系の記憶装置は SSD であったため、その起動時間に差があり、事前の手順確認時には発生しなかったタイムアウトが本番では発生してしまったとのことである [日経コンピュータ 2013]。なお、SSD の起動がなぜ一定時間内に終わらなかったか、についてはこの SSD が交換後 3 年以上を経過しており、NAND 回路のビット誤りが増えたための誤り訂正に時間を要したのではないか、との可能性が報道されている [日経エレクトロニクス 2013] が詳細は不明である。いずれにしても、この事例においては手順の確認を実機で事前に実施したにもかかわらず、本番と違った環境での確認であったために記憶装置の起動時に予期しないタイムアウトが発生し、事故を起こしてしまった。

このような例はまれではなく、例えば大規模な EC サイトを運営している企業のシステムにおいても同様の事例が見られた。（システム障害に関する経験交流の場での非公式報告）。すなわち、検証のための環境においてあらかじめ十分に確認した保守の手順を本番環境に適用したところが、長時間応答が無く、確認した手順とは違った振る舞いになったと判断し、あわてて手順書には無い操

作を行って事故を起こしてしまった例である。本番環境には膨大な量のデータがあるが、検証の環境にはそのような大量のデータを持たなかった。このため、検証時には直ちに応答が返った処理が、本番環境では長時間を要したためにこのような現象となった。結果的に手順は間違っていなかったが、検証した時とは異なったシステムの動作にあわてたオペレータが誤ったオペレーションをしてしまい、事故に至ってしまったとのことである。

このいずれの事例も、検証の環境が本番環境とは同一で無かったことに起因した問題である。リソース量などの制約から、本番環境と全く同じ規模の検証環境を持つ方が稀であって、どこのシステムにおいても起こりうる問題である。検証環境での事前の検証はもちろん非常に有効であるが、それには限界があり、特に量に関する条件、時間に関する条件などは本番とは異なる可能性を十分に意識しておくことが重要である。手順の事前確認は大切ではあるが、それは万全でないことを肝に銘じたい。

4. ソフトウェア検証環境の限界

事例 1325 に示す緊急地震速報の事例は、幸いなことに結果的には誤報であったが、警報を受けた多数の市民が大きな揺れに身構え、交通機関は運転を見合わせるなど影響は大きかった。気象庁のその後の発表 [気象庁 2013] によれば、この障害は二つの事象が重なって発生したとのことである。その一つは、海底に敷設されている地震計のデータを受信し、時刻を付与してデータ処理装置に送信する装置の故障、もう一つはそのような機器の故障をカバーするためのソフトウェアの不良である。ある地震計のデータに、前者の原因で不正な時刻が付与されたデータが処理装置に送られた。受信したデータ処理装置のソフトウェアは、送られてきたデータの時刻が不正で、データの連続性が途切れたことを検出した場合には、入力データを捨てて直前の値を保持するのが本来行うべき処理であった。ところが、今回は不正な時刻が付与された入力データを 0 に置換して以降の処理を行った（前述の第二の事象）ため、データに大きな変位が表れてしまい、地震計に大きな加速度が急激にかかり強い揺れが発生したと判断され誤った地震速報が出された。

このように機器の障害に対してそれをカバーするための機能がソフトウェアに実装されることは一般に行われるが、そのようなソフトウェアに不良があると今回のような問題を発生してしまう。機器の異常を契機に動作するソフトウェアは、その検証が非常に難しい。ソフトウェアの検証のために、機器の障害を実際に発生させるのはそれほど簡単ではない。このために実際の機器を疑似するハードウェアシミュレータなどによってハード障害を疑似して可能な限りソフトウェアの確認をするが、所詮それには限界がある。また、故障の再現ができたとしても、そのすべてのケースを網羅的に検証することは現実的ではない。特にタイミングにかかわる故障などをすべて再現して事前に確認することは不可能であり、このようなソフトウェアの動作は実質的にはぶっつけ本番にならざるを得ない。前 3 節で述べた例と、問題の性質は大きく異なるが、本番環境と検証環境が異なることによる限界という意味では同様の問題を示している。

5. むすび

2013 年後半 6 ヶ月間の情報システムの障害について、報道などをもとに整理し報告した。今期の事故件数はやや低い水準であったが、大きく減少したとは言えない。今期の事故事例の中からもこれからの開発・運用にあたって参考にすべき多くの教訓を汲み取ることができる。今後とも、これらの経験を社会の共通の財産として共有し、少しでも事故を防ぎ、安心・安全な IT 社会に向けて地道な努力を続けていく必要があろう。

SEC では、様々な事故の事例から学んで教訓として見える化する活動を 2013 年 4 月から開始し、それらをステークホルダーで共有・活用を促す部会活動を 8 月から始めている。現在同部会には通信、銀行、証券、保険、鉄道、電力、ガス、行政の 8 業界から有識者に委員として参画いただいている。

報道されるシステム障害を一層減らすために、より多くの業界の方々に、この事業への積極的な参画を呼びかけたい。

重要インフラ IT サービス教訓見える化活動
並びに同・高信頼化部会に関するお問合せ先：
IPA/SEC 目黒　達生 03-5978-7543, t-meguro@ipa.go.jp

【参考文献】

[日経エレクトロニクス 2013]SSD への交換が引き金 JR 九州のシステム障害、日経エレクトロニクス 2013.8.19、pp12
[日経コンピュータ 2013] システム障害で 11 万人に影響 部品交換で無停止機がストップ、日経コンピュータ 2013.9.19、pp88-90
[気象庁 2013] 気象庁報道発表資料 : 8 月 8 日 16 時 56 分頃の緊急地震速報の過大な震度予想の原因と対処について ,2013.8.21
[松田 2011] 松田晃一・金沢成恭：情報システムの障害状況 2010 年データ、SEC journal No26,Vol. 7, No3,pp.102-104,Oct.2011
[松田 2 2012] 松田晃一・金沢成恭：情報システムの障害状況 2011 年後半データ、SEC journal No28,Vol. 8, No1,pp.6-pp.8,Mar.2012
[松田 3 2013] 松田晃一・鈴木三紀夫・大髙浩：情報システムの障害状況 2013 年前半データ、SEC journal No34,Vol. 9, No3,pp.142-pp.146,Sep.2013
[経産省 2009] 経済産業省、（独）情報処理推進機構（IPA）、（社）日本情報システム・ユーザー協会：重要インフラ情報システム信頼性研究会 報告書、2009.3

SWEBOK V3.0 の紹介

SC7/WG20※1 エキスパート　新谷 IT コンサルティング 代表
新谷　勝利

1 はじめに

SWEBOK は Software Engineering Body Of Knowledge の省略で、2001 年に SWEBOK Trial 版が発行され、2004 年に SWEBOK 2004(以降 2004 年版とする) が発行され、2013 年に SWEBOK V3.0（以降 V3.0 とする）が発行されている。

現在、SWEBOK V3.0 は以下の URL※2 から無償で入手可能であり、本文は URL に記述の手順によりダウンロードしたものをベースにしている。また、日本語訳も進められており年内には発行の予定とのことである。V3.0 を紹介するにあたり、現在翻訳されている「ソフトウェアエンジニアリング基礎知識体系—SWEBOK 2004※3 と比較することにより、その改訂の程度を見ていただくことが V3.0 をより理解し易いと考え、随時参照する。参照するにあたり、V3.0 のオリジナルの英語と筆者による仮約を括弧で示す。仮訳はできるだけ 2004 版の訳に合わせている。2004 年版の訳で、その後 ISO の関連する規格の JIS 化で用いられた訳語はその旨本文に示す。

2 SWEBOK 開発及び維持の目的

V3.0 の Introduction to the guide（ガイドへの序説）は、2004 年版の第 1 章ガイドへの序説に相当し、ソフトウェアエンジニアリングについて ISO/IEC/IEEE Systems and Software Engineering Vocabulary（SEVOCAB）※4 における定義として以下を説明している。

「ソフトウェアの開発、運用及び保守に対して、システマチックでよく訓練された定量化可能なアプローチを適用すること、すなわち、エンジニアリングをソフトウェアに適用すること」

2004 年版の翻訳においては、IEEE コンピュータソサイエティの定義が用いられており、V3.0 ではより実践的な定義となっているように思える。

2004 年版の開発においては、21 カ国から 120 人の査読者が参加したのに対して、V3.0 の開発においては、33 カ国 150 人の参加があったということで、ソフトウェアエンジニアリングの基礎知識体系をより多くの国に広めるということが達成されているように思える。特筆すべきは、2004 年版のエディターで V3.0 にも継続して参加したのは 6 人であり、新規のエディターとして北米以外から、特に中国とインドから多く加わっていることである。SWEBOK の開発に当たっては、誰でも IEEE のホームページから、あるいは、ISO/IEC JTC1 SC7/WG20 の委員であれば査読に参加可能であった。エディターとして日本からの貢献が無いことも寂しいが、査読者も筆者を含め 6 人しか記録されていないのは残念である。

上述の手順を経て、SWEBOK はその技術内容が最新の状況を反映するように、読む人に理解し易く、そして、使用し易いように、2001 年の最初の発行以来改訂されてきている。どのような改訂が、最新の V3.0 に対してその前の 2004 年版からなされたかのの概要については次章にて紹介する。

3 V3.0 と 2004 年版の比較

両書の目次構造上の比較をすると以下の表にまとめることができる。目次の各章はそれぞれ知識領域に対応し、2004 年版の 11 に対して、V3.0 では、15 の知識領域を

