IoT時代のソフトウェア・エンジニアリングについて考える

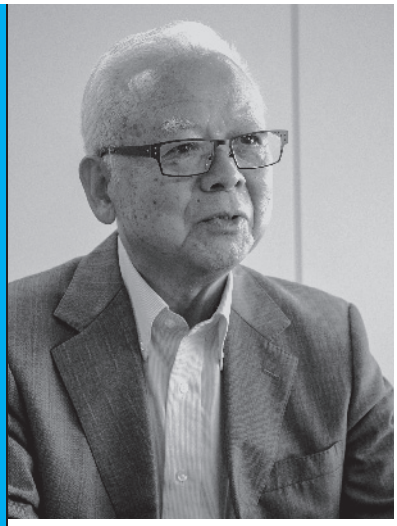
北陸先端科学技術大学院大学／東京工業大学
名誉教授
片山 卓也

IPA/SEC所長
松本 隆明

ソフトウェア開発力の強化のためソフトウェア・エンジニアリングの啓発と、実践に資するべく発刊してきたSEC journalも、本号で50号を迎えた。これまで本誌が果たしてきた役割や今後の方向性、更にIoTの本格的展開が期待される現在にふさわしい先進的なソフトウェア・エンジニアリングのあり方について、SEC journal論文賞表彰委員長でもある片山卓也先生にお話を伺った。

SEC journalが果たしてきた役割

松本　50号を迎えた本誌ですが、昨年度までの論文の採録数は68件となっています。数は決して多いとは言えないかもしれませんが、SEC journalが果たしてきた役割は小さくないと思っています。片山先生はどう見られていますか。



片山 卓也（かたやま　たくや）

東工大電気工学科卒業（1962）、同情報工学科助教授（1974）、教授（1985）、北陸先端大情報科学研究科教授（1991）、学長（2008）、中央大研究開発機構教授（2014）、日本ソフトウェア科学会理事長（1991）、名誉会員、情報処理学会理事（1985）、名誉会員、専門：形式的ソフトウェア開発方法論、高信頼組込みシステム、ソフトウェア発展・進化機構、法令工学

片山　大きな意味があったと思っています。実践的な論文を載せるアカデミックな場所というのは決して多くありません。どうしても新規性や独創性が評価の基準になってしまうので、原理は分かっているがプラクティスに意味があるというような論文を載せる場所は限られています。業界誌がたくさんあった頃はそれなりに投稿場所もあったけれど、今はその数も減りました。国のソフトウェアの政策に責任を持つSECのような機関が場所を提供していることは意義深いと思います。

松本　ありがとうございます。確かに、ソフトウェア・エンジニアリングに範囲を絞ったプラクティス論文誌は、SEC journal以外にはないように思います。論文の質という点では、どう評価されていますか？

片山　それなりにきちんとしたものであると思います。注文も付けましたが、それは質が良い悪いということではなくて、どの辺りの論文を採るかということなのです。私としては、試しにやってみたというものが載っていて、業界における技術導入に道を拓くような論文がたくさんあって欲しいと思っていたのですが、実際掲載されている論文は、ある程度評価が決まっているものを取り上げたものが多かったように思います。逆に言えば、新しいソフトウェアの作り方を巡って、試しにこんな新しいツールを使ってみたらできました、というようなものが少なかったのではないでしょうか。これは論文の集め方というか、テクニックの問題もあるのではないかと思います。応募を待って審査をするというスタイルでは、発掘できない。業界の人は基本的に忙しく、しかも新しいことに取り組んでいるような人は、ほかにもたくさん仕事を担っていることが多く、論文を書く時間がないという状況にあります。新しい試みをSEC journalで紹介するためには、こちらから依頼をして書いてもらうなど、論文を集めるための努力も必要になるでしょう。いわば“枯れた技術”であれば書く手法も決まっていて、書きやすいし、集まりやすい。しかし、例えば以前クラウドが出てきた当初は、まだ細部では分からないことがたくさんあった時期で、産業界でも色々なことを試しているグループがありました。しかし、そういう話がSEC journalにはあまり出てこなかった。それは残念だと思います。

松本　産業界からの応募をもっと増やしていく必要があるということでしょうか？

片山　産学連携を一生懸命にやっている大学から、というのもあると思いますが、大学の人が産業界のことを想像して書いたものは、その中には将来的に非常に面白いものになるアイデアが含まれている可能性はありますが、しかしそれはアカデミックなジャーナルで拾えば良いわけです。やはり産業界に根ざして、産業界の難しい問題を新しい手法で解いたものが欲しいですね。

