# 3. 日本のソフトウェア工学の今と未来

丸山勝久　立命館大学

## ソフトウェア工学を取り巻く状況

現代社会において，ソフトウェアが社会基盤のさまざまな局面を支えていることに異論はないであろう．もはや，我々の生活は，ソフトウェアなしでは成り立たない．しかしながら，現在のソフトウェアが社会的要請に対して十分に応えているかといえば，残念ながら，そうとは言い切れない．たとえば，金融システムの信頼性や電子商取引の安全性に対する不安感，会社統合時におけるシステム統合の迅速さに対する不満感は多い．

このような状況において，安心・安全な社会を支え，我々の生活を豊かにしてくれるソフトウェアをどのように作ればよいかという課題に挑戦しているのが，ソフトウェア工学である．もう少し控えめにいうと，ソフトウェア工学とは，コンピュータソフトウェアを対象として，その開発，運用，保守における生産性と品質の向上を実現するための技術体系や学問体系のことである[1)]．ソフトウェア工学を研究するとは，ソフトウェアを開発するための理論，原理，技術などを探求する活動を指す．また，ソフトウェア工学を実践するとは，理論，原理，技術に基づく方法論や，表記法，ツールを活用して，ソフトウェアを開発，運用，保守することを指す．

ここでは，まず，ソフトウェア工学を取り巻く状況を示す．残念ながら，ソフトウェア工学の研究者あるいは教育者という立場の筆者から見ても，ソフトウェア工学はいまだ工学として確立しているとはいえない．その理由として，**図 -1** に示す 4 つの観点から考えてみたい．

### ■ 人間の知的作業に依存した開発

最初に思いつくのは，ソフトウェア開発の中心が人間の知的活動におかれている点である．ソフトウェア製品のコピーは容易であるため，ソフトウェア開発においては量産コストが存在しないと考えてよい．多くの製品開発では，量産工程の改善によって生産性の向上が見られるのに対して，ソフトウェア開発ではこの方法は使えない．このため，生産性を向上させるには，人間の知的活動を支援，あるいは，知的活動でない部分を極力自動化するしかない．さらに，製品として広く市場で利用される複雑なソフトウェアを，単独で開発することは無理である．このため，数多くの人間がソフトウェア製品の開発（量産作業ではなく，分析・設計作業）に携わることになる．つまり，大規模，かつ，さまざまな知的作業を，組織的に集約する必要がある．人間の知的活動を管理すること，さらに，それらを適切に束ねることが，本質的に難しいことに異論はないであろう．

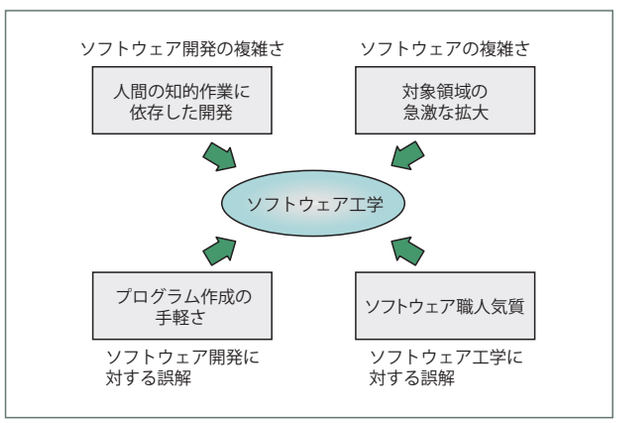


図 -1　ソフトウェア工学を取り巻く状況の関係

ソフトウェア開発が人間の知的作業に大きく依存している点は，ソフトウェア工学の目的である生産性だけでなく，品質にも大きく影響を与える．ソフトウェア工学では，良い作り方（開発プロセス）が良い製品（開発プロダクト）を生産することを前提としている場面が多い[☆1]．しかしながら，単純作業ではない高度な知的活動において，プロセスとプロダクトの関係にはまだまだ未知な部分が多く，この解明は挑戦的な課題である．

