① 共通問題の作成
～ワークショップを通して～

基専

丸山勝久（立命館大学情報理工学部）
鵜林尚靖（九州大学大学院システム情報科学研究院）

共通問題の作成を目指して

ソフトウェア工学の共通問題の作成は簡単なのだろうか．これは，2013 年 9 月号の特集「ソフトウェア工学の共通問題」を読んだ後の，筆者らの素直な感想である．多くの読者が同じような感想を持ったかもしれない．あるいは，共通問題の存在を知って，自分でも作成できると考え，その作成を試みた読者がいるかもしれない．

筆者の1人である丸山は，「酒屋問題再考」の記事[1]で，新たな共通問題の作成を目指して考慮すべき 4 つの視点（開発工程，意思決定，開発活動，成果表現）を示した．これらが現状のソフトウェア工学で必要な視点を網羅しているかどうかについては議論が必要だが，約 30 年前の記事[2),3)]のように単一の視点だけで共通問題を作成することが適切でないことを示している．ただし，多様な視点を提示しただけでは，共通問題を示したことにはならない．むしろ，このような提示により，共通問題を作成することの困難さが露呈した感がある．

このような議論が，ソフトウェア工学研究会主催の 2014 年 1 月のウィンターワークショップの「ソフトウェア工学の評価」セッションで行われた．その結果，ソフトウェア工学の共通問題を作成するためには開発のコンテキストに合わせた視点を設定することが重要であるという意見が出された．ここで，開発コンテキストとは，開発対象ソフトウェアの種類（ドメイン）を指すのではなく，開発における方針のようなものを指す．たとえば，プロダクトライン開発（再利用開発），オープンソース開発，アジャイル開発，ハード／ソフトウェア協調開発，オフショア開発，派生開発（レガシー活用）などである．もちろん，開発コンテキストには，開発対象ソフトウェアに対するステークホルダ（利害関係者）も含まれる．

ウィンターワークショップでは，ソフトウェア工学の共通問題の作成にあたり，視点を設定することの重要性は認められるものの，開発のコンテキストから切り離して視点を設定することはできないのではという点が指摘された．言い換えると，共通問題が対象とする開発のコンテキストを決めることで初めて，視点の捉え方や優先度が設定できるのではないかという指摘である．このような指摘を受け入れると，共通問題を作成することの困難さは，複数の視点が存在することだけではなく，複数の視点の混ざり合い具合の多様性に起因しているのではないかと推測できる．

なぜワークショップなのか

いずれにしろ，共有問題の作成に関して開発コンテキストという切り口が与えられたので，これを足掛かりに共通問題の作成に関する議論の展開が期待できる．しかしながら，開発コンテキストの多様性は 1 人の研究者，実践者，教育者で対応できるものではない．そこで，2013 年の「ソフトウェア工学の共通問題」の記事執筆者，および，ウィンターワークショップにおける「ソフトウェア工学の評価」セッションの参加者に呼びかけ，2014 年 5 月 18 日にワークショップを開催した．ワークショップの参加者を次に示す．

岸知二（早稲田大学）
野田夏子（芝浦工業大学）
細合晋太郎（九州大学）
沢田篤史（南山大学）
位野木万里（工学院大学）
鵜林尚靖（九州大学）

- **立場**
  - ◆何を重視しているのか
  - ◆何を達成あるいは推進しようとしているのか
- **評価対象**
  - ◆どのような技術があるのか
  - ◆どんな技術を比較するのか
- **評価項目**
  - ◆どのような観点で比較するのか
  - ◆善し悪しの判断は

図 -1　共通問題作成フレームワークの形式

- **立場**
  - ◆派生開発の推進
  - ◆適応保守／完全化保守の推進
- **評価対象**
  - ◆リバースエンジニアリング技術
  - ◆リエンジニアリング技術
- **評価項目**
  - ◆ソフトウェアの改変作業に必要な労力は少ないか
    - ✓理解しやすいか
    - ✓変更個所を特定しやすいか
    - ✓プログラムが変更しやすいか
    - ✓変更時にテストがしやすいか