それを書けるのは産業界の人か、産業界と密な連携をしている大学や研究機関の人だと思います。SEC journalは、そういう人の論文を集めよう、題材は産業界にある、というところからスタートしました。その方針は正しかったと思います。

松本　ただ産業界の場合だと、先生もおっしゃったようにみなさん忙しいですね。

片山　忙しいことに加えて、あまり価値のないものまで秘密にする傾向がありますね。

松本　新しい技術を製品に適用しました、ということになると企業として守秘上それを表に出しにくいという面もあるのでしょうか。

片山　それを出したからといって企業が困るということはないと思います。かえって、あの会社はこんな素晴らしいテクノロジーを使っているということで、評価は上がると思います。秘密にしておく価値はあまりないでしょう。

プラクティスにかかわるホットな報告が欲しい

松本　企業の方から研究成果を発表していただこうとする場合、論文の書き方に慣れていないという問題もあります。

片山　そこは査読する人が何か言えばそれで解決するでしょうし、多少書き方が悪くても構わないと思います。いわゆるアカデミックな論文の書き方に則って、まず背景を述べ、その中で自分はこれをこう解決した、それのどこが新しくて、ほかとの比較ではどういう意義があるのか、といったオーソドックスな書き方は難しい。ほかとの比較といっても、そもそも他社がどうやっているかなんて知らない場合があるでしょう。とくに産業界にいる場合は、他社と比較して変なことを言ったら失礼になるかもしれません。ですからそこは緩やかにして、何をやったかということを中心にすれば良いのではないでしょうか。論文の価値の最も重要なところは、アカデミックなジャーナルの場合は明らかに独創性であり、何かを新発見したということでしょう。一方業界に関する話というのは、新しい発想というよりは、既存の技術や知見を新たに適用したというところにあると思います。場合によっては二番煎じでも良くて、追試を行ったらうまくいったというのでも良い。独創性を評価の主眼に置くことは少し問題だと思います。

松本　なるほど。ただ、もう少し最低限のことは書いて欲しいと感じる論文もあることは事実です。単にやったことを書いただけでは、ほかの人にとっては役に立たない。どのような課題があってどれだけ苦労したか、実行の過程において何を工夫したのかということは明確にして欲しいですね。

片山　確かにそこは要求すべきでしょう。しかし論文としての形の素晴らしさではなく、適用した中身の素晴らしさを第一に見ていきたい。外見は悪くても中身のおいしいものを採るということをすれば、論文を読んだ人が同じようなことをやってみようとしたときの参考になると思います。

松本　プラクティスなのだから、「やってみよう」というところが重要ですね。そうでないと産業界に広がっていかないわけですから。

片山　産業界からのデータを分析してモデルを作ったとか、産業界でよくやっていることを厳密に適用して評価してみたというものなど、最初から結果がある程度分かっているような論文が多かったような気がします。もう少しホットで今話題になっている手法を適用してみるとか、そういうほうが読んでいても面白いですよね。

松本　両方あるのかもしれないですね。既存技術に近いところだけれど、例えば人材育成に関する論文がそうですが、それほど目新しいことをやっているわけではない。しかし、企業ではソフトウェア技術者をどう育成していくかということについて非常に苦労されていて、その事例は大いに参考になります。

片山　確かに人材育成については、優秀論文に残ったものもありましたね。

松本　それと同時に、全く新しいことに取り組んでみてこうでした、という論文もあって欲しいですね。

片山　そうですね。いずれにしても、SEC journalのような雑誌は世界的に見てもそうあるわけではない。今後もぜひ活用していただきたいと思います。

来るものを待つのではなく書いてもらう

松本　論文の投稿を増やすという意味では、SEC journalに論文を載せることが学位取得のポイントになるという大学も幾つかあるので、そういう大学をもっと増やしていったらどうかとも考えています。

片山　なるほど。しかしそれは難しそうですね。学位の基準は大学によって違うし、その基準は大学のポリシーを表しています。ですから、そういう論文が載るようなジャーナルになって欲しいとは思いますが、それなりに色々なことをやらないといけません。例えば、ソフトウェア・エンジニアリングで言うとInternational Conference on Software Engineering（ICSE）でもインダストリペーパーはたくさん採ろうとしていて、レベルもなかなか高い。ですから、覚悟を決めてそれと競うような内容にしていく必要があります。そのためにも開発現場に声をかけて良い論文を集め、相乗効果で全体のレベルも上げていく必要があります。