☆1 このような前提が成り立たないのであれば，開発プロセスを改善しても開発プロダクトの品質の向上は達成できない．

### ■ 対象領域の急激な拡大

ソフトウェア工学がなかなか確立しない２つ目の理由として，ソフトウェア工学が対象とする領域の急激な広がりがあげられる．ソフトウェアが実体を持たず，実世界に対して抽象的なものであるという性質は，さまざまなものへの適用という点から非常に有利である．ソフトウェアが社会基盤のあらゆる所に浸透し，さまざまな局面を支えているという現状は，このことを端的に表している．ソフトウェアの適用領域が広がれば，ソフトウェア工学の対象領域も連動して広がり，もはやその領域を研究者や実践者の都合で勝手に限定することはできない．

ソフトウェアが持つもう１つの性質として，それ自体が単独では動作せず，必ずハードウェアとかかわりを持つことがあげられる．つまり，ソフトウェアは，それにかかわるハードウェアに大きく依存する．このため，ソフトウェアを考える際には，ハードウェアの進歩を常に意識しなければならない．ここで，ソフトウェアを取り巻くハードウェアに目を向けると，それは急激に進歩している．たとえば，計算機の演算速度や記憶領域，ネットワーク帯域の増加は，数年前であれば不可能だと思われていた処理を可能としている．同時に，ハードウェアの進歩は，ソフトウェアシステムの利用者が望む要求の高度化を引き起こす．また，ハードウェアの進歩により，ソフトウェア開発支援環境も進歩するため，ますますソフトウェアによって達成できることが増える．このように，ハードウェア技術やソフトウェア技術の進歩は，要求されるソフトウェアをますます高度化･複雑化している．

対象領域の広さや多様さを扱うためには，抽象化という考え方が重要であり，ソフトウェア工学では一般化された手法がより好まれる☆2．一方，一般化された手法は，そのままでは現実の問題に適用することはできないため，ソフトウェア開発現場では，ソフトウェア工学の成果がなかなか活かされない．ソフトウェア工学の成果を実際の開発現場に浸透させるためには，それぞれの対象領域に特化した手法，あるいは，一般化された手法を対象領域に特化（具体化）する手法が要求される．残念ながら，ソフトウェアの高度化・複雑化の速度に，ソフトウェア工学の抽象化技術や具体化技術が追い付いていないというのが現状である．

### ■ プログラム作成の手軽さ

計算機ソフトウェアはプログラムとして記述されるが，一般的にプログラム作成はハードウェア製作に比べて手軽である．ハードウェアの製作には，特殊な設備や工具が必要であることが多い．また，一度製作してしまうと，それを簡単に作り直すことができない．これに対して，計算機が普及した現代において，プログラムの作成環境（エディタやコンパイラ）を手に入れる費用は小さく，誰もが手軽にプログラミング可能である．また，通常のプログラム実行において，ハードウェアが壊れることはなく，何度でもプログラムを作り直すことができる．さらには，プログラムはコピー可能であるため，以前に作成したものや他人が作成したものを改変して，新しいプログラムを作成することも容易である．

ソフトウェアシステムを動作させる実体がプログラムであることより，プログラムがソフトウェア開発の主要な生産物であるという考えは間違いではない．しかしながら，プログラム作成はソフトウェア開発において，ごく一部の作業であり，プログラムが手軽に作成できることと，ソフトウェア開発が容易になることは同じではない．限定された条件のもとで，特定の要求を満たすプログラムを作成することに比べて，さまざまな状況（利用方法や実行環境）を想定した上で，特定の要求を実現するソフトウェアを作成することは，とりわけ困難である．また，納品後の保守や将来の改変のために必要な情報を残しておかなければ，そのソフトウェアを長期間安心して利用することはできない．