【脚注】
※1　SC7/WG20 は、ソフトウェア及びシステムの知識体系とプロフェッショナルの形成にかかわる国際標準を審議する委員会。SWEBOK を審議。
※2　http://www.computer.org/portal/web/swebok/swebokv3
※3　ソフトウェアエンジニアリング基礎知識体系—SWEBOK 2004ー、松本吉弘訳、オーム社、平成 17 年 6 月 20 日、ISBN4-274-50029-2
※4　www.computer.org/sevocab

V3.0	2004 年版
Introduction to the guide （ガイドへの序説）	第 1 章　ガイドへの序説
Chapter 1: Software Requirements （第 1 章：ソフトウェア要求）	第 2 章　ソフトウェア要求
Chapter 2: Software Design （第 2 章：ソフトウェア設計）	第 3 章　ソフトウェア設計
Chapter 3: Software Construction （第 3 章：ソフトウェア構築）	第 4 章　ソフトウェア構築
Chapter 4: Software Testing （第 4 章：ソフトウェアテスティング）	第 5 章　ソフトウェアテスティング
Chapter 5: Software Maintenance （第 5 章：ソフトウェア保守）	第 6 章　ソフトウェア保守
Chapter 6: Software Configuration Management （第 6 章：ソフトウェア構成管理）	第 7 章　ソフトウェア構成管理
Chapter 7: Software Engineering Management （第 7 章：ソフトウェアエンジニアリング・マネジメント）	第 8 章　ソフトウェアエンジニアリング・マネジメント
Chapter 8: Software Engineering Process （第 8 章：ソフトウェアエンジニアリングプロセス）	第 9 章　ソフトウェアエンジニアリングプロセス
Chapter 9: Software Engineering Models and Methods （第 9 章：ソフトウェアエンジニアリングモデル及び手法）	第 10 章　ソフトウェアエンジニアリングのためのツール及び手法
Chapter 10: Software Quality （第 10 章：ソフトウェア品質）	第 11 章　ソフトウェア品質
Chapter 11: Software Engineering Professional Practice （第 11 章：ソフトウェアエンジニアリングプロフェッショナルプラクティス）	第 12 章　ソフトウェアエンジニアリングに関連するディシプリン
Chapter 12: Software Engineering Economics （第 12 章：ソフトウェアエンジニアリングエコノミックス）	
Chapter 13: Computing Foundations （第 13 章：コンピューティング基盤）	
Chapter 14: Mathematical Foundations （第 14 章：数学基盤）	
Chapter 15: Engineering Foundations （第 15 章：エンジニアリング基盤）	
Appendix A: Knowledge Area Description Specifications （付録 A: 知識領域記述のための仕様）	付録 A SWEBOK の Ironman バージョンにける知識領域記述のための仕様
Appendix B: IEEE and ISO/IEC Standards Supporting the Software Engineering Body of Knowledge (SWEBOK) （付録 B: ソフトウェアエンジニアリング基礎知識体系（SWEBOK）を支援する IEEE と ISO/IEC の規格）	付録 C IEEE 及び ISO ソフトウェアエンジニアリング標準の SWEBOK 知識領域への割り付け
Appendix C: Consolidated Reference List （付録 C: 統合された参照文献リスト）	

定義している。各知識領域は構造化され最初のレベルは副知機領域を定義している。各副知識領域は更にトピックスとして細分化され、各トピックスは必要に応じ副トピックスとして細分化されている。2004 年版から V3.0 の改訂では、単に構造が異なるのみならず、トピックス等の入れ替えも行われているので、ここでは、単なる目次から得られる違いを指摘するだけでなく、知識領域、副知識領域、トピックスとそれぞれの版の内容まで確認し、違いを明記している。

両方を新しい章番号で示される知識領域と副知識領域の順番に見てゆくと、以下のことに気がつく。

① 章の番号は別として、2004 年版では第 10 章で「ツール」とタイトルに入っているが V3.0 では入っていない。これは、V3.0 では、「ツール」を一つにまとめた知識領域とするのではなく、各知識領域の副領域としたことによる。

② 章番号の知識領域の比較だけでは分からないが、2004 年版の第 3 章のソフトウェア設計に User

Interface Design（ユーザーインターフェース設計）という副知識領域が追加されて V3.0 の Chapter 2: Software Design（第 2 章：ソフトウェア設計）になっている等があり、このような各知識領域内の副知識領域の変更点は後述する。

③ 2004 年版の第 12 章では「ディシプリン」という用語が用いられ、V3.0 の Chapter 11（第 11 章）では Professional Practice（プロフェッショナルプラクティス）という用語が用いられている。共に日本語として日常的に使用されないが、ディシプリンは、2004 年版の第 12 章では、以下の副知識領域をカバーするものとされている。

- コンピュータエンジニアリング
- コンピュータサイエンス
- マネジメント
- 数学
- プロジェクトマネジメント
- 品質マネジメント
- ソフトウェアエルゴノミクス
- システムエンジニアリング

V3.0 においても同じくディシプリンという用語はもちいられているが、2004 年版にあるソフトウェアエルゴノミクスは外され、一部は前述②に紹介されているように、V3.0 の第 2 章にユーザーインターフェース設計として新しい副知識領域が設定されている。また、2004 年版におけるコンピュータエンジニアリングと数学は、V3.0 では、新しく第 13 章コンピュータ基盤と第 14 章数学基盤という知識領域が新しく設定されている。ディシプリンというものの必要性は 2004 年版から変わったものではないが、V3.0 においては、関連する知識領域において、具体的に紹介されている。

④ V3.0 の Chapter 11: Software Engineering Professional Practice（第 11 章：ソフトウェアエンジニアリングプロフェッショナルプラクティス）は副知識領域として、以下の幅広い知識トピックスをカバーしている。

- Professionalism（プロフェッショナリズム）
- Group Dynamics and Psychology（グループダイナミックス及び心理学）
- Communications Skills（コミュニケーションスキル）

これらは、③に示す 2004 年版の第 12 章のディシプリンの中の副知識領域の一つであるプロジェクトマネジメントの一部をカバーしている。

⑤ 2004 年版の第 12 章のマネジメント副知識領域は、以下のトピックスをカバーしている。

- 会計学
- 財政学
- マーケティング及び営業
- オペレーション・マネジメント
- 情報システムマネジメント
- 法律
- 人材マネジメント
- 経済学
- 定量分析
- ビジネス政策及び戦略

当副知識領域はディシプリンの 1 つであるが、V3.0 では、次の⑥で示すように、第 12 章として独立した知識領域でほとんどがカバーされている。

⑥ V3.0 の Chapter 12: Software Engineering Economics（第 12 章：ソフトウェアエンジニアリングエコノミックス）は以下の副知識領域をカバーしており、⑤にて示す 2004 年版のマネジメント副知識領域にオーバーラップするものがあるがエコノミックス手法に特化拡張している。

- Software Engineering Economics Fundamentals（ソフトウェアエンジニアリングエコノミックス基礎）
- Life Cycle Economics（ライフサイクルエコノミックス）
- Risk and Uncertainty（リスクと不確実性）
- Economic Analysis Methods（エコノミックスアナリシス手法）
- Practical Considerations（実践上の考慮事項）

⑦ ③で示す 2004 年版の第 12 章は、V3.0 の Chapter 11-15（第 11 章から第 15 章）に拡張されているといってもよいであろう。よって V3.0 の本文では、副知識領域における変更はあるものの、Chapter 12：Software Engineering Economics（ 第 12 章：ソフトウェアエンジニアリングエコノミックスから Chapter15：EngineeringFoundations（エンジニアリング基板））が大幅に強化拡張がされた改訂版

と言えるであろう。

⑧ V3.0 の付録は、Appendix B(付録 B）は、システムズエンジニアリング及びソフトウェアエンジニアリングにかかわる国際規格を策定・維持している ISO/IEC JTC 1/SC 7 と IEEE の間の 10 年にわたる協働作業の成果として SWEBOK がカバーする領域にかかわる規格を数多く紹介すると共にどの規格がどの知識領域に特に関係するかを示している。特に Appendix C（付録 C）は IEEE におけるソフトウェアエンジニア認証に参考になるものがリストアップされており、2004 年版以降の関連情報の変化がまとめられている。

2004 年版にある用語集については、翻訳者が日本人読者のために追加されたもので、V3.0 が翻訳される時にも同様なものが追加されると読者にとり有用であろう。

現在、ソフトウェアエンジニアリングにかかわる用語集としては、「2. SWEBOK の開発及び維持の目的」で紹介した SEVOCAB が広く使用されている。ただ、残念ながらこれはまだ日本語に翻訳されていない。

4 V3.0 における主たる改訂箇所の概説

① Chapter 1: Software Requirement（第 1 章：ソフトウェア要求）

- Requirement Analysis（要求分析）副知識領域に以下が追加されている。
  - Formal analysis（形式手法による分析）: 形式手法に基づく仕様記述が、1）要求の記述において誤解の回避、2）要求の理由付け、において幾つかの適用分野において有効であることが紹介されている。

② Chapter 2 :Software Design（第 2 章：ソフトウェア設計）

- Key Issues in Software Design（ソフトウェア設計における主要な問題）副知識領域に以下が追加されている。
  - Security（セキュリティ）: セキュリティに関連する諸事項を設計段階から取り込む。
- 新しく User Interface Design(ユーザーインターフェース設計）副知識領域が追加されている。

③ Chapter 3: Software Construction（第 3 章：ソフトウェア構築）

- Software Construction Fundamentals（ソフトウェア構築の基礎）副知識領域に以下が追加されている。
  - Reuse（再利用）: 再利用は 2004 年版の「実践上考慮するべきことがら」副知識領域及び V3.0 の Practical Considerations（実践上考慮するべきことがら）にもあるが、構築の基礎としても考慮が必要とされている。
- 新しく Construction Technologies(構築の技術）副知識領域が追加され、API 設計とその利用というトピックスをはじめ 16 の技術項目を紹介している。