図 -2　保守性に関する共通問題作成フレームワークの例

丸山勝久（立命館大学）

開催の準備期間が短かったこと，また日曜日に開催したことで，残念ながら企業からの参加者はいなかった．

議論が発散しないように，ワークショップでは，開発コンテキストを明確にした議論をお願いした．一方，共通問題の作成に関する方針そのものも討論対象としたため，共通問題の作成に関するさまざまな意見が交わされた．

共通問題に関する議論

ここでは，共通問題の作成に関して，ワークショップで取り上げられた議論のうち，3 つを紹介する．ここで紹介する 3 つは，筆者らが勝手に選択したものである．また，ここで述べる内容に関しては，必ずしもワークショップの参加者で合意がとれているわけではない．ソフトウェア工学分野では共通問題をどのように考えているのかの参考意見だと受けとっていただきたい．

- **立場**
  - ◆プログラミング速度の加速
- **評価対象**
  - ◆プログラム開発環境
  - ◆ソフトウェア検索ツール
- **評価項目**
  - ◆プログラム記述に必要な労力は少ないか
    - ✓コードの共有ができているか
    - ✓コード検索やコード補完が有効に働くか
    - ✓バグ修正やコード改変に関する情報が容易に得られるか

図 -3　開発効率に関する共通問題作成フレームワークの例

❖ 共通問題作成フレームワーク

共通問題の作成を目的としたフレームワークが提案された．ここでは筆者らの示した 2 つのフレームワークを紹介する．

まず丸山が提案した共通問題フレームワークを示す．**図 -1** は，フレームワークの形式である．共通問題が扱う開発コンテキストを，立場，評価対象，評価項目で捉えている．立場とは共通問題を利用する意義を指す．評価対象と評価項目は，共通問題の利用方法に関する記述を指す．

この形式に基づき作成した 2 つの共通問題作成フレームワークを**図 -2** と**図 -3** に示す．これらのフレームワークでは，保守性や開発効率という開発コンテキストにおいて，共通問題を作成する際に明確にすべき観点が示されている．その反面，具体的な共通問題を作成する際には，開発対象ソフトウェアに関する要求仕様や設計を，フレームワークに与える必要がある．また，評価項目に記述された内容を確認するための作業の詳細を決定しなければならない点にやや難がある．

これに対して，鵜林は組合せ型の共通問題作成フレームワークを提示した（**図 -4**）．文献 1）で示した 3 つの視点（開発工程，意思決定，開発活動）に「技術」を加えた合計 4 つの視点と，ベース問題を組み合わせることで，共通問題を作成するフレームワークである．

ここで，「技術」という視点に関しては少し説明が必要であろう．開発工程，意思決定，開発活動の視点で扱っている内容は現在のソフトウェア工学（の教育）で一般的な概念，あるいは一般的になりつつある

概念で構成されている．それぞれの視点における分類が，今後急激に変化することはないであろう．これに対して，技術で扱う概念は，いまだ研究の初期段階にあるものや，実際の開発現場および保守現場での適用が少ないものを指している．これは，研究コミュニティにおいて共通問題を作成する場合を想定している．

このフレームワークを適用することで作成した共通問題の具体例を**図 -5** に示す．視点やベース問題を自由に組み合わせることで，さまざまな共通問題を作成することができる．