動作するプログラムができればよいという考えから脱却し，プログラム作成と（品質や保守性を意識した）ソフトウェア開発の違いを十分認識しておく必要があるが，このような認識が広まっていないのが現状である．

### ■ ソフトウェア職人気質

本稿の読者の中には，書籍「ソフトウェア職人気質」[2)] を読まれた人も多いだろう．この本では，ソフトウェア開発を工学という枠に閉じ込めるという考えを払拭するために，ソフトウェア開発における技芸に光を当てている．ソフトウェア職人気質とは，プログラミングというものが熟練の必要な技芸であることを常に理解しているという感覚を指す．ソフトウェア工学研究者の筆者から見ても，その主張に共感できる部分は多く，ソフトウェア工学の負の部分（ソフトウェア開発作業の中心に人間が存在することを無視して考えられた役に立たない手法やツール）を的確に指摘している．

たしかに，現時点で，ソフトウェア工学がソフトウェア開発を十分に支援しているかといえば，必ずしもそうとはいえない．現在稼働中のソフトウェアには，ソフトウェア工学の成果に基づいて作成されていないものも数多く存在する．また，ソフトウェア工学の成果の中には，

☆2 ソフトウェア工学の研究者は，モデル化という言葉を好んで用いる．ソフトウェア工学におけるモデル化とは，実世界の構造や振舞いから，特定の文脈や観点で注目する部分だけを抜き出すことで，主に，概念や関係を一般化した論理的なモデルを作成することを指す．