④ Chapter4 : Software Testing（第 4 章：ソフトウェアテスティング）

- Test Techniques（テスト技法）副知識領域に以下が追加されている。
  - Model-based（モデルベース）: 要求時の形式記述に対応するテスト技法として紹介している。

⑤ Chapter 5 : Software Maintenance（第 5 章：ソフトウェア保守）

- Techniques for Maintenance（保守のための技法）副知識領域に以下が追加されている。
  - Migration（マイグレーション）: ソフトウェアのライフ中に異なる環境で運用される場合があり、この場合の考慮事項を紹介している。
  - Retirement（除却）: ソフトウェアは必ず何処かでそれ以上運用しないという決定をするが、その決定に関連しての考慮事項を紹介している。

⑥ Chapter 6 : Software Configuration Management(第 6 章：ソフトウェア構成管理）

- 特に構成上の改訂は無い。

⑦ Chapter 7: Software Engineering Management（第 7 章：ソフトウェアエンジニアリング・マネジメント）

- 特に構成上の改訂は無い。

⑧ Chapter 8: Software Engineering Process（第 8 章：

ソフトウェアエンジニアリングプロセス）

― 副知識領域が全面的に改訂されている。新しい名称は以下の通りである。

・ Software Process Definition（プロセス定義）同じ用語が 2004 年版にも用いられているが、V3.0 では再構成され、この副知識領域では、プロセスそのものを定義している。

・ Software Life Cycles（ソフトウェアライフサイクル）副知識領域は、2004 年版の「プロセス定義」副知識領域に類似している。

・ Software Process Assessment and Improvement（ソフトウェアプロセスアセスメント及び改善）副知識領域は、2004 年版の「プロセス査定」副知識領域を拡張している。「ISO/IEC 15504　プロセスアセスメント」が広く理解され導入が進んでいることもあり、JIS 用語としては「プロセス査定」ではなく「プロセスアセスメント」になっている。

・ Software Measurement（ソフトウェア測定）副知識領域は、2004 年版の「プロセス及びプロダクト計量」に類似している。

⑨ Chapter 9 : Software Engineering Models and Methods（第 9 章：ソフトウェアエンジニアリングモデル及び手法）

― V3.0 にては、「ツール」はすべて各知識領域に移動したので、当章では、以下の副知識領域をカバーしている。

・ Modeling（モデリング）: ソフトウェアエンジニアにとり、対象をモデル化し関係者間で理解を共有することが重要になり、モデリングする際の考慮事項を説明している。

・ Types of Models（モデルの型）: 主として UML を適用する時のモデル図の紹介をしている。

・ Analysis of Models（モデルの分析）: ソフトウェアエンジニアが対象を種々のモデルを用いていかに有効に分析できるかを紹介している。

・ Software Engineering Methods（ソフトウェアエンジニアリング手法）: 2004 年版にアジャイルが追加されている。

⑩ Chapter 10 : Software Quality（第 10 章：ソフトウェア品質）

― 2004 年版の「ソフトウェア品質の基礎」副知識領域に、以下の新しいトピックス追加している。

・ Software Safety（ソフトウェアセーフティ）: ソフトウェアによる安全を考慮することはますます必要になってきており、考慮事項を紹介している。

⑪ 2004 年版の「第 12 章ソフトウェアエンジニアリングに関連するディシプリン」は拡張され、新しい知識領域として以下のように定義されている。

― Chapter 11: Software Engineering Professional Practice（第 11 章：ソフトウェアエンジニアリングプロフェッショナルプラクティス）
カバーする副知識領域は、３. ④を参照。

― Chapter 12 : Software Engineering Economics（第 12 章：ソフトウェアエンジニアリングエコノミックス）
カバーする副知識領域は、3. ⑥を参照。

― Chapter 13 : Computing Foundations（第 13 章：コンピューティング基盤）
実に多く以下の 17 の副知識領域を紹介している。

・ Problem Solving Techniques（問題解決技法）

・ Abstraction（抽象化）

・ Programing Fundamentals（プログラミング基礎）

・ Programming Language Basics（プログラミング言語基礎）

・ Debugging Tools and Techniques（デバッギングツールと手法）

・ Data Structure and Representation（データ構造と記法）

・ Algorithms and Complexity（アルゴリズムと複雑性）

・ Basic Concept of a System（システムの基本概念）

・ Computer Organization（コンピュータ構成）

・ Compiler Basics（コンパイラー基礎）

・ Operating System Basics（オペレーティングシステム基礎）

- Database Basics and Data Management（データベース基礎とデータ管理）
- Network Communication Basics（ネットワークコミュニケーション基礎）
- Parallel and Distributed Computing（並列及び分散コンピューティング）
- Basic User Human Factors（ユーザー人間工学基礎）
- Basic Developer Human Factors（開発者人間工学基礎）
- Secure Software Development and Maintenance（セキュリティを考慮したソフトウェア開発と保守）

— Chapter 14: Mathematical Foundations（第 14 章：数学基盤）

2004 年版第 12 章の「数学」副知識領域では、線形代数等 7 トピックスが列挙されているが、「集合等」以下の 11 の新副知識領域を紹介している。

- Set, Relations, Functions（集合、関係、機能）
- Basic Logic（ロジック基本）
- Proof Techniques（証明技法）
- Basics of Counting（カウンティング基礎）
- Graphs and Trees（グラフと樹）
- Discrete Probability（離散確率）
- Finite State Machine（有限状態機械）
- Grammars（文法）
- Numerical Precision, Accuracy, and Errors（数値精度、正確度、誤差）
- Number Theory（整数論）
- Algebraic Structures（代数構造）

— Chapter 15 : Engineering Foundations（ 第 15 章：エンジニアリング基盤）

2004 年版第 12 章の「コンピュータエンジニアリング」副知識領域における統計に関連するトピックス等一部類似のものもあるが、再構成され 7 つの副知識領域を紹介している。

- Empirical Methods and Experimental Technique（経験的方法と実験的技法）
- Statistical Analysis（統計的分析）
- Measurement（測定）
- Engineering Design（エンジニアリング設計）
- Modeling, Simulation and Prototyping（ モデリング、シミュレーション、プロトタイピング）

5 終わりに

既に翻訳されている 2004 年版が入手可能であることから、V3.0 の紹介にあたっては、特にどの知識領域が 10 年経過した今改訂されているかに言及した。ただし、今回の紹介にあたっては、主として副知識領域を示すトピックスレベルで 2004 年版と V3.0 を比較しているが、（一部はその下のトピックスレベルまで）各トピックスの詳細な内容及び参照している資料の調査まではできていない。2004 年版においても、V3.0 においても、各副知識領域で紹介するトピックスから参照資料を調べソフトウェアエンジニアリングの体系により深くアクセスすることを目的としており、V3.0 を理解し、現場での実践に活かすためには、参照資料を入手し読むことが必要になる。ここで問題は、参考資料がほとんど日本語に翻訳されていないことである。2004 年版から 10 年経過した現在では、インターネットの進歩により少なくとも参考資料の入手は以前より楽になったが、これら資料を読み V3.0 の全体像を理解するにはかなりの労力が必要なことには変わりはない。筆者は 2004 年版にて参照資料とされ、必要と思えるものの一部を入手したが、それでも机上に立てて並べると机一杯になった。残念ながらそれらをすべては読めていない。ソフトウェアが社会においてその重要性を増している現在、ソフトウェア開発にかかわるプロフェッショナルは V3.0 にて紹介されている知識領域をある程度身につけることが望まれ、大学におけるゼミ等、あるいは企業内の勉強会等で学習する必要があると考える。特に、V3.0 では 2004 年版から拡張された第 11 章から第 15 章に関しては、大型かつ複雑化するソフトウェアの開発にあたりある程度の基礎力の強化が必要になって来ていることを反映しているものと思われ、より多くのソフトウェア開発にかかわるプロフェッショナルのみなさんに V3 を先ずは読んでみようと感じていただければ小論の目的のほとんどは達成したことになると考える。

一般社団法人実践的プロジェクトマネジメント推進協会（PPMA）の紹介

「定量的プロジェクト管理ツール（EPM-X）」の利用促進をもとに、実践的プロジェクトマネジメントの普及を目指す。

一般社団法人実践的プロジェクトマネジメント推進協会（PPMA）代表理事
今井　元一

2012 年 7 月に発足した一般社団法人実践的プロジェクトマネジメント推進協会（PPMA※1）は、独立行政法人情報処理推進機構（IPA）が開発しオープンソースとして公開した「定量的プロジェクト管理ツール（EPM-X※2)」の普及活動を担う民間移管先として、2013 年 4 月より正式に認定されました。PPMA では普及セミナーの開催や導入コンサルティング、EPM-X の ASP サービスの提供など様々な活動を通じて、実践的・定量的なプロジェクトマネジメントの紹介・導入支援に努めている。

1 はじめに

独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）では、第二期中期計画（2008 年 4 月～ 2013 年 3 月）において､ソフトウェア開発プロジェクトの定量的な見える化を重点施策の一つとして取り組んできました。その成果として、定量的プロジェクト管理ツール「EPM-X」を開発し、2012 年 4 月にオープンソースとして公開しました。

EPM-X の公開は想定以上の反響をいただき、既に一万件を超えるダウンロード実績があります。中には小規模な範囲で試行をしたり、部門内での本格的な運用を開始した企業もあります。