松本　ただ投稿されたものだけでやっていてもたぶんレベルは上がらないということですね。

片山　論文の査読をする人の意識や取り組み方という問題もありますね。実際そ

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）。

のことに不安を抱かせるような論文も幾つかあったように記憶しています。良いカンファレンスの査読者になるということはそれ自体名誉なことであって、名前も公表されます。そこに入れるかどうかが自分の実力の証になる。ですから、査読をするときもかなりしっかり調べて、この論文が良いか悪いかを委員会で論じることが求められます。全員で議論しながらカンファレンスを作るので、査読者もしっかり取り組みます。いずれにしても、評価が高まれば論文も査読も相乗効果でレベルが上がっていくと思います。

もともと日本の業界は一生懸命やっても表に出さないですから、お願いして書いてもらう。招待論文でも構わないから、これはすごいなというものが毎号1本は載っているというようになれば、みるみる良くなっていくと思います。例えば、SEC で新しい先進技術の適用事例の「事例集」を作っていますよね。ああいうところに出てくるようなものを、こちらからお願いして長めのものを書いてもらうのはいかがでしょうか。学者であれば良いジャーナルに掲載されることが学位や評価の基準になりますし、名誉になる。しかし業界の人にとってはそれほど名誉ではない。だから待っていても論文は投稿されない。良いことをやっている人を見つけてお願いして書いてもらう、それをやっていけば良い論文が投稿されるようになり、執筆者の社内評価も高まり、良い循環になっていくのではないでしょうか。今でもサーベイなどについてはみなさんある程度参考にしているでしょう。更に先進的な事例などについても誰かに書いてもらう。SEC journalに招待されて論文を掲載することが名誉なんだということになれば、投稿論文のクオリティも上がっていくでしょう。

再び注目を集めるプログラムの世界

松本　ソフトウェア・エンジニアリングという言葉が出てきたのは1968年のNATOの会議だと言われ、もう50年になるわけですが、この50年間のソフトウェア・エンジニアリングの進歩ということについて、どうお考えですか？

片山　明らかに色々なことが進歩して、本当にエンジニアリングの分野になりましたね。NATOの会議の頃は、大規模プログラムの時代です。私も興味を持って当時の論文を読んでみましたが、大規模プログラム開発の失敗が相次ぎどうしようかということで、当時のコンピューターに関するそうそうたる研究者が会議を開いて、今で言うソフトウェア・エンジニアリングからはもう少し広い、プログラミングとか言語なども含めたカンファレンスだったようです。エドガー・ダイクストラ（Edsger Dijkstra）の「GoTo文は有害である」（“GoTo Statement Considered Harmful”）という論文などが発表された時期で、構造化プログラミングや制御構造のハイアラーキーの話とか、検証に関する不変表明法などが議論された時代でした。ICSEもその頃にできました。ICSEの論文で見ていくと、その後、今から30年くらい前からソフトウェアの開発プロセスやプロジェクト管理の話が盛り上がった時期が長く続きましたが、その後またプロダクトに関する話が多くなり、プログラムやソフトウェアの解析や構造化に関する論文が多くなったという印象です。それはなぜかと言うと、大きなものを間違いなく作るにはプログラムが分析できなければいけないからです。それからメンテナンスの問題があります。メンテナンスの問題は泥臭い問題だと誰もが考えていると思いますが、メンテナンスに関する国際会議の論文集などを見ると、プログラムの解析の話が随分多いと言う印象です。ツールを使っていかに危ないところを見つけるかとか、波及分析をどうするとかいう話で、色々な技術が進んでプログラムやソフトウェアを分析できるツールが出てきたことによって、そのような技術的な話が増えているのではないでしょうか。

松本　なるほど。いったんプロセス寄りになりマネジメントのほうに注目が集まっていたものが、またプログラミングのほうに戻ってきたということですね。

片山　要するに仕様を書くことも含めて、物事をちゃんと書かない限り分析のしようがないということで、そちらに揺り戻しているということだと思います。一時私もプロセスの研究をしましたが、研究の中心が管理系の話になり、レベル幾つだからこの会社は良いとか悪いとかいう話になった頃プロセスから離れました。最近はアジャイルやDevOpsなど面白い話が色々あるようですね。