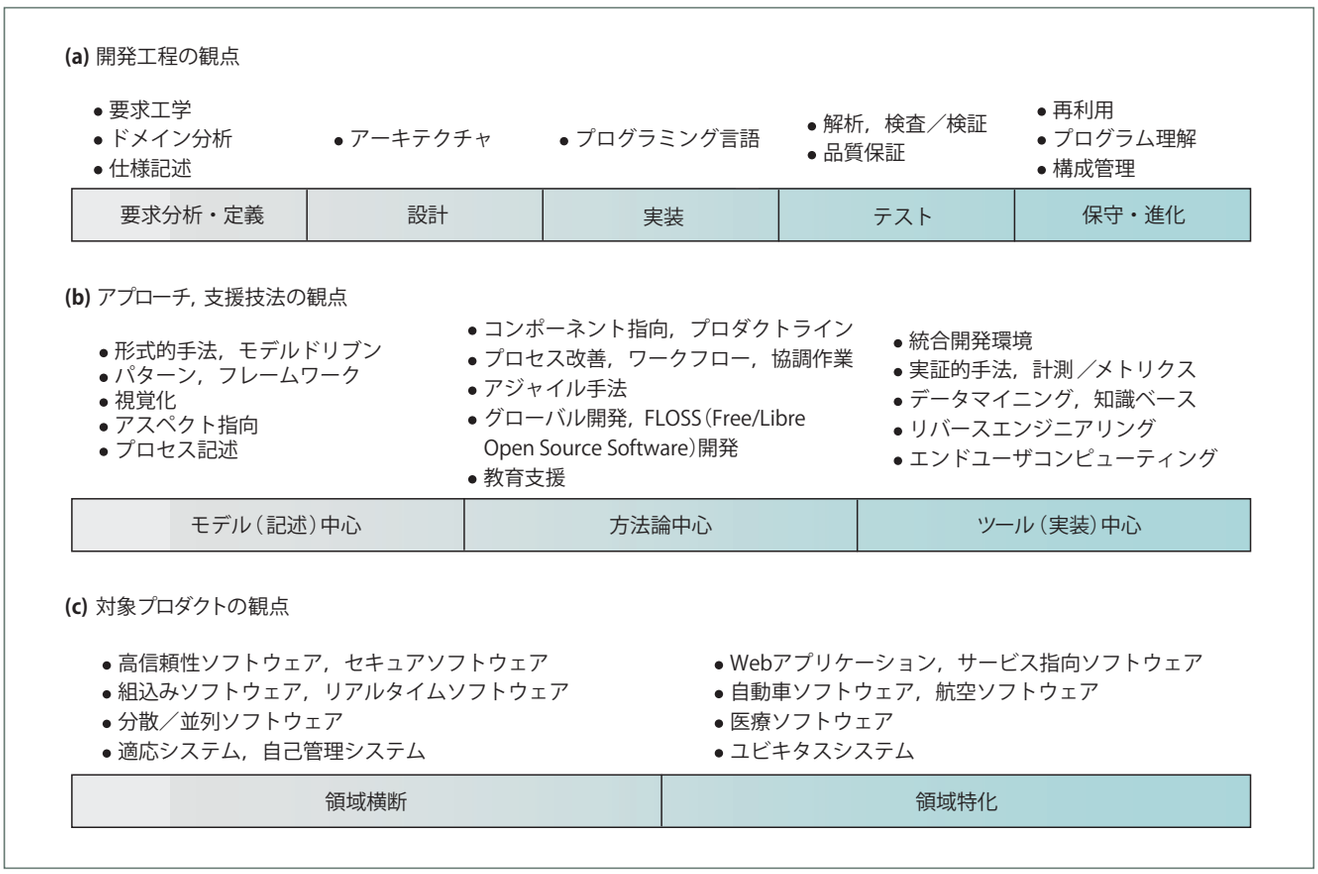


図 -2　ソフトウェア工学における近年の研究テーマ

実際の開発においてその活用を敬遠されるものも多い．このような観点から，ソフトウェア工学に対して否定的な意見が存在するのは事実である．

しかしながら，ソフトウェア工学に対する否定的な意見の多くは，ソフトウェア工学の定義にとらわれすぎていることに起因していると，筆者は考えている．IEEE 標準（Std 610-1990）によると，ソフトウェア工学とは，ソフトウェア開発，運用，保守に対する系統的かつ規則化された定量的アプローチのことを指す．このようなアプローチがソフトウェア開発を成功させる唯一の方法であると，ソフトウェア工学コミュニティが考えているのであれば，ソフトウェア工学が熟練開発者の技芸を軽視しているという主張は正しい．しかしながら，多くのソフトウェア工学研究者は，ソフトウェア開発における開発者の活動や生産されたソフトウェアがすべて定量化可能であるとは考えていないし，定義としてのソフトウェア工学とその現実の違いを認識している．よって，ソフトウェア工学が職人気質を軽視しているという考えは誤解である．

## ソフトウェア工学研究の今

当然のことだが，ソフトウェア工学を取り巻く状況の変化に応じて，ソフトウェア工学研究も変化してきている．本章では，近年のソフトウェア工学の研究テーマを簡単に紹介し，ソフトウェア工学研究の現状を説明する．

ここでは，**図 -2** に示す 3 つの観点で，研究テーマを分類した☆3．ソフトウェア工学分野において，この分類が正しいわけでも，さらに，ここで取り上げたものだけが主要テーマではないことに注意してほしい．

ソフトウェア工学の活動を，ソフトウェアの開発のライフサイクルで捉えることは自然であるため☆4，観点 (a) に示すような開発工程ごとに，国際会議やワークショップ（要求工学に関する RE，テストに関する ISSTA，保守に関する ICSM など）が以前から開催されている．ソフトウェア工学の対象プロダクトの種類が少ないころは，ソフトウェアの作り方も限定されており，開発工程ごとに議論が閉じていても差し支えなかった．

☆3 IEEE-CS/ACM によって，毎年開催されているソフトウェア工学分野の主要会議である，ソフトウェア工学国際会議（ICSE : International Conference on Software Engineering），ソフトウェア工学の基礎国際会議（FSE : International Symposium on Foundations of Software Engineering）の論文募集テーマ，会議セッションのタイトル，併設ワークショップ名などを参考にした．