2012 年 7 月に EPM-X の企画・開発に携わった有志により立ち上げられた PPMA は、EPM-X の利用促進を通じた実践的なプロジェクトマネジメントの普及／運用支援により、IT プロジェクトの健全化を実現し、我が国産業界の健全な発展と国民生活の向上に寄与することを目的としています。2013 年 4 月からは、IPA/SEC から正式に移管を受け、EPM-X の普及に関する多様な活動を実施しています。

2 定量的プロジェクト管理ツール（EPM-X とは）

EPM-X は、ソフトウェア開発プロジェクトにおいて、タスクの進捗、課題・障害の解決状況、工数の予実等を適切に把握するためのツールです。プロジェクト進捗過程で発生する定量的なデータを用いて、品質や進捗の状況をグラフィカルな画面を通じて定量的に把握できるようになっています。

その機能を利用して、リスクを可視化し、問題を早期に発見する定量的プロジェクト管理を実現することができます。

また、相対的に遅れているプロジェクト管理ツールの普及を促進するため次のような考慮が施されています。

- 一括インストーラの提供により深い技術スキルがなくとも容易に環境構築できる。
- 開発の現場でよく使われている実績あるオープンソースで構成しているため馴染みやすい。

  課題管理　：Redmine,Trac
  版管理　　：Subversion,Git

【脚注】
※ 1　PPMA: Practical Project Management Association
※ 2　EPM-X: Empirical Project Monitor Extended

BI ツール　：Birt

ETL ツール：Pentaho

EPM-X の機能上の特徴について、以下に説明します。

(1) グラフによる視覚的なレポーティング

EPM-X では、各定量データの収集・集計結果を、自動的にグラフ表示し、定量データによる状況の把握を視覚的に行えるようになっています。グラフの形状変化により、プロジェクトに異常が発生し始めた兆候を直感的に気づくことができるところが、高評価を得ています。

EPM-X が提供している多様なグラフを一覧の形で表 1 に示します。

表 1　表示グラフ一覧

グラフ名	概　要
1) WBS（タスク）・品質管理グラフ	
試験計画項目密度	試験項目のカバレッジを確認
試験進捗率	試験項目の進捗を確認
WBS 進捗推移	プロジェクトの進捗推移の把握
WBS 進捗変化	プロジェクトの生産性変化を把握
EVM 評価	EVM による生産価値把握と将来予測
ソフトウェア規模推移	ソースコードによる規模の推移、及び計画値との対比
工数の予実	開発工数の予実把握と完了工数の推定
遅延重要タスク抽出	進捗の遅れている WBS（タスク）を抽出
2) 障害・課題管理グラフ	
障害件数変化	障害件数、未解決数の推移、計画値との対比を把握
障害解決予測	障害の未解決数と解決生産性から、解決完了日を推測
障害原因分析	現在の障害の数を原因別に分類
障害発生密度	開発品質の把握
障害滞留状況	長期間解決されていない障害を抽出
長期未解決課題抽出	長期間解決されていない課題を抽出
3) 負荷管理グラフ	
負荷状況	開発グループ／開発者の負荷を把握

(2) チケット駆動開発

チケット駆動開発とは、作業をタスクに分割し、おのおのにチケットを割当て、チケットの状態をモニターしながら開発プロジェクトを進める開発スタイルのことです。

チケットには、「担当者」が割り当てられ、「作業中」「完了」などのステイタスを報告することによって、タスクの進捗状況が把握できます。EPM-X は、チケット駆動型開発のプロジェクト管理プラットフォームである Redmine や Trac を中核として構成されています。

(3) 定量データの自動収集

定量的なプロジェクト管理の普及を阻害している最大の要因として、データ入力・収集の手間があげられる。この対策として、EPM-X では、Redmine や Trac などのプロジェクト管理プロットフォームのチケットから進捗情報や障害情報を、Subversion や GIT などのバージョン管理ツールから変更情報を、自動的にデータとして収集することで、運用負荷の低減を図っています。

(4) データの可用性

定量データの可用性を高めるために、EPM-X では、チケットデータと Microsoft Excel、Microsoft Project、CSV 形式データとの相互変換機能を提供しています。また、グラフデータについても、PDF、Microsoft Word、Microsoft PowerPoint 形式の文書として保存が可能とし、報告書の作成に対する利便性向上を図っています。

(5) 柔軟性・拡張性

使用しているオープンソースのツールや環境は、普及度が高く一般的に使われているものを選んでいます。また、プログラム、データベース、インターフェイスも公開しているため、定量データ項目の追加やグラフの追加などを、比較的容易に自由に利用者が行えるようになっています。

3 PPMA の事業活動

PPMA では EPM-X の普及拡大、及びオープンソースとしての継続的提供を目指し次のような活動を行っています。

(1) EPM-X 広報・普及活動

PPMA では、次のような広報・普及活動を行っています。

① EPM-X 紹介のための無償セミナーの開催

2013 年 3 月以降 12 月までに 8 回実施。普及活動のために、IPA/SEC との共催を中心にほぼ 1 ヶ月に 1 回のペースで開催しており、今後も継続する予定である。

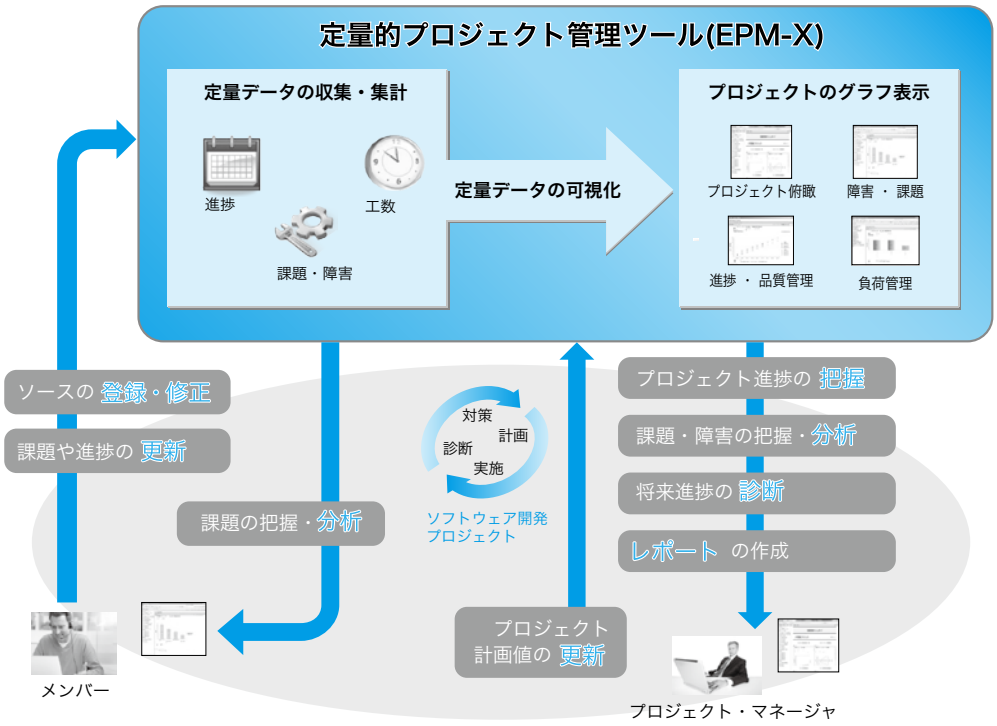


図1　システム概要図

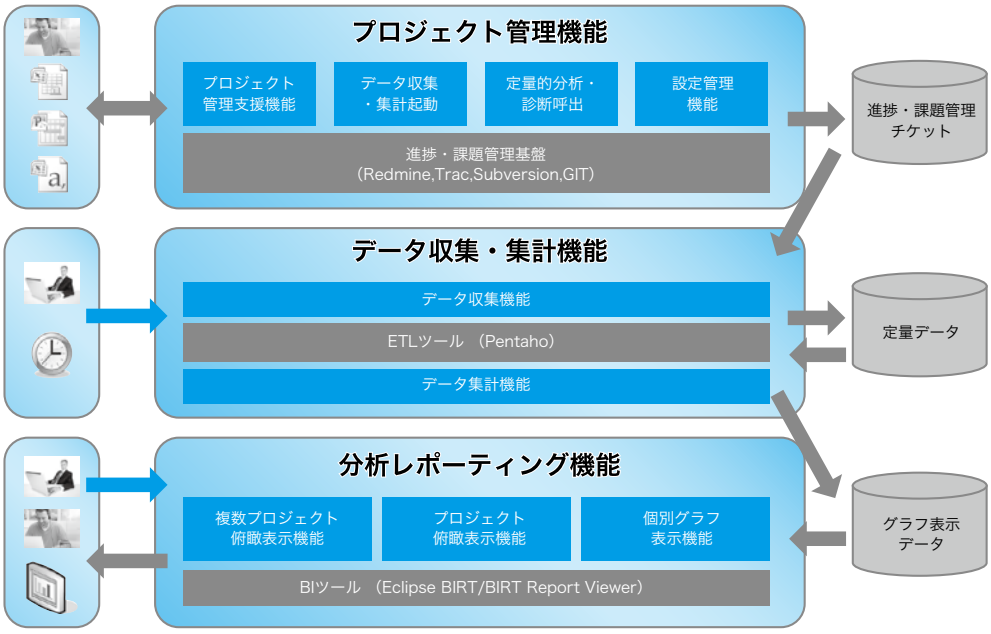


図2　ツール構成図

② 展示会への出展

組込み総合技術展（ET2012,ET2013）の IPA ブースやベンダーセミナーで EPM-X の紹介を行っています。

③ 企業、団体の個別訪問による紹介

企業・団体・勉強会などへの個別訪問を多数行い、定量的なプロジェクト管理の意義と EPM-X の紹介を行っています。

(2) EPM-X に関する技術開発の検討

利用者や EPM-X セミナー参加者の声を受け、PPMA では EPM-X の機能の改善について継続して検討しています。また、プロジェクトの定量的見える化の基本コンセプトに従った適用範囲の拡大についても重点課題とし