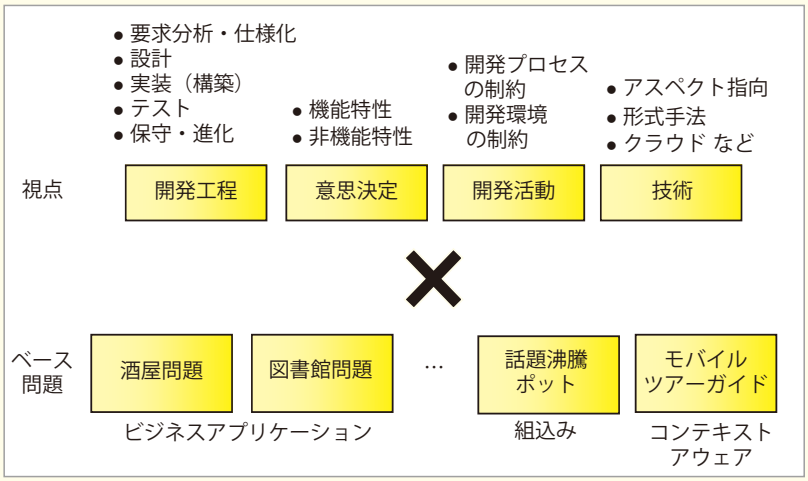

図 -4　組合せ型共通問題作成フレームワーク

- **酒屋問題 × 要求仕様化**
  - ◆酒屋問題には曖昧な要求記述が含まれている．どの記述が曖昧かを指摘するとともに，その修正方法を示せ．
- **図書館問題 × 非機能特性**
  - ◆図書館問題において，スケーラビリティを意識したアーキテクチャ決定を示せ．
- **話題沸騰ポット × 再利用**
  - ◆話題沸騰ポットのプロダクトラインを設計せよ．その際，非機能特性をフィーチャーモデルで示せ．
- **モバイルツアーガイド × アスペクト指向**
  - ◆モバイルツアーガイドをアスペクト指向で設計せよ．また，アスペクト指向設計の結果を，オブジェクト指向設計の結果と比較せよ．
- **図書館問題 × 設計 × アジャイル**
  - ◆図書館問題をSCRUM（アジャイル手法）で開発した場合のプロダクト・バックログを示せ．

図 -5　組合せ型共通問題作成フレームワークにより作成した共通問題の例

2 つの共通問題作成フレームワークを比較してみると，前者のフレームワークは共通問題の利用を強く意識していることが分かる．共通問題を作成するにあたり，その共通問題を利用することで明らかにしたいこと（共通問題に求められる要件）をまず明確にすべきという考えに基づく方法である．一方，後者のフレームワークは共通問題の作成のしやすさを重要視した方法となっている．ソフトウェア工学におけるさまざまな手法や技術を比較するための共通問題を作成するという点では，手軽で適用しやすいフレームワークといえる．共通問題に求められる要件を最初から厳密に定義せず，具体的な共通問題を集めることで，新たな知見を得る．このような作業の繰り返しにより共通問題に求められる要件を段階的に定義していくことで，共通問題の洗練を行うことができる点が強みである．

❖ 誰のための共通問題か

ワークショップでは，プロダクトライン開発やアーキテクチャ設計（モデリング）という開発コンテキストにおける共通問題の作成に関する議論に多くの時間が割かれた．その際，参加者から，正確性だけでなく合目的性を共通問題で扱うことの重要性が指摘された．

正確性や合目的性とは，ソフトウェアの品質特性（ISO 9126）の機能性に関する副特性である．機能性とは，ソフトウェアが必要な機能を実装している度合いを指す．正確性とは，ソフトウェアの実装がその設計仕様をどの程度満たしているかどうかを指す．これに対して，合目的性とは，ソフトウェアの設計仕様が利用者の目的をどの程度満たしているのかを指す．合目的性が高ければ，ソフトウェアが利用者の作業および利用者の具体的目標に対してより適切な機能を提供していることになる．その結果として，利用者の作業効率や目標達成度が高まるはずである．