☆4 多くのソフトウェア工学の書籍が，ライフサイクルにあわせて章立てされている．

また，ソフトウェア開発を管理するという観点からは，開発工程ごとに技術が分割されている方が都合がよい．

これに対して，近年の開発対象プロダクトの広がりは，ソフトウェアの作り方を変化させ，その結果として研究テーマを開発工程で区切ることが適切ではなくなってきている．たとえば，セキュアソフトウェアの開発においては，すべての開発工程を通して，一貫してセキュリティやリスクを意識する必要がある．つまり，セキュアソフトウェアをどのように開発するかという議論を，開発工程ごとに行っていても意味がない．むしろ，開発工程の区切りを超えた研究が必須である．

さらに，研究テーマの広がりに関する近年の特徴として，ハードウェア技術やデータ処理技術などソフトウェア工学を取り巻くさまざまな技術の進歩により，ソフトウェア開発に対するアプローチや開発支援技法が多様になったことがあげられる．従来のハードウェア性能では扱うのが困難であったアプローチ（大規模な状態演算を要求するプログラム解析やモデル検査，大量のデータ処理を要求するマイニング手法など）に手が届くようになってきたのは，計算機の性能が向上したおかげである．

また，Web 技術の発展は，ソフトウェア開発の分散化やグローバル化をより促進させており，このような観点に基づく技法（たとえば，FLOSS 開発支援）の研究も近年活発になってきている．特に，Web の登場は，アプリケーション開発期間の短縮化をより加速させており，従来のような大規模／長期的開発だけでなく，小規模／短期的開発を支援する技法の研究（たとえば，アジャイル開発支援）にも注目が集まっている．同時に，ソフトウェア開発の主体が，ソフトウェアや開発者を管理することから，開発者間のコミュニケーションを促進することに移ってきていることに起因した研究テーマも登場している．たとえば，熟練経験者の経験や知識の共有を目指した，ソフトウェアパターンの研究がこれに相当する．

もちろん，ここで述べたことだけで，ソフトウェア工学研究の動向を捉えることはできない．しかしながら，ソフトウェア工学研究に関するアプローチや対象プロダクトの広がりは，図 -2 に示す研究テーマの多様さを見れば理解できるはずである．近年の傾向として，特定のアプローチや対象プロダクトに特化したコミュニティ（国際会議やワークショップ）が，数多く登場してきている点も，このことを裏付けている．

## ソフトウェア工学の未来

ここでは，まず，ソフトウェア工学研究の方向性を紹介する．その後，日本のソフトウェア工学研究の将来について，個人的な見解を述べる．

### ■ ソフトウェア工学研究の方向性

本稿の目的は，ソフトウェア工学における現在流行している研究テーマを洗い出すことでも，将来の研究テーマを予測することでもない．よって，個々の研究テーマに関する議論は避けるが，ソフトウェア工学の世界的動向に興味のある読者は，ICSE 2007 において開催された Future of Software Engineering（FOSE'07）セッションの予稿集[3]を読むとよい[☆5]．もちろん，ICSE の FOSE セッションだけが世界のソフトウェア工学の流れを決定しているわけではないが，個々の研究の動向を知るのに大いに参考になる．

ここでは，文献 4）において，Osterweil 氏がソフトウェア工学全体の未来について述べた部分を簡単に紹介する．彼は，ソフトウェア工学の未来として 2 つの方向性を示している．1 つ目は，ソフトウェア工学の適用領域が今よりもますます広がり，さまざまな（伝統的な）分野における課題を解決していくだろうという予測である．ソフトウェア工学研究の動向から見て，当然の展開である．ただし，研究者の外部に存在する実践的課題の解決を最優先に考えてソフトウェア工学の理論を構築していくという伝統的な方法，すなわち，ソフトウェア工学の研究の動機付けを研究者の外部に求めるという姿勢に対して，彼は疑問を投げかけている．このような方法が，今後も有効であり続けるかどうかについて検討する必要があるだろう．