松本　プロセスに関しては最近はIoTの大きな流れもあり、きっちり上流から固めていくというスタイルではソフトウェアが作りにくくなっています。何がつながるか分からないというように要件そのものもあいまいになっている。それならアジャイル的にとりあえずプロトタイピングでやってみて、それをブラッシュアップしていくという開発手法になっていくことが考えられるのではないでしょうか。今までのような、がっちりした手順では立ちゆかなくなっているケースが増えているように思います。

片山　全部を決めてから作り始めるということがやりにくくなっている、それは事実ですね。もう一つは、保守の問題にかかわるけれど、何か付け加えようという話になったときに、どうやって付け加えるか、中身を全部知っている人がいるわけではないので、できているものを分析しない限りものが作れなくなってしまうわけです。従って、きっちり分析できるように物事を作るとか、人間はいい加減だから機械に分析させて、おかしなところがないことをチェックするという技術がこれからもっと必要になってくると思います。IoTの環境で、作ったもの同士が、相手が分からないままつながるわけはないんです。この世界にマジックはない。論理と論理のぶつかり合いですから、きちんと分析してつなげないといけない。分析できるように物事ができていること、分析できるようなフォーマリズムで物事が書いてあるということが必要です。アジャイルやDevOpsでも同じことだと思います。作って動かしてみないと人間の想像力は働きませんから、まず動かしてみて色々なことが分かるということでしょう。そしてその次のサイクルに入るときに必要な分析を行えることが重要です。あらかじめ色々なことが分かっているものとは作り方が違うということだと思いますね。

今求められるシステムズエンジニアリングとは

松本　そういう意味で、私たちは最近システムズエンジニアリングという考え方に注目しています。日本語で表現してシステ

ム工学と言ってしまうと制御系のイメージを持たれがちですが、システムの開発をなるべく抽象化したところから全体像を捉えて、それから徐々に具体化して物に落としていくという考え方でやっていかないと、下から個別に物を積み上げて作っていくという考え方でやったのでは、今の世の中では全体の設計ができなくなってくる。

片山　その通りですね。しかしそれがなかなかうまくいかない。

松本　なぜなんでしょうか。

片山　物を作り始めると作っているところに張り付いてしまうんです。本来、そこでもう一回全体を見渡し、常に抽象的な世界を更新しながら下のほうも作って、上から下まで矛盾のない世界で物事を完結させていくことが求められるのですが、みなさん忙しすぎるから、上のほうを作ったら下に行ってしまって、後は下だけで物事をやってしまっているということではないですか。

松本　下のほうはイメージしやすいんですね。部品などは目に見えるものになっている。そこで、これとこれを組み合わせればこうなるよねと下から積み上げるビルディングブロック的に考えていくと分かりやすいんですね。上から抽象的に俯瞰して考えるというのはなかなか難しい。この辺りの発想のし方から変えていかなければならないと思います。

片山　目の前の問題の解決だけを考え、そのシステムを作ることだけを考えたらあまり問題にならないのに、抽象的な上のほうのこともきっちり考えておこうということですから、無駄なことと感じるかもしれません。しかし、そうやっておけば後は確実に楽になるのです。

松本　幅広く業界を横断して色々な物が連携し、サービスなどが展開されるようになると、一つのサービスだけに限って考えていれば良いとはいかなくなりますね。うまく連携させることができるように考えていかなければなりませんし、だんだん難しくなってきています。

片山　そこで必要だと思うのは、ある程度動かせるというか、上のほうも推論ができるように作り上げておく、ということです。人間は物事を忘れたり、そもそも全体を見通すことは不可能に近いので多くの見落としをします。見落としをしないのは機械です。機械は疲れを知らずにとことん調べてくれますから、機械に分析させれば良い。もちろんそのためには色々なことを決めなくてはなりません。スペックをロジックで書き、それをだんだん具象化して最終レベルに達したときに下にコードができる、上には論理式があるというように階層的にでき上がっている、そういう発想でパリの地下鉄のシステムを作り上げたという話があります。これくらい徹底してやれれば、何かを変えるというときにも、計算機の力を借りて推論をやり直すとか、新しい要求を付け加えると、ここがバッティングするよ、というようなことを機械が調べてくれるようになる。でも、残念ながら日本にはそういう文化はないですね。ここまで徹底することが通常の商業的ソフトウェア開発で可能かどうかは分かりませんが、それでも開発や進化・変更プロセスにもっと計算機による推論能力を使っても良いのではないかと思います。計算パワー的には十分可能な時代になったと考えます。