利用者の満足度を意識した合目的性の導入は，共通問題を用いてソフトウェアの品質を比較する際に有効である．産業界からは，合目的性を意識した共通

問題は大いに歓迎されるだろう．

一方，そのような共通問題は，ソフトウェア工学の技術や手法を比較することには適さないのではという意見もあった．ソフトウェアには利用者だけでなく，開発者や保守者など多様なステークホルダがいる．このため，利用者視点の合目的性という観点だけで評価すると，他のステークホルダにとっての観点を捉えることができないからである．たとえば，図 -3 にある「コードの共有ができているか」という評価項目は，合目的性という観点からは思いつかない．また，ソフトウェアの利用者だけを見ていても，研究の促進，研究成果の普及，教育効果を目的とした共通問題の存在には気付かないだろう．

このように，ソフトウェア工学の共通問題における利害関係者は，共通問題が対象としているソフトウェアの利害関係者よりも広い．ソフトウェア工学の共通問題を作成する際には，ソフトウェアの利用者に加えて，実践者（開発者や保守者），研究者，教育者，教育を受ける人（学生など）の存在も十分に検討することが大事である．

❖ 共通問題の新たな意義

（ソフトウェア工学だけではないが）論文を読んでいると，「妥当性への脅威（threats to validity）」という用語をよく見かける．被験者の心理的バイアスによる影響や被験者の偏りによる影響などが，評価実験結果の妥当性（再現性）に対する脅威となる．

ここで，ソフトウェア工学が対象としているソフトウェア開発や保守は通常，人間の行う作業である．このため，開発作業や保守作業の支援を目的とした研究開発では，その評価実験の多くの場合において，妥当性への脅威がつきまとう．妥当性への脅威が存在する以上，同じ開発や保守を想定しても，同じ実験結果が得られるとは限らない．このような状況においては，追試（追実験）が有効である．

追試の必要性は理解されつつあるものの，現時点でソフトウェア工学の研究において追試が実施されることは非常に少ない．理由の 1 つとして，現状では研究（論文）ごとに異なる評価実験が行われることが多いことが挙げられる．この場合，追試を行おうとしても，同じ実験環境を構築するのに非常に手間がかかる．

ソフトウェア工学分野において共通問題が整備されると，このような状況を打破できる可能性が高まる．評価実験において同じ共通問題を利用することで，実験環境の再利用が可能となる．つまり，共通問題の作成を，技術や手法を比較するという目的で捉えるのではなく，追試の際の実験環境の共有化として捉える．共通問題により追試が手軽に実施できるようになることで，追試のコストを大幅に下げることができ，追試の促進が達成される．これにより，より多くの実証データの収集が期待でき，ソフトウェア工学の成果がより産業界に受け入れられる可能性が高くなると考えてよいだろう．

共通問題の作成にとりかかろう

ワークショップを通して得られた筆者らの意見を紹介した．この記事が，共通問題の作成を後押しすることを望む．また，他分野においても，共通問題を考えるきっかけになれば幸いである．

参考文献

1) 丸山勝久：酒屋問題再考―新たな共通問題の作成を目指して―，情報処理，Vol.54, No.9, pp.886-889（Sep. 2013）．
2) 山崎利治：共通問題によるプログラム設計技法解説，情報処理，Vol.25, No.9, p.934（Sep. 1984）．
3) 山崎利治：共通問題によるプログラム設計技法解説（その 2），情報処理，Vol.25, No.11, p.1219（Nov. 1984）．

（2014 年 7 月 4 日受付）

丸山勝久（正会員）maru@cs.ritsumei.ac.jp

1993 年早稲田大学理工学研究科修士課程修了．同年，日本電信電話（株）（NTT）入社．博士（情報科学）．2000 年より立命館大学．ソフトウェア保守と進化，ソフトウェア開発環境の研究に従事．

鵜林尚靖（正会員）ubayashi@acm.org

1982 年広島大学理学部数学科卒業．1999 年東京大学大学院総合文化研究科広域科学専攻博士課程修了．博士（学術）．（株）東芝，九州工業大学を経て，2010 年より九州大学教授．2013 年より本会ソフトウェア工学研究会主査．