このような観点から，Osterweil 氏は，ソフトウェア工学のもう 1 つの方向性として，好奇心ドリブン研究（curious-driven research）を提唱している．好奇心ドリブン研究とは，従来の課題ドリブン研究（problem-driven research）を補うものであり，より深い本質的な疑問に対する回顧である．このような疑問は，実世界の問題に長期間および真剣に取り組んできた結果として，研究者の心に発生するものである．たとえば，「設計（design）とは何か」，「モデルとは何か」，「ソフトウェアとは何か」という本質的な疑問が好奇心ドリブン研究を進める上での動機となる．ただし，このような好奇心ドリブン研究の成果は，課題ドリブン研究の成果に比べて，論文として採録されにくいという悲観的な意見も彼は持っている．これに対しては，ソフトウェア工学研究の将来の方向性を決定するのは我々研究コミュニティ自身であり，現在，それを議論する時期にあると主張している．

☆5 IEEE CS および ACM の Digital Library から手に入る．また，同様のセッションは ICSE 2000 でも開催されている．

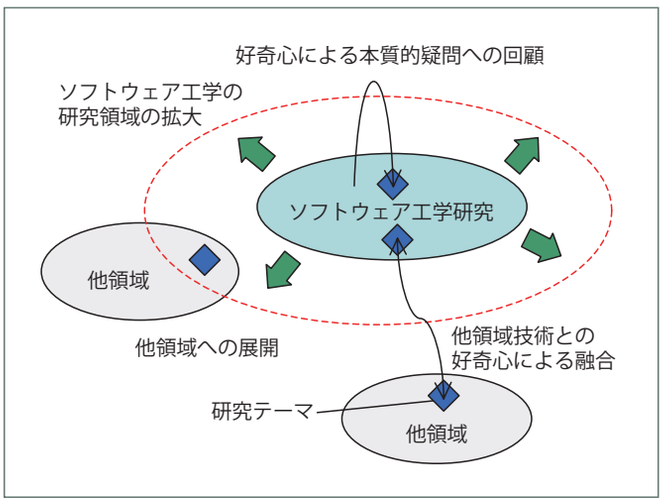


図 -3　ソフトウェア工学研究の将来（その 1）

## ■ 日本のソフトウェア工学研究の将来（その1）

日本のソフトウェア工学の将来を考える上で，まず，Osterweil 氏の意見を取り上げてみたい．彼の示した 1 つ目の方向性は，日本のソフトウェア工学研究の状況においても，そのまま当てはまる．日本には，世界的に見ても非常に高いレベルにある技術が数多く存在する．それらの技術領域に対するソフトウェア工学研究成果の適用の試みや，他領域における課題に対するソフトウェア工学的解決の試みは，今後も数多く実行されるであろう．

筆者は，好奇心ドリブン研究に関しては，Osterweil 氏と少し異なる印象を持っている．彼は，ソフトウェア工学における本質的な疑問に着目することを，ソフトウェア工学の将来の方向の 1 つであると考えている．個人的意見になるかもしれないが，筆者のまわりには，このような疑問を常に持ちつづけている（特に，日本人）ソフトウェア工学研究者は多い．よって，このような好奇心ドリブン研究は，日本において今より流行するとは考えていない．

一方，日本人研究者には，試行的好奇心[☆6]が少し足りないように感じている．試行的好奇心とは，なんでも強引に試してみるという感覚を指す[☆7]．たとえば，セキュリティの概念を UML に押し込んでみたり，Web 技術とソフトウェア技術をとりあえず混ぜ合わせてみたりする強引さは，日本のソフトウェア工学研究にはあまり見られないし，研究としてなかなか認められにくい．他領域への進出に応じて，課題ドリブン研究を続けるのも 1 つの選択枝である．しかしながら，ソフトウェア工学研究の領域の拡大をまたず，ソフトウェア工学における研究テーマや研究成果を他領域の技術と，好奇心に任せてとりあえず融合させてみることを，日本のソフトウェア工学研究における好奇心ドリブン研究と捉えるのはどうだろうか．