プログラムは人間の創造性の産物

松本　正にそこはAIですよね。AIを使って自動的に学習して、ここが問題になるとか。

片山　そうですね。今で言うディープラーニングがすぐに使えるかどうは分かりませんが。

松本　どうでしょうか。ある程度人間が抽象化したものを具現化していく作業は機械学習によってできるようになってくるのではないかと思うのですが。

片山　ワトソンの技術なんかがそれに向いているのではないかと思います。今話題のディープラーニングは、規則としては特徴付けしにくいデータ集合をニューラルネットに学習させることに成功し、碁のチャンピオンに勝ってしまうような性能を発揮しました。しかし、記述された規則や論理としてその集合を特徴付けているわけではありません。一方、プログラムやソフトウェアはそうではなくて、規則を記号として明確に書いたものであり、その学習にディープラーニングの考え方が上手く使えるかどうかは今後の研究によるところが大きいと思います。具体的な話では、プログラムエラーの膨大なデータベースから学習して、デバッグやテストを効率的に行うことが可能になると良いのですが。いずれにしろ、プログラミングやソフトウェア開発へのAI技術の適用は今後の大きなテーマだと思います。

松本　私もAIの進出でプログラマーの仕事がなくなるとは思っていません。プログラムというのはある意味では創造性の産物です。芸術に近いようなところがある。人間が創造性を発揮してプログラムを作るところに発見があるような気がします。機械で厳格にすればできるというようなものではないと思います。

片山　そうですね。

松本　私の経験でも、プログラムと言ってもある機能を実現するためだけのコードであれば、誰でも作れる可能性があると思います。その中でも本当にきれいにできているプログラムと言ったら良いのか、メンテナンスも含めて誰が見ても分かりやすいプログラムというのはすごく良いプログラムであって、それは機械では作れない。

片山　メンテナンスのしやすさとは何かということを機械が理解しなければいけないわけで、そのためには今流のやり方で言えば、メンテナンスのしやすいプログラムを膨大な数を集めて機械に食べさせてみて、ということになると思うけれど、そんな単純なことでうまくいくとはとても思えません。そもそも出来の良いプログラムを大量に集めること自身が非常に困難です。もっと本質的な原理的な飛躍が必要だと思います。

松本　そういう意味では、ソフトウェア・エンジニアリングの方向性が、プログラミングのほうにシフトしつつあるということはすごく良いことだと思います。ソフトウェアがこれから非常に大きな役割を果たしていく中で、プログラミングはますます重要になるのではないかと考えています。

片山　そうですよね。私は一時、JAVAが出てきた頃ですが、新しいプログラミング言語なんてもう生まれないのではと思いましたが、大きな間違いでしたね。

松本　PythonやRubyなどが出てきた。

片山　最近は百花繚乱で、日本発のRubyなどが世界的に使われ

ているのは嬉しいことです。また、関数型言語のような硬い言語が商用開発でも使われるようになりました。ただ、プログラミング言語の研究開発を継続的に行っている人が欧米に比べて少ないのは残念です。我々日本人は流行のことに流れやすいようです。

松本　ソフトウェア・エンジニアリングを専門にされている先生の数もあまり多くありませんし、昔からプログラムやプログラミングに一貫して取り組まれている先生は少ない。そこに日が当たるようなことを考えていかなくてはいけませんね。

片山　アメリカとの比較で言うと、もともと人口は日本の倍以上ありますが、世界中からエンジニアを吸い付けていますよね。それは世界的な企業が優秀な人を集めて仕事をしていることに加えて教育制度も大きいですね。実は近代的な大学院教育を始めたのはアメリカなんです。大学制度はヨーロッパで作られましたが、大学院はある意味でアメリカが確立したと言って良い。そこに海外から優秀な学生をたくさん引き入れて、その人たちが新しい学問を作り上げ、ITインダストリを作り上げたわけです。その点、アメリカはすごいと思いますね。

価値ベースの評価にしないとやりがいが見えてこない

松本　話は変わりますが、ソニー生命保険が今年の春インターネットで行った「日本の中高生のなりたい職業」の調査で、男子中高生のトップが「ITエンジニア・プログラマー」でした。

