事故前提社会に向けたユーザ・ベンダ間での開発データ共有 第3回
―ソフトウェアタグ普及に向けた法的議論と利用技術基盤―

奈良先端科学技術大学院大学
先端科学技術研究調査センター
教授
久保　浩三

奈良先端科学技術大学院大学
先端科学技術研究調査センター
研究員
小柴　昌也

奈良先端科学技術大学院大学
情報科学研究科
特任助教
角田　雅照

奈良先端科学技術大学院大学
情報科学研究科
研究員
松村　知子

この解説では、文部科学省StagEプロジェクト※1の概要を紹介すると共に、同プロジェクトが作成したソフトウェアタグ規格と、ソフトウェアタグ支援ツールについて述べてきた。最終回となる今回は、ソフトウェアタグ普及に向けたより実践的な取り組みとして、ソフトウェアタグの意義と技術的課題についての法的観点からの議論、及び、ソフトウェアタグ利用の技術基盤となるユーザ・ベンダ協調型プロジェクト管理とソフトウェア開発データ分析モデリング言語について解説する。

1. はじめに

ソフトウェアタグとは、ソフトウェア開発に関する実証データから、ソフトウェアやその開発プロジェクトの特徴量を算出し、ユーザにも理解しやすく、可視化や評価にも利用しやすい形式でとりまとめた情報パッケージである。ソフトウェア開発終了後にソフトウェア製品に添付され、ベンダからユーザへ提供されることになるが、開発途中に進捗報告書に添付するといった利用形態も考えられる。ユーザ・ベンダ間でのデータ共有のメディアとも言えるが、実開発プロジェクトに適用するためには、開発データ共有の目的や目標といった抽象的な概念を、ソフトウェアタグ規格で規定された具体的なデータに対応付けたり、タグ収集・可視化・評価ツールを活用したりするためのより実践的な技術が必要となってくる。

本技術解説では、ソフトウェアタグ普及に向けた法的観点からの議論と2つの利用技術基盤について概説する。これらは、主にソフトウェアタグ利用シナリオとして具現化することになるが、ソフトウェアタグ規格やソフトウェアタグ支援ツールにも今後反映されていくものである（図1）。

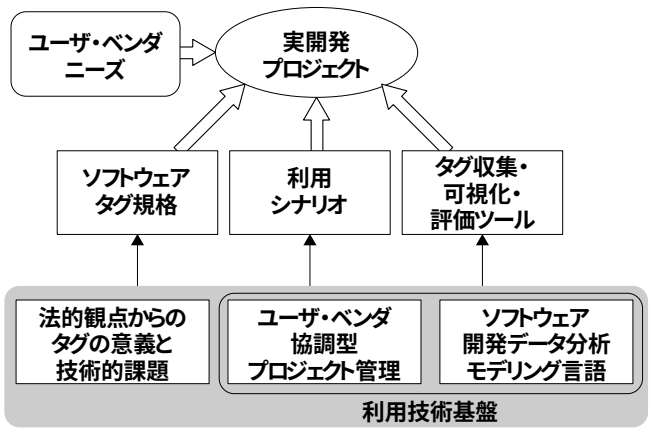

図1　ソフトウェアタグ技術を支える法的観点からの議論と利用技術基盤

2. ソフトウェアタグに対する法的議論

2.1　目的と背景

ソフトウェア開発を成功させるためには、要件定義、レビュー等、すべての段階で注意が必要であり、ユーザ

※1　StagEプロジェクト：Software Traceability and Accountability for Global software Engineering：エンピリカルデータに基づくソフトウェアタグ技術の開発と普及

とベンダの双方において、契約の明確化、プロセスの可視化、人材育成等のいろいろな試みが行われている。しかし、ソフトウェア開発において、品質、コスト、納期のすべてにおいてユーザ・ベンダ間での紛争が多発しており、現状においては減少傾向にはなく、問題がますます複雑化してきているようにも見受けられる。

ソフトウェアタグに対する法的観点からの議論の目的は、ソフトウェア不具合時の事後的な紛争処理を分析することで、紛争を未然に防止することにある。また、ソフトウェアタグを、プロジェクト管理に用いるだけでなく、ユーザとベンダで共有し、これを活用することによって、紛争処理に係る労力を軽減することも目的の1つである。