**図 -3** に，これまでの意見をまとめた．各領域（楕円で表現）内のひし形（青色）は研究テーマを表す．「他領域への展開」と「好奇心による本質的疑問への回顧」は，Osterweil 氏の提案である．一方，「他領域技術との好奇心による融合」は，筆者の提案を指す．

もちろん，研究テーマや技術の融合により高い成果を出すためには，個々の研究に関する十分な取り組みが必要であることはいうまでもない．しかしながら，そのような研究だけにこだわらず，組合せによる新しさを許容する（ときには，徹底的に議論する）雰囲気をソフトウェア工学コミュニティ内部に育てることが大事である．

## ■ 日本のソフトウェア工学研究の将来（その2）

個人的な観点での主張をもう少し続けると，現在のソフトウェア工学研究がソフトウェア工学的でない点に少し不満を感じている．ソフトウェア開発が高度な知的作業であることはすでに述べたが，ソフトウェア工学の研究活動も高度な知的作業である．さらに，ソフトウェア工学研究の成果は，単なる研究者の創作物でなく，ソフトウェア開発や生産したソフトウェアを介して現実世界に作用する．このように考えると，少なくともソフトウェア工学研究においては，ソフトウェア工学的な考え方を採り入れてもよいのではないだろうか．

たしかに，研究活動に生産性という概念はそぐわないかもしれない．しかしながら，ソフトウェア工学を取り巻く環境の変化の速さや，世界的傾向である成果発表までに費やす時間の短縮化という現状を考えると，迅速な成果発表という観点での生産性と成果に対する品質とのバランスを重視した研究活動は必須である．たとえば，部品化再利用（コンポーネント指向）技術はソフトウェア開発の生産性の向上とソフトウェアの品質の向上を同時に探求した技術であるが，ソフトウェア工学研究による成果（技術）を適切にモジュール化しておくことで，技術のブラックボックス的再利用やインタフェースによる組合せが可能となるかもしれない．組み立て可能かつ再利用可能な技術が複数存在すれば，それらの組合せを試してみることは自然である．これは，前節で述べた好奇心ドリブン研究と相性がよい．さらに，ソフトウェア工学研究の成果に対して，特定の成果や分野に依存しない共通の評価実験や検討事項と，個別の成果や分野に応じてカスタマイズが必要な評価実験や検討事項を認識し，それぞれフローズンスポットとホットスポットとし

☆6 筆者の作成した造語である．

☆7 このような好奇心が海外で認められているのかどうかは分からない．

て整理することが考えられる．このように構築した技術成果のフレームワークを用いることで，既存研究を拡張した場合，どのような評価実験や検討を行えばよいのかが明確になるであろう．

また，ソフトウェア工学分野における各技術の位置付けや役割を UML やパターン言語を用いてモデリングするという試みも興味深い．特集「要求工学」における文献 5）では，要求工学に関するメタモデルの構築を試みているが，このような活動をソフトウェア工学の全分野に対して行っていくべきであろう．さらには，アジャイルソフトウェア開発の研究や実践を通して得られた知見や技術を，ソフトウェア工学研究活動自体にも転用するというのはどうだろうか．開発に顧客を取り込むことは，開発者を研究活動に取り込むことに似てはいないだろうか．

### ■ 日本のソフトウェア工学研究の将来（その3）

最後の主張は，やや挑発的である．ソフトウェア開発は高度な知的活動に大きく依存しているという事実を正しく受け止めると，ソフトウェア開発には，今以上に多くの熟練開発者や職人気質が必要になるであろう．もしソフトウェア工学が，このような状況に対して何もできないのであれば，ソフトウェア工学の外部に，新たな分野（たとえば，ソフトウェア開発学）を立ち上げるというのはどうだろうか．もちろん新しい分野を立ち上げることが根本的な解決であるとは考えてはいないが，単にソフトウェア工学を否定したり，現状に嘆いていたりするよりは前向きである．ソフトウェア開発において，開発者個人の活動や開発者間のコミュニケーションを最優先に考え，職人（ハッカー）が利用したくなるような技術やツールを提供していくことを主体とした研究は楽しそうである．また，ソフトウェア開発自体を楽しいものに変えてくれる技術やツールを考えていくこともおもしろい．