て捉え今後の技術開発計画に反映していきたいと考えています。

① プロジェクト管理ツールとしての機能強化

EPM-X を構成するミドルウェア、OS の最新バージョンへの対応に関しては継続的に検討を進めています。また、文書管理のサポートも上流工程のドキュメントを中心としたプロセスの管理に不可欠との認識から然るべきタイミングで機能強化を図りたいと考えています。

② 組込みソフトウェアを中心としたソフトウェア高信頼化への貢献

ソフトウェアの高信頼化に向けては開発段階でのレビューの実施状況や障害発生状況を監視・管理するとともに、出荷後／運用段階での障害情報の適切なフィードバック等の定量データにもとづく一貫した品質管理が重要です。また開発プロセスに起因する品質問題への対応を円滑に進めるためにもプロセスのパフォーマンスを定量的に表現し恒常的に改善を図る必要があります。EPM-X の基本的アーキテクチュアを品質管理やプロセス評価に適用することは極めて効果的であると考えています。

(3) EPM-X の ASP による提供サービスの開始

EPM-X の本格的な活用を計画している利用者からの強い要望に応えて、PPMA では ASP サービスによる EPM-X 利用環境の提供を 2013 年 8 月より開始しています。これは、サーバーやネットワークの調達、導入・運用のための技術者の確保等、EPM-X の利用開始に必要なハードルを下げることを目的としています。

単に環境を用意するだけではなく、準備作業から運用の支援までライフサイクルを通してのコンサルティングサービスも提供しています。

(4) PPMA サービスメニューの整備

PPMA や EPM-X への関心の高まりに対応するため次のような施策を開始しました。さらに 2013 年 10 月より PPMA のホームページを一新し、協会の活動の活性化に努めています。

① 会員制度の刷新

利用者のニーズに対応し情報発信や相互交流を活発化するため会員制度を改訂しました。会員は主にスタンダード会員（無償会員）、アドバンスト会員（有償会員）の 2 タイプとし、法人のみならず個人の参画を促進しています。また、この会員制度をもとに EPM-X コミュニティの形成をも目指してます。

② 情報発信の強化

PPMA ホームページの刷新と同時に当協会ホームページからも EPM-X をダウンロードできるサービスを設置しました。さらに、最新のサポート情報や EPM-X セミナー等での講演資料をアーカイブスとして公開したり、頻度の高い一般的な質問に対しては FAQ を整備し公開に向けて準備中です。

4 終わりに

PPMA は、発足してまだ 2 年目という若い協会ですが、オープンソースである EPM-X を通じて定量的なプロジェクトの見える化を積極的に推進していきたいと考えています。

本格的な会員募集を始めてから、まだ 3 ヶ月ほどですが、既に 170 名を超える会員の登録をいただいており、会員間の相互交流の中や、事例研究などを通じて、実践のための知識の共有化等に重点的に取り組んで行きたいと考えています。

また、国家プロジェクト TERAS をはじめとしてソフトウェアの高信頼化への貢献を目指す企業・団体と連携をしつつ、広汎な領域でのプロセスの定量的マネジメントの普及拡大に努めて行きたいと考えています。

PPMA は、オープンソースである EPM-X を利用した定量的なプロジェクト管理が日本のスタンダードとして広く日本の産業界に定着し、さらに、グローバルな市場も含めて、多様な領域におけるプロセスマネジメントへの適用を目指して活動して行く所存です。

国立情報学研究所 トップエスイー

国立情報学研究所 アーキテクチャ科学研究系 特任教授
田辺　良則

http://www.topse.jp/

国立情報学研究所では、先端的なソフトウェア工学の知識を持ち、実問題への適用ノウハウを身につけた、スーパーアーキテクトを育成するため、2004 年度より IT 技術者教育プログラム「トップエスイー」を運営している。モデリング能力の向上を重視したプログラムで、230 名以上の育成実績をあげている。

1 ソフトウェア技術者育成

近年、ソフトウェアの重要性は以前にもまして高まっており、大規模で複雑なシステムを、堅牢に、しかも短期間に構築することが期待されている。しかし、技術者の多くは、依然として経験と勘に依存したソフトウェア作りに頼っており、科学的アプローチが不足している。一方、ソフトウェア工学研究においては、開発現場で研究結果を実践・検証する機会が十分でない。

このような状況を改善し、ソフトウェア開発現場に最新の研究成果を導入して人材育成を行うべく、国立情報学研究所では、2004 年に IT 技術者の育成のための教育プログラム「トップエスイー」プロジェクト（代表：本位田真一 NII 副所長）を開始した。「サイエンスによる知的ものづくり教育」を掲げ、トップレベルのエンジニアを育成することに主眼をおいている。約 2 年間の教材開発期間の後、2006 年度より実際の育成を開始し、現在までの 8 期（各期は 1 年から 1 年半）にわたる育成実績は約 230 名（図 1）、現受講生は 40 名強である。なお、プロジェクト開始時には、文部科学省科学技術振興調整費（2004-2008 年度）の補助を受け、2009 年度からは NII の自主事業として実施している。

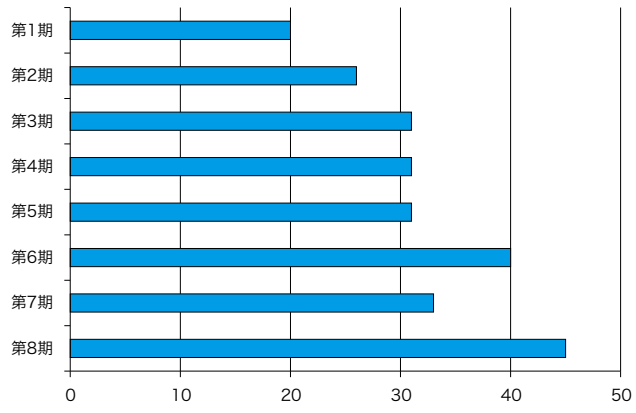


図 1　受講者数の推移

トップエスイーが目指す人材像は、ソフトウェア工学や計算機科学からの知見を十分に活用して問題解決を行える、トップレベルのソフトウェア技術者である。このためにモデリング能力をもっとも重視している。各工程において適切な抽象度を持つモデルを構築し、分析、判断、変換を繰り返しながらモデルを詳細化、洗練化させていくことになるからである。講義と後述する修了制作において、様々なツールや手法の活用を通してこの能力を養成する。単に従来の経験に基づいてソフトウェアを作るのではなく、新しい案件に潜在する問題を発見したり解決したりする力のある人材を育てている。

2 カリキュラム・教材

講義は、プロジェクトマネジメント、要求工学、アーキテクチャ、モデル検査、形式仕様記述、クラウドの 6 つの専門科目群に編成されており、各科目群をコースと呼んでいる。このほか、より基礎的な内容を持つ共通科目群がある。 1 つないし 2 つのコースの科目を集中して履修するように受講生には推奨しており、そのコースに関連した修了制作を実施することを想定しているが、意欲のある受講生は複数のコースの科目を履修し、コースをまたがった修了制作を行うことも可能である。

教材開発は産業界、学界で活躍する研究者・実務者によって行われている。産学双方の出身者の議論により、最新の研究成果を反映しつつ、開発現場で求められる内容を指導できるよう、教材を精錬するプロセスを取る。主として教材開発に携わったメンバーが講師となって授業を行う。現在の講師数は 45 名で、うち 6 割が産業界、残り 4 割が学界に属している。講義の構成では、単なる座学による勉強にとどまらずに実践を行うことを重視し

写真１　グループ演習

写真２　ロンドン大学との共同短期集中演習

ている。科目の特性に応じて、グループ演習（写真１）による受講生同士や講師との議論や、中規模の例題によるプログラム作成などの経験を通して、生きた知識の体得を図る。企業内での技術普及方法を考えさせたり、著作権に関する模擬裁判を実施したりするユニークな講義もある。

開発した教材の普及にも力を入れている。NII が運用する、ソフトウェア工学の講義を配信するウェブサイト edubase Stream[※1] において「トップエスイーチャンネル」を持ち、月替わりで講義を公開している。

また、近代科学社より「トップエスイー入門・基礎・実践講座」シリーズとして、これまで９冊の教科書を刊行し、なお続刊予定である。

3 受講システム

受講期間は原則として１年間であり、４月から翌年３月までである。受講生が業務をこなしつつ学べるように、講義は平日夕方と土曜日に行われる。平日は２コマ（1コマ＝90分）、土曜日は４コマである。修了に要する単位を 12 単位（15 コマ＝２単位）に設定しており、受講生は自分の業務上の必要や興味に基づいて科目を選択する。2014 年度の時間割を図２に示す。年間を２ヶ月ずつの４学期に分割し、残りの期間は、集中講義や「開講前講義」と称する基礎的な科目にあてている。

修了するためには、講義履修のほか、「修了制作」と呼ばれる卒業研究・修士研究に相当する研究を実施してレポートを提出し、審査に合格することも必要である。修了制作においては、受講生自身の問題を、トップエスイーで学んだ手法を用いて解決をはかる。期間中は担当講師が一対一で指導を行う。

受講生の多くは IT 技術者であり、入社 5-10 年目程度の者が多い。所属企業は、メーカー、システム・インテグレーター、ユーザ企業など多岐にわたっている。ソフトウェア開発を現に行っている者が多いが、ドメインは組込み系、エンタープライズ系、ウェブ系など分散している。特定の領域に特化した知識ではなく、モデリング能力に代表される一般的な力を伸ばす教育を行っていることを反映している。