片山　それは素晴らしい。

松本　すごく良いことですね。せっかく今の若年層がそういうことに関心を持ち始めているのだから、教育もきちんとしていかなければいけないと思います。

片山　ではもう「3K」ではなくなりましたか。

松本　中高生ですから、職場の実態をどう見ているかそれは分かりません。しかしいずれにしてもプログラミングに興味を持っているのは事実でしょう。

片山　きちんとした処遇をしていかなければなりませんね。

松本　同志社大学の中田喜文先生が行ったITエンジニアの処遇の各国比較がありますが、そこでは日本は先進国の中で最低ラインです。給与が低いだけでなく勤務時間も長いし、満足度も低い。

片山　そこをなんとかしないと中高生が育たないでしょう。

松本　せっかく関心を持っていても、実際に職場に入ってがっかりしてしまいそうです。

片山　なぜそうなんでしょうね。

松本　産業構造そのものにも大きな問題があると思います。日本のソフトウェア業界は、プログラムの価値にお金を払うのではなくて、その作成にどの程度の工数がかかったか。何人が何日働いたかで計算している。そういう契約形態では価値ベースにはならない。そこが大きいような気がします。

片山　フランスなどもあまり高くないのではないですか？

松本　高いのはアメリカとドイツですね。日本は中国と同じ程度です。

片山　アメリカが成功した大きな理由の一つは、新しいビジネスと結び付けてITを進歩させたことにあるのではないかと思います。ああいうし方で新しい技術を作りマネジメントのやり方も変えて会社を発展させていく。日本では難しそうですね。もっと硬くかっちりとしていて、新しいことはできないけれど安定しているというのが日本のITのシステムなのかなと思います。それはそれで社会的な価値は十分にあると思いますが。

松本　そうですね。高品質で売っていくというのも一つの方向ですし、新しいビジネス、新しいサービスで売っていくのも一つのやり方で、両方あるかなと思います。ただ、今後ダイナミックな社会になっていくと、高品質だけで売っていくのは難しくなっていくのではないかと思います。

IoTの普及で再認識される「形式手法」の価値

松本　これまで先生は、数学的な手法を用いて正しいソフトウェアを開発する形式手法（フォーマルメソッド）で、リジットにプログラムを作ることに取り組まれてきましたが、残念ながら形式手法はなかなか産業界に一般的な方法として普及していかないのですが、それについてはどうお考えでしょうか。

片山　本当にクリティカルな、故障したら困るというところでは使わざるを得ない技術だと思います。同じものを今までと同じ方法で作れるかと言えばそうではない。必要なテストの数が膨大になり、経済的に引き合わないということになってしまう。どれだけ注意深く作るべきか、ということにかかわると思いますが、それほど注意深く作る必要がないものは今まで通りの方法で作れば良い。私は形式手法の一番の使い道は、長く使うもののメンテナンスだと思っています。ずっと使い回すために何か新しい機能を付け足すというとき、テストする人もいない、中身も分からないということになると、そのソフトウェア自体を分析できるかどうかが最後の手段になります。

松本　おっしゃる通りですね。検証に近いというか。それが仕様通りになっているかということがちゃんと証明できるという仕組みを入れておかないと、後から追加したときにこの辺りに影響があったということが分からなくなってしまう。

片山　IoTの世界に新しいものをつなぐときには、いい加減につなげば良いということではない。新しいものが矛盾なく接続されなければならない。規格も重要で、いい加減な規格のものをつなげと言われてもできない。無理は通らないんです。そのためには分析可能な形で設計書とかシステムのコードなどが書かれていなければならない。そこに形式手法の使い道があるのではないかと思っています。間違いのないものを一から作るという意味では、人間はそれなりに注意力があり、またテストでもある程度のことは分かるから、そこそこのものは作れてしまう。しかし、作ってしまって巨大になったものがどう振る舞うかということの細部は人間は分からないので、そこは機械にやってもらうしか手がないのです。しかし、これはなかなか理解してもらえません。

松本　SECとしてもそういう考え方を広めていかなければならないと考えています。これからますますそういう時代になってくるわけですね。あいまいな要求条件の下で作ることになるし、一度作ったものを環境が変わっても使えるようにしていかなくてはいけません。それをどうやって保証していくのか、ある程