たとえば，開発中の意見や感想（文字だけでなく音声や画像）をプログラム中に自由に記録できるツールと，それを開発者の blog などを通して自由に閲覧できる仕組みがあると開発が楽しくなるかもしれない．また，自分の記述したプログラムやその開発履歴を公開することで，同じようなプログラム／プログラミングスタイルを持つ仲間を世界中から自動的に探し出して，紹介してくれる仕組みはどうだろうか．

個人的には，（日本の）ソフトウェア工学コミュニティは，新たな分野の立ち上げや参入に好意的であると信じている．また，多くのソフトウェア工学研究者は，ソフトウェア開発の楽しみを知っているはずである．よって，ソフトウェア工学分野の外部ではなく，その内部にこのようなコミュニティが形成される可能性は高いと感じている．

以上，日本のソフトウェア工学の将来に対する個人的な意見を述べた．本稿で述べてきたように，ソフトウェア工学研究には，取り組むべき課題が数多く残っており，その数はますます増加している．同時に，ソフトウェア工学研究では，さまざまな知識や分野背景を持つ研究者との対話や参入を求めている．

ソフトウェア工学の研究者として成功するには，自分の専門を深く掘り下げるのと同時に，自分の専門の外にも目を向けて，自分の成果との融合の可能性を求めるという姿勢が必要である．技術の融合は，論文を読むだけでは生まれない．自分の想像力を信じ，自由な発想や好奇心を持って研究テーマを探し，それをコミュニティ内で徹底的に議論していくことが重要である．つまり，研究者どうしの発想や好奇心をぶつけ合う自由な場や，研究者と実際の開発者とが本音で対話できる場が必要である．

最後になるが，本稿がソフトウェア工学の発展に少しでも寄与できれば幸いである．

**謝辞**　本稿の内容は，筆者が今までに議論や討論を行ってきた数多くの研究者，開発者，学生などに大きく影響を受けている．情報処理学会ソフトウェア工学研究会主催のシンポジウムやワークショップにおいて，議論・討論にお付き合いいただきました皆様に感謝いたします．また，本稿を執筆するにあたり参考となる講義資料を快くご提供いただきました，国立情報学研究所の本位田真一教授に感謝いたします．

**参考文献**

1）鯵坂恒夫：ソフトウェア工学入門，サイエンス社（2008）．
2）McBreen, P.：Software Craftsmanship：The New Imperative, Addison Wesley（2001）（村上雅章訳：ソフトウェア職人気質，ピアソン・エデュケーション（2002））．
3）Briand, L. C. and Wolf, A. L., Ed.：Proc. Future of Software Engineering（FOSE'07）, ICSE 2007（2007）．
4）Osterweil, L. J.：A Future for Software Engineering ?, Proc. FOSE'07 of ICSE 2007（2007）．
5）鎌田真由美：要求工学の現状と課題，情報処理，Vol.49, No.4, pp.347-356（Apr. 2008）．

（平成 20 年 4 月 30 日受付）

**丸山勝久（正会員）**
**maru@cs.ritsumei.ac.jp**

1991 年早稲田大学理工学部電気工学科卒業．1993 年同大理工学研究科修士課程修了．同年，日本電信電話（株）（NTT）入社．2000 年より立命館大学助教授，2007 年より同大教授．現在，総合理工学院情報理工学部に所属．博士（情報科学）．ソフトウェア開発環境，ソフトウェア再利用，プログラム解析の研究に従事．