受講生を継続的に送っているいくつかの企業では、トップエスイーの科目を自社の教育コースの中に組込んだり、トップエスイーの修了をキャリアパス上の資格の１つとして認定するなどの活用をしている。

4 トピックス

近年のトピックをいくつか紹介する。

(1) ロンドン大学との連携

英国ロンドン大学（UCL）と NII は長く連携関係にある。計算機科学学部長のフィンケルシュタイン教授が、トップエスイープロジェクトを高く評価しており、2011 年度以来毎年、トップエスイー受講生・修了生と UCL の学生による共同短期集中演習を実施している（写真２）。現

【脚注】

※１　http://stream.edubase.jp/

表 2：2014 年度トップエスイー講義時間割

<table>
<tr><th></th><th>月</th><th>火</th><th>水</th><th>木</th><th>金</th><th colspan="2">土</th></tr>
<tr><td>開講前</td><td>要求工学入門</td><td>基礎理論</td><td>ソフトウェア
工学入門</td><td></td><td></td><td colspan="2"></td></tr>
<tr><td rowspan="2">1 学期
(4-5 月)</td><td rowspan="2">PM 概論</td><td rowspan="2">テスティング
基礎</td><td rowspan="2">形式仕様記述
(基礎・VDM)</td><td rowspan="2">設計モデル
検証 (基礎)</td><td>セキュリティ
概論</td><td>クラウド
入門</td><td></td></tr>
<tr><td>構造化分析法</td><td>クラウド
実践演習</td><td>ソフト開発
見積り手法</td></tr>
<tr><td rowspan="2">2 学期
(6-7 月)</td><td rowspan="2">リスク
マネジメント</td><td rowspan="2">コンポーネント
ベース開発</td><td rowspan="2">形式仕様記述
(B メソッド)</td><td>業務アプリ向け
シナリオ分析</td><td rowspan="2">設計モデル
検証 (応用)</td><td colspan="2">分散処理アプリ
演習</td></tr>
<tr><td></td><td colspan="2">モデル検査事例
演習</td></tr>
<tr><td>夏期集中
(8 月)</td><td colspan="3">モデル駆動開発</td><td colspan="3">定理証明と検証</td></tr>
<tr><td rowspan="2">3 学期
(9-10 月)</td><td rowspan="2">PM
支援ツール</td><td rowspan="2">ソフトウェア
パターン</td><td>形式仕様記述
(Event-B)</td><td rowspan="2">並行システム
の検証と実装</td><td rowspan="2">実装
モデル検証</td><td>分散システ
ム基礎</td><td>アジャイル
開発</td></tr>
<tr><td>ゴール指向分析</td><td>形式仕様
記述 (実践)</td><td></td></tr>
<tr><td rowspan="2">4 学期
(11-1 月)</td><td>性能
モデル
検証</td><td>SW
設計法
通論</td><td rowspan="2">アスペクト
指向開発</td><td>問題
指向
要求
分析</td><td>テス
ティン
グ応用</td><td rowspan="2">プログラム
解析</td><td>安全要
求分析</td><td>SW
メトリ
クス</td><td rowspan="2">クラウド
基盤
構築演習</td><td>SW 再利用
演習</td></tr>
<tr><td colspan="4"></td><td colspan="2"></td><td>SW 保護と著
作権</td></tr>
<tr><td>冬期集中
(1 月)</td><td colspan="3">オブジェクト指向分析法</td><td colspan="3">概念モデリング</td></tr>
</table>

在まで、組込み開発、テスティング、プロジェクト管理等をテーマに各 1 週間の集中演習を行ってきた。2014 年度以降は、分散開発を念頭により長期間の演習を実施する予定である。

(2) 遠隔受講制度

従来の受講生は、事実上東京近郊に勤務する人に限られていた。関西等の技術者の受講を可能にするため、2013 年度より遠隔受講システムを導入した。遠隔受講者は、インターネットを通して教室における講義（動画配信）と教材にアクセスでき、双方向の通信により質問等を行うこともできる。配信対象の講義履修によって、修了するために必要な単位を取得することができる。修了制作もインターネット会議システムを活用し、年数回の出張による打合せと組み合わせることで、在京の受講生に比べても遜色のない成果を得ている。

(3) 大学院との連携

NII は、電気通信大学、北陸先端科学技術大学院大学、情報セキュリティ大学院大学等と協定を締結し、協力関係を結んでいる。この関係を利用し、学位の取得を望む受講生に、修了制作の結果を更に発展させて大学院での博士研究とすることを推奨している。修了制作の指導にあたった講師が、引き続いて学位取得まで協力を行う体制を取っている。

5 おわりに

トップエスイーは、先端技術を学ぶ意欲のある技術者にとって、魅力的なプログラムとなっている。受講期間の 1 年間のみではなく、修了後も継続して新しい技術を身につけたり、受講によって得られた他社の技術者や産学両分野の講師との人脈を広げていくことを重視し、修了生が参加できる勉強会やセミナーの開催、交流会の実施など様々な活動を行っている。これらを通して、今後も受講生・修了生のコミュニティを育てていきたいと考えている。

第11回クリティカルソフトウェアワークショップ（11thWOCS^2）開催報告

独立行政法人宇宙航空研究開発機構（JAXA）
情報・計算工学センター ソフトウェアエンジニアリングチーム
大久保　梨思子　　氏家　亮
独立行政法人情報処理推進機構（IPA）
技術本部 ソフトウェア高信頼化センター（SEC）企画グループ
荒川　明夫

1. 開催概要

クリティカルソフトウェアワークショップ（WOCS^2）は、独立行政法人宇宙航空研究開発機構（JAXA）と独立行政法人情報処理推進機構（IPA）が共催する、宇宙・航空、自動車などのミッションクリティカルなソフトウェアの開発・運用・保守に関する技術やプロセスに焦点を当てたワークショップである。WOCS^{2}は2002年から開催しており、2009年からはIPAとの共催で産、学、官の枠をも超え、ソフトウェアシステムの安全についての議論の場を提供している。

第11回を迎えた今回のWOCS^{2}では、「SecurityとSafetyを融合させたシステムを考える」を主テーマとして掲げ、2014年1月15日（水）から1月17日（金）の3日間、御茶ノ水ソラシティカンファレンスセンターにて開催した。

2. 専門セミナー

1/15には専門セミナーとして、ソフトウェアの安全性と信頼性を確保するための技術に関するセミナーを開催した。このような専門セミナーの開催はWOCS^{2}において初めてである。今回はJAXAが実施するソフトウェアIV&V（独立検証及び有効性確認）の基礎について講義を行った。

ソフトウェアIV&Vとは、ソフトウェアのVerification（検証）とValidation（妥当性確認）をIndependent（独立）に実施することであり、JAXAでは1990年代から実施してきた。平成25年にはIPAにより「通称：ソフトウェア品質説明のための制度ガイドライン」が整備され、製品やシステムの品質説明力強化のために、供給者から独立性を確保した第三者による認証活動が注目されている。

セミナーではまず、IPA鈴木基司、JAXA片平真史からIV&Vを導入することの意義や効果について講義が行われた。IV&Vは決して網羅的なバグだしではなく、重大な技術的欠陥を早期に抽出することを目的としており、適切な評価観点／手法／範囲を選択することの重要性について説明を行った。

次に、前述の評価観点／手法の具体例について、JAXA梅田浩貴と川口真司から「IV&Vガイドブック【虎の巻】」（JAXA発行）を用いた講義が行われた。演習ではセンサー付き宇宙用便座のシステム仕様を題材とし、評価観点の詳細やV&VとIV&Vの違いについて説明が行われた。

（http://www.ipa.go.jp/sec/events/20140115.html で講義資料を公開中。また、JAXA講義資料及びIV&Vガイドブックの入手については IVV_INFO@jaxa.jp まで。）

3. 基調講演・招待講演

1/16にはソフトウェアの安全性、信頼性、セキュリティに関する基調講演、招待講演が計6件行われ、約150名が聴講した（表1）。

【基調講演】では、三菱航空機株式会社の篠田 和英氏、技術研究組合制御システムセキュリティセンター（CSSC：Control System Security Center）／電気通信大学の新 誠一氏、株式会社ソニーコンピュータサイエンス研究所の所 眞理雄氏にご登壇いただいた。

篠田氏からは、高い安全性が要求される民間航空機において、発生確率が算出できる物理的な故障だけでなく、発生確率が定義できないソフトウェア開発中のエラーなどへの対策が重要視される現状についてご説明いただいた。更に、その様な対策を実現し、安全性を担保するためには安全・開発保証プロセスが重要であり、Mitsubishi Regional Jet（MRJ）の開発で如何に安全・開発保証プロセスに取り組んでいるかを型式証明の話を交えてご講演いただいた。