度形式的に、きちんと均一化されていないとできません。

片山　やりようがないんです。

松本　試験のケースもべらぼうに増えてくるから、いちいち実地試験でやっていたら大変です。全部つながなければいけなくなる。

片山　しかもそれぞれのシステムも、作っているときだったら、担当者がいるから、その人に聞けば分かる。しかし人が去ってしまった後でつながなくてはいけないのだから、残っていて信頼できるのはコードだけです。それを分析するしか手がない。それは形式手法的なやり方でしかできません。

松本　実はSECでも昨年「システム再構築を成功に導くユーザガイド」というものを出していて、今、内容の改訂を進めようとしているところです。産業界は膨大なレガシーシステムを抱えていて、それをIoTやビッグデータなどの新たな要件にどう対応させていくかが課題になっている。一から新しく作り直すわけにはいかないので、既存のシステムをどうテーラリング、あるいはチューニングしていくかということで悩んでいるわけです。まだ形式手法の適用というところまでは踏み込めていないのですが、いずれそのようなツールや手法も考えていくことになると思います。

片山　できあがった物を分析するのは非常に難しいことで、ドキュメントなども完全なものであるかどうかも分からない。しかしそういうところにサイエンスやエンジニアリングを作るということが重要だと思っています。

巨大化するシステムのメンテナンスをどうするか

松本　企業から見ると、そういうところはコストに見えて投資に見えないんですね。既存のものをなんとかうまくもたせようという“メンテナンス”になってしまって、それならできるだけコストをかけずにやろうと。

片山　結局人をずっと張り付けておくだけになる。

松本　そういうケースが多いですね。しかしそういう人もいずれはリタイアします。

片山　そういう場合、企業はシステムを捨てているのでしょうか？

松本　いやそう簡単には捨てられません。最近の調査でも20年、30年と同じシステムを使っている企業が多いですね。できるだけコストを抑えて、現在あるものをだましだまし使っていくというようなケースがほとんどです。

片山　そういうことに関するエンジニアリングやサイエンスを作るという点では、SECに大きな役割がありますね。方法論も含めて展開するというのは民間企業ではできませんからね。

松本　“巨大システムのメンテナンスのためのエンジニアリングをどうするか”ということですね。そこに最新鋭のコンピューターサイエンスを使いたい。学のところで研究は行われているのでしょうか？

片山　興味を持っている人はいると思いますが、具体的にはよく分かりません。実用的な意味合いだけでなく、学術的にも面白い問題ですので、具体的な研究プロジェクトでも始まれば、多くの研究者が集まると思います。学の世界でも研究費がえられるようなテーマを設定しないと博士学生や研究員などのスタッフが集められないので、どうしても時代に流されます。

今はAI関連でないと大きな研究費が取れないので、本当に必要なことを頑固にやり続けることが困難です。しかしヨーロッパやアメリカにはそのような頑固さがありますね。例えばコンパイラの研究者がアメリカには、少数ですが、まだいるでしょう。日本でコンパイラの研究をやっている人はもう産業界にも大学にもいないと思います。OSの研究者もそうではないでしょうか。

松本　確かに以前はOSの研究者がいましたが、現在はそう多くはないのではないかと思います。

片山　国民性でもありますが、重要なことは投資だと思います。メンテナンスの問題は地味ですが、国や産業界にとって大変重要ですから、それに見合った投資をして欲しいと思います。

松本　企業が投資をしにくい分野ですし、そういうことこそ産学連携で官も巻き込んでやっていかなければいけないのでしょうね。

片山　SECが中心となってやってくれると良いですね。最前線のコンピューターサイエンスやAIを駆使してメンテナンスの科学や技術体系を作るということをやればインパクトは大きいと思います。

松本　重要なテーマですね。どんどんシステムは肥大化しているし、人の力でメンテナンスしていくのは難しくなって行く一方だと思います。

今日は、メンテナンスの話をはじめ、今後考えていかなければならない貴重な問題提起をいただいたと思います。最後にSECに期待されることをぜひお伺いしたいと思います。

片山　SECは日本に一つしかないソフトウェアに関する研究開発機関であり、実用・普及に貢献する機関です。ソフトウェアが良くならなければ国も良くならない。SECの役割は非常に大きいと思います。長年内部にいらっしゃると、価値が見えにくくなるかもしれませんが、国の重要なファンクションを担っていらっしゃるわけですから、ぜひともがんばっていただきたいと思います。

松本　はい、しっかり進めていきたいと思います。本日は貴重なお話をありがとうございました。