2.2 技術と法律の連携

ここで紹介する法的議論は、ソフトウェアタグに関する技術的議論との連携を目指したものである。例えば、後述するように、ソフトウェア開発における紛争の分析結果に基づいて、ソフトウェアタグへの要望をまとめている。技術と法律の連携を深めるため、法学部教授、弁護士等の法律の専門家、そして、ソフトウェアタグの技術的専門家の双方で構成される「ソフトウェア構築可視化に伴う法的諸問題委員会」を設置している。

このような技術と法律による連携は、一般的に見られるものではない。確かに、これまでも、建設に係る法、通信に係る法、情報と法、原子力と法等にも技術と法律の連携は見られる。しかし、これらは、新たな技術を社会に適用する場合に生ずる種々の問題を調整するために、法律を制定し、その交通整理を行うことが主である。

今回用いた手法は、ソフトウェア開発における紛争を紹介、分析することで法律家が関与し、新たな技術課題解決による新技術の開発手法の提案を行うという新規なものである（図2）。この手法が、他の技術分野でも応用出来るかどうかの検討も今後行っていく。

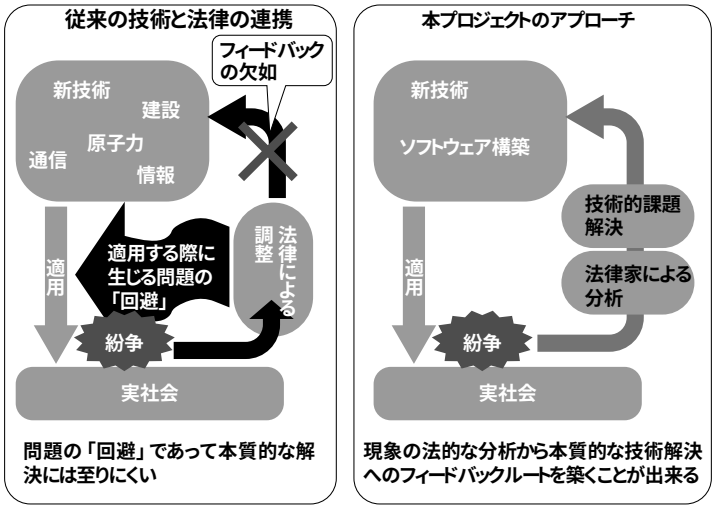

図2　法的分析から技術的課題解決へのループ形成

2.3 具体的な取り組み

2.3.1 裁判例の抽出と分析

分析対象としたのは、平成2年以後のソフトウェア開発委託取引を巡る20件の裁判例である。裁判事例は、TKC法律情報データベースにおいて、以下のキーワードにより抽出した。

（瑕疵（かし）＋仕様＋開発）×プログラム＝41件
（瑕疵＋仕様＋開発）×ソフトウェア＝26件
（瑕疵＋仕様＋開発）×ハードウェア＝13件

ここから知的財産権に係る紛争を除外し、さらなる調査により4件を追加した。

これらの裁判例を分析すると、ユーザのシステム開発要求をベンダに伝えることの困難さに紛争の原因の1つがあることが分かった。通常、ソフトウェア開発は、RFP※2により、ユーザの要求がベンダに伝えられる。また、RFPを更に具体化した要求仕様書が用いられることもあり、これに基づいてベンダが要件定義を作成する。しかし、ユーザの要求をすべて伝えることは容易ではない。

また、ユーザからベンダに対して、開発工程の段階でも仕様変更やユーザの要求が提示されることが多く、当初よりも見積りが大きく膨れ上がり、また納期も延びることとなる。ソフトウェア開発

※2　RFP：Request For Proposal，提案依頼書

委託契約は、いわゆる請負型で行われることが多く、ユーザは、自分の要求が満たされないと、業務が完成していないとして支払いを拒むケースがあり、紛争の元となっているように見受けられた。

2.3.2 アンケート調査

以上のように、ソフトウェア開発委託取引を巡る裁判事例は、非常に少ない。そこで、判例分析で導き出された観点である「ユーザの要求がベンダに正確に伝わっているか」という点についてアンケート調査を行った。調査では、ソフトウェア開発費の見積額と開発に実際に要した経費との差異（見積り差異）に着目した。2008年12月から2009年3月にかけて、インターネットも活用したアンケートの結果、ユーザ91名、ベンダ224名、合計315名から回答を受け取った。得られた主な結果は次の通りである。