表１：基調講演／招待講演のプログラム

10：00 〜 10：10	開会挨拶 独立行政法人 宇宙航空研究開発機構 執行役　井澤一郎
10：10 〜 10：40	**オープニング講演** 独立行政法人 情報処理推進機構 ソフトウェア高信頼化センター　所長　松本隆明
10：40 〜 11：30	**民間航空機の安全・開発保証プロセスについて 〜 MRJ 開発における取り組み〜** 三菱航空機株式会社 技術本部　開発保証部　部長　篠田和英
11：30 〜 12：20	**Safety Assessment Method for Flight Operation System. 〜 Lessons from "RNP AR approaches" to Haneda Airport.** DNV ビジネス・アシュアランス・ジャパン株式会社 代表取締役社長　前田直樹
13：40 〜 14：30	**制御系セキュリティの国内での取り組み** 技術研究組合制御システムセキュリティセンター 理事長　新誠一
14：30 〜 15：20	**組込みシステムのセキュリティ対策 〜繋がる車でのケーススタディ〜** 独立行政法人 情報処理推進機構 技術本部 セキュリティーセンター 情報セキュリティ技術ラボラトリー主任 中野学
15：40 〜 16：30	**ハイブリッド認証に向けての工学的アプローチ 〜機能安全とセキュリティの同時認証のための方法論〜** 独立行政法人 産業技術総合研究所 セキュアシステム研究部門システムライフサイクル研究グループ招聘研究員 田口研治
16：30 〜 17：20	**DEOS：巨大・複雑で変化し続けるシステムのディペンダビリティーを達成する** 株式会社ソニーコンピュータサイエンス研究所 エグゼクティブアドバイザー / ファウンダー　所眞理雄

（篠田和英氏）


（新誠一氏）


（所眞理雄氏）

新氏からは、近年制御系システムでセキュリティ対策が何故重要視されているか、対策の実現が何故難しいかについて、社会インフラなどでのセキリティ攻撃実例を交えてご説明いただいた。更に、制御系システムのセキュリティ問題への対策を検討するために CSSC が開発した様々な模擬システムのご紹介、CSSC が展開する評価認証・標準化活動のご紹介、今後の制御系システムのセキュリティ対策の展望についてご講演いただいた。

所氏からは、現代のコンピュータシステムは、機能、構造、システム境界が時間的に変化しつづけるシステム（オープンシステム）であり、その変化への対応、潜在的な障害の除去、説明責任の達成が非常に困難である現状についてご説明いただいた。更に、現状を打破するために検討された知識・技術体系である「オープンシステムのためのディペンダビリティ工学（DEOS：Dependability Engineering for Open Systems）」の検討過程、各検討段階で議論された課題、手法やツールを含む具体的な課題解決方法について、応用事例を交えてご講演いただいた。

【招待講演】では、DNV ビジネス・アシュアランス・ジャパン株式会社の前田 直樹氏、情報処理推進機構の中野 学、株式会社シーエーブイテクノロジーズ／独立行政法人産業技術総合研究所の田口 研治氏にご登壇いただいた。

前田氏からは、セーフティクリティカルなシステム開発で重要な役割を果たす安全性評価について、羽田空港 D 滑走路でのセーフティアセスメントの経験を基に、効果的な安全性評価の実現についてご講演いただいた。中野からは、組込み機器におけるセキュリティの現状とセキュリティ分析手法について、自動車を例としたご講演をいただいた。田口氏からは、現在の製品・プロセス認証の問題点を踏まえ、安全性とセキュリティの規格の同時認証の重要性とその実現方法についてご講演いただいた。

（基調講演、招待講演ともに、http://www.ipa.go.jp/sec/events/20140115.html で講演資料を公開中）

4. 一般講演

1/17 には一般講演として、様々な産業分野の企業、大学、研究所から全 18 件の安全及びセキュリティに関するご講演をいただき、約 130 名が聴講した。

（http://www.ipa.go.jp/sec/events/20140115.html で講演資料を公開中）

■ 11thWOCS^2 一般講演受賞者

安全に関するセッション（Safety セッション）での優秀賞の受賞者を表 2 に、セキュリティに関するセッション（Security セッション）の受賞者を表 3 に示す。

表 2：Safety セッション受賞者

Safety セッション	
最優秀賞	**「CARDION: 概念段階におけるハザード・脅威の抽出手法」**伊藤　昌夫（株式会社ニルソフトウェア）
優秀賞	**「イプシロンロケット搭載ソフトウェアによる安全設計の実装について」**井上　知也（株式会社 IHI エアロスペース）
	「JAXA-PAM と CMMI の補完的活用によるプロセス評価の効率化」込山　博（日本電気株式会社）

表 3：Secuity セッション受賞者

Security セッション	
最優秀賞	**「組込み開発におけるセキュリティ対策及び試験手法に関する一考察」**平井　康雅（株式会社 NTT データ）
優秀賞	**「著作権保護システムの車載セキュリティ応用と Safety への展開について」**倉内　伸和（パナソニックアドバンストテクノロジー株式会社）
	「形式手法を用いた安全・セキュリティ分析手順と要件の抽出方法の提唱」和田　学（株式会社ヴィッツ）

Safety セッションでは、宇宙機をはじめとした、高い信頼性が必要なシステム・ソフトウェアの開発や、安全性・信頼性向上のための取り組みについて講演があった。最優秀賞の伊藤氏は、開発の初期段階（概念段階）でハザード・脅威を抽出する手法について、車載ソフトウェアを例に講演された。

Security セッションでは、安全性とセキュリティを同時に確保することの難しさや相互の依存関係について多くの講演があった。最優秀賞の平井氏は、セキュリティが安全を脅かす 1 要素であると捉え、新たな枠組みを持つセキュリティ試験手法への取り組みについて講演された。従来、安全とセキュリティは別分野として発展してきたが、平井氏の講演は今回の WOCS^2 で掲げた「Security と Safety を 融合させたシステムを考える」というテーマに合致した講演であった。

5. SEC journal 論文書表彰式

1/17 の一般講演の後、WOCS^2 会場内にて平成 25 年 SEC journal 論文賞の受賞論文の発表を行った。最優秀賞を受賞した酒井氏の「アプリケーション保守サービスの定量化手法」についての論文は、従来見積もることが難しかった保守サービス量のコストや工数を独自の手法を用いて数値化し、これを可能にした点が高く評価された。

表 4：受賞者 3 編

SECjournal 論文賞	
最優秀賞	**「アプリケーション保守サービスの定量化手法」**酒井　大（日本アイ・ビー・エム株式会社グローバルビジネスサービス）
優秀賞	**「システム価値向上を目的とした Scrum の試行・評価」**中村　伸裕（住友電気工業株式会社 / 大阪大学）
所長賞	**「若年技術者向けソフトウェア開発研修プログラムの開発と評価」**大森　久美子（NTT　サービスイノベーション総合研究所　ソフトウェアイノベーションセンタ）

（SEC journal については、下記 WEB ページをご覧ください。

http://www.ipa.go.jp/sec/secjournal/index.html）

6. 今後の WOCS^2 について

今回の WOCS^2 では、昨年のアンケートを踏まえ、過去の WOCS^2 で扱っていない航空機分野の基調・招待講演を行い好評であった。今回もアンケートを通して多くのご意見をいただいたので、次回 WOCS^2 の基調・招待講演を含む全体運営に反映していく予定である。また、IV&V 専門セミナーについては、同内容の開催やより詳しい研修の要望が多く、新たなセミナーを企画していきたい。

次回 WOCS^2 については、2014 年 6 月以降に開催告知を行う予定である。

7. 謝辞

WOCS^2 プログラム委員の皆様、後援団体の皆様にはワークショップ成功のためにご支援いただきました。ここに深謝いたします。

Column

「変化の時代」

IPA 顧問 学校法人・専門学校 HAL 東京 校長
鶴保　征城

昨年を表す漢字は「輪」であったが、最近の状況をまとめて表すと「変」ではないかと思う。これは、「変な」ということもあるが、「変化」の「変」だ。ダーウィンの「最も強い者が生き残るのではなく、最も賢い者が生き延びるのでもない。唯一生き残ることができるのは、変化できる者である」という言葉を、いまこそ思い起こすべきだ。

大手電機メーカーの凋落はこの変化に対応できなかった結果だ。どのように対応できなかったというと、二つの面がある。一つは、対応できたプロセスが偏っていたということ。

およそ商品化には、コンセプト、ストーリー、デザイン、製造（ソフトの場合はプログラミング）、マーケティング、プレゼンテーションなどのプロセスがあるが、変化はすべてのプロセスで起こる。大手電機メーカーは製造プロセス、つまり技術の変化には対応できたが、他のプロセスの変化を見逃したと思う。この根底には、ユーザーである生活者自身が日々進化していることを、忘れがちであったということだ。

もう一つは、真の意味で変化を受信する体質になっているかどうかだ。「変化への対応が大事」ということは、かなりの以前から多くの企業トップが指摘していたが、「変化を前提としなければ滅びる」、とまで腹をくくっていた企業は少ないのではないか。膨大な情報から自分たちに都合のよい部分だけを受信しているのでは、変化を受信し対応していることにならない。

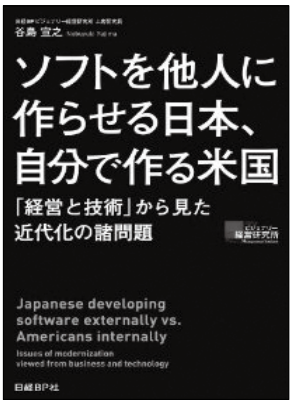
「商品が充足している」時代だということも、変化をドライブしている。この時代に必要なことは、単に作れるものを作るのではなく、コンセプト→ストーリー→デザインに磨きをかけることだ。そのためには、未来に対する理想、志、夢といったものが欠かせない。物ではなくストーリー（物語）を語る。技術者には苦手なところだと思うが、ここをブレークしなければならない。

大企業が生き延びるためには、小さな夢の実現に努力している社員を大事に育てることだ。社内だけでなく外部との連携（M&A）も積極的に行えばよい。M&Aでよく言われるのは、企業体質が合うか合わないかということだが、合えばよいが、たとえ合わなくてもそこから新しい企業生き残りの芽がでたと考えればよい。

我々は、入念に準備して 100 の規模の事業を作ろうとしがちだ。今の時代、これは無理がある。それよりも、1 の規模の事業を 100 作るほうが楽だ。社会も企業も個人もすべての仕組みやマインドが、100 の規模の事業を行うようになっているが、これを変えなければならない。

変化はあらゆる面で起こっている。エネルギー、メディア、高齢化、グローバル化、アジア進出、女性社会参画、大都市集中など。虚心坦懐に変化を受信し、分析しなければならない。更に詳しく見ていくと、また変化が起こっている。かつてのような安定というものは存在しない。

安定しないということが安定である、ということを体得しなければならない。



ソフトを他人に作らせる日本、自分で作る米国

谷島 宣之　著

ISBN: 978-4-8222-7378-1
日経 BP 社刊
四六判・412 頁
定価 1,500 円（税別）
2013 年 12 月刊

30 年にわたる IT 記者の経験から思うこと

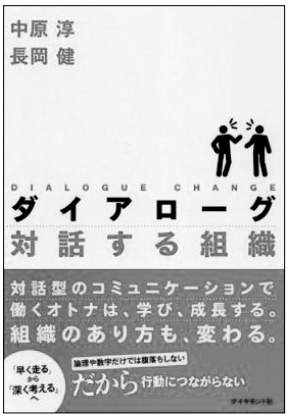
最近フェイスブックで情報を入手することが多いが、この本もそうである。書籍情報注文サイトを見て翌日には自宅に配達されている。この本は、大学卒業後 30 年近くも IT の技術動向、ベンダー、ユーザーの技術者、経営者と接してきた谷島氏にしか書けないトピックスが満載である。まずは書評者が一読した後に読者に目を通していただきたいと思ったものを以下に目次の章番号で列記する。もちろんこれは書評者の思いであり、皆さんにはどこから読んでもらってもさしつかえはない。この本は何回も読み直したくなるものである。

目次は 2 段落なので、その章の何番目のトピックスかを示し、特定のトピックスにジャンプできるように 3 段落にて紹介している。

1-5-3：Collaborative は「協調性」ではない
2-1-2：手段に集中、目的を忘れる
2-6：あえて労働集約
2-6-2：隣の事に気がまわらず
3-1-1：普通の日本語で考えよう
3-3-1：我々は情報責任を果たしているか
3-4：数値目標の暴走をどう防ぐか
4-4：「システム内製」こそ理想の姿
4-5：“ 技術の暴走 ” に社長が備えておくこと

谷島氏も述べておられるが、あるトピックスに目が行った場合、関係する参考文献を読みたくなる。この本にはそれが多くしめされており、より深く背景等勉強をする場合にも有効であろう。

（新谷勝利）



ダイアローグ 対話する組織

中原 淳、長岡 健　著

ISBN: 978-4478005675
ダイヤモンド社刊
単行本・224 頁
定価 1,600 円（税別）
2009 年 2 月刊

『対話という学びのコミュニケーション方法』

ソフトウェア開発を取り巻く環境は大きく変化している。ビジネス、自社やアライアンス含む組織、個人の仕事や生活など、環境は絶えず変化している。ソフトウェア開発に必要な知識を学び、スキルをつけること、そして組織活動の方向性やノウハウを共有し徹底することは難しくなっている。

本書は、学びや組織としての考えや知識の共有には、対話（ダイアローグ）が有効であると説く。これまでは知識や思いを一方的に説明すること、○○ウェイとしてスローガンを掲げることやストーリーテリングで伝える方法も行われてきた。しかしお客様の求めるものが多様化するだけでなく、組織のメンバーの価値観も多様化しており、これまでの情報伝達方法では期待する成果が得られないことが多くなっている。

こんな時代には価値観や立場の違いを前提として、リラックスした環境での対話が有効である。雑談でも議論でもない対話である。

この対話の方法は簡単そうで、なかなか難しい。リラックスした環境で、互いの違いを理解しつつ、話をしなければならない。特に日本人は自分のことについて、照れずに説明し主張することが苦手である。しかし、対話を通じた組織活性化については、日本企業の事例が多く紹介されている。ソフトウェア開発のプロセス改善において、この対話という方法は非常に有効だと感じた。私がまたプロセス改善に従事することがあれば、対話を重視した改善活動をしたい。

本書の最後には対話による新たな学びとして、学びのサードプレイスについて紹介されている。近年は、技術者コミュニティや IT 勉強会が多くなっている。このようなインフォーマルでパブリックな学びの場が、これからのソフトウェア開発を高度化させてくれると期待している。

（渡辺登）

編集後記

今回発行の 36 号では、2 つの所長対談をお届けしています。

1 つ目は、国立情報学研究所（NII）副所長 本位田真一氏との対談で、次世代を担うソフトウェア・エンジニア像とその育成についてお話いただき、その具体的な活動である「トップエスイー」については、‘組織紹介’にて、その内容や仕組み、実績などを紹介しています。

2 つ目は、米国マサチューセッツ工科大学教授ナンシー・レブソン氏とコンピュータの発展と普及によりシステムが複雑になった今日、なぜ障害が発生するのか、システムやソフトウェアの障害分析や原因を導き出す方法など先進的な手法と適用事例を含めて教授の著書をベースにお話をお伺いしました。ナンシー教授の考えや、今後の取り組みなど対談で分かりやすく語られています。

さらに、ナンシー教授には対談後に特別セミナー「Engineering a Safer World」で講演いただきましたので‘トピックス’にて概要を紹介しています。

なお、これらの講演資料や動画は、IPA/SEC の Web で公開していますので、こちらもご覧下さい。

（編集長）

編集部より

次世代のソフトウェア・エンジニアリング等に関して、忌憚のない意見をお待ちしております。下記の FAX またはメールにてご連絡ください。

SEC journal 編集部　　FAX：03-5978-7517　　e-mail：sec-journal_customer@ipa.go.jp

SEC journal　編集委員会

春の兆し　　（撮影：k-endou）

SEC journal® 第 10 巻 第 1 号（通算 38 号） 2014 年 3 月 31 日発行

編集兼発行人　独立行政法人情報処理推進機構
技術本部　ソフトウェア高信頼化センター
所長　松本隆明
〒 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
URL：http://www.ipa.go.jp/sec/
e-mail：sec-journal_customer@ipa.go.jp

SEC journal 論文募集

独立行政法人情報処理推進機構（IPA） 技術本部 ソフトウェア高信頼化センターでは、下記の内容で論文を募集しています。

論文テーマ

・ソフトウェア開発現場のソフトウェア・エンジニアリングをメインテーマとした実証論文または先導的な論文
・ソフトウェアが経済社会にもたらす革新的効果に関する実証論文

論文分野

品質向上・高品質化技術、レビュー・インスペクション手法、コーディング手法、テスト／検証技術、要求獲得・分析技術、ユーザビリティ技術、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、組織経営、イノベーション

応募要項

締切り　：1月・4月・7月・11月　各月末日
査読結果：締切り後、約1カ月で通知。「採録」と判定された論文は SEC journal に掲載されます。
応募方法：投稿は随時受付けております。応募様式など詳しくは HP をご覧ください。
http://www.ipa.go.jp/sec/secjournal/papers.html

SEC journal 論文賞

毎年「採録」された論文を対象に審査し、SECjournal 論文賞として最優秀賞、優秀賞、所長賞を表彰し、それぞれ100万円、50万円、20万円を副賞として贈ります。

ITパスポート試験 のご案内

ー ビジネスにＩＴを活用する すべての社会人のための「国家試験」ー

- ビジネスにＩＴを利活用するためには、情報システム部門に限らず、利用する側の社員一人ひとりにも“ＩＴ力”が求められています。
- ｉパス（ＩＴパスポート試験）は、セキュリティ、ネットワーク等のＩＴに関する基礎知識をはじめ、企業活動、経営戦略、会計や法務、プロジェクトマネジメントなど、幅広い総合的知識を測る国家試験です。
- ｉパスを通じて、社員一人ひとりに“ＩＴ力”が備わることにより、組織全体の“ＩＴ力”が向上し、様々なメリットが期待されます。

ｉパスのメリット

ＩＴを活用した業務効率化とビジネス拡大に！

ｉパスを通じて習得したＩＴの基礎知識を活かすことで、業務にＩＴを積極的に活用し、業務効率化につながります。また、ＩＴに関する基礎知識は、社内の情報システム部門等との円滑なコミュニケーションにも役立ちます。営業職であれば、顧客に対して製品やサービスを具体的にわかりやすく説明できるようになり、顧客のニーズをより深く把握できるようになり、ビジネスチャンスの拡大にもつながります。

情報セキュリティ対策・コンプライアンス強化に！

社員一人ひとりが、情報セキュリティやモラルに関する正しい知識を身につけ、意識することで、情報セキュリティに関する被害を未然に防ぐことができ、「情報漏えい」などのリスク軽減、企業内のコンプライアンス向上・法令順守に貢献します。

経営全般に関する知識など幅広い知識がバランスよく習得できる！

ｉパスは、ＩＴに関する知識にとどまらず、企業活動、経営戦略、会計や法令など、ＩＴを活用する上で前提となる幅広い知識がバランスよく習得できます。そうした知識が身につくことにより、業務の課題把握と、ＩＴを活用した課題解決力が備わり、組織全体の業務改善につながります。

詳しくは、ｉパス Web サイトをご覧ください。https://www3.jitec.ipa.go.jp/JitesCbt/index.html
※企業の活用事例、企業の声、合格者の声など魅力的なコンテンツがご覧になれます。

SEC journal No.36

IPA 独立行政法人 情報処理推進機構
技術本部 ソフトウェア高信頼化センター

SEC jounal No.36
第 10 巻第 1 号（通巻 38 号）
2014 年 3 月 31 日発行

ISSN 1349-8622

独立行政法人情報処理推進機構