- ベンダの回答によると、ソフトウェア開発における見積り差異は、人月計算で1.8倍、金額計算で2倍であった
- 見積り差異が生じたプロジェクトでの見積り方法は、「大まかな業務の説明」に基づくものがベンダ41%（最も多い原因）、ユーザ24%（2番目に多い原因）であった（図3）
- 見積り差異の発見段階は、ベンダでは「詳細設計」（21%）を筆頭に、「基本設計」（19%）、「要件定義」（13%）という意見が多く、プログラミングに着手するまでの早い段階で問題が発覚していることが多い（図4）

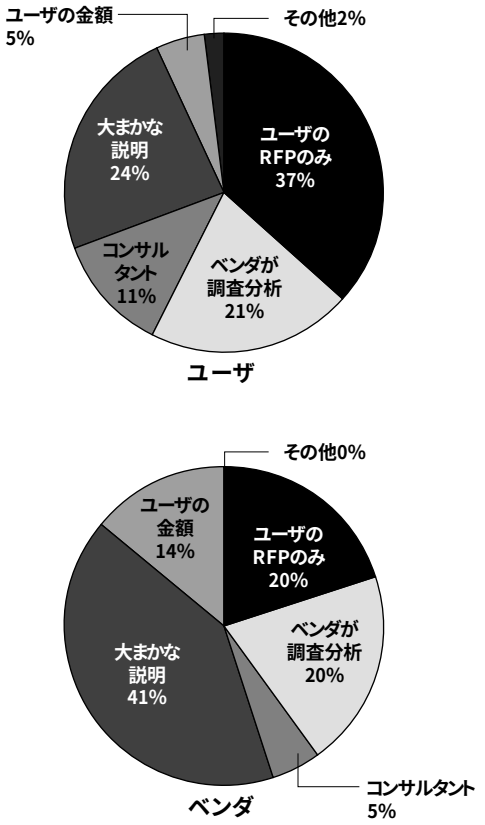

図3　見積り差異が生じたプロジェクトでの見積り方法

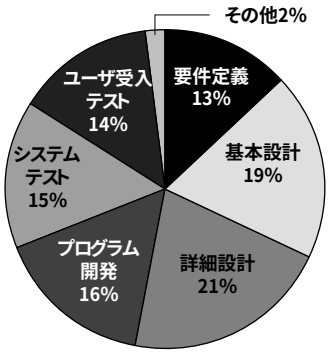

図4　見積り差異の発見段階

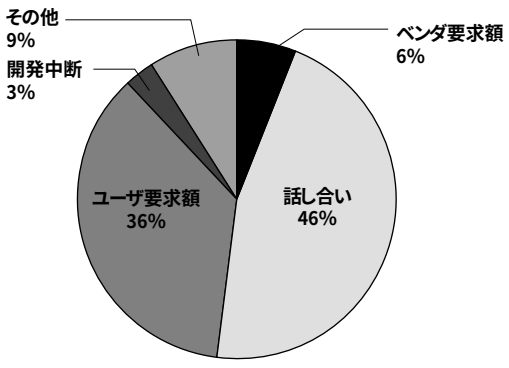

図5　見積り差異の解決手段

・見積り差異の原因は、ユーザ・ベンダ共に「現場からの予想外の要求」（ユーザ30%、ベンダ27%）が、最も多い
・見積り差異の解決手段は、「話し合い」（46%）を除き、ベンダが「契約通りの金額しか請求しなかった」（36%）が最も多かった（図5）

このように、アンケート全体を通じて、ユーザとベンダ間には意見の隔たりがあることが分かった。また、見積り差異が生じる原因として、あいまいな受発注による契約が根底にあることも分かった。

2.4　ソフトウェアタグの意義と技術的課題

裁判例等の紛争やアンケート回答を分析していくと、ソフトウェア開発における紛争は大きく2つの型に分類出来ることが分かった。そして、法的議論の1つの結果として、それら2つの型それぞれにおけるソフトウェアタグの意義と技術的課題（技術的議論への要望）が明らかとなった。詳しくは表1を参照されたい。

ソフトウェアタグの内容をどのように要求するかは、契約書の一部分として定めることであり、例えば、段階的な請負契約をすることを業界標準として定めたときに、その段階ごとに、当事者が内容を確定するために、どのようなソフトウェアタグを必要とするかを検討する必要がある。今後は、当事者自身が、ソフトウェアタグをどのように活用していくかの観点から契約とソフトウェアタグについて法的議論を深めていく予定である。

3. ユーザ・ベンダ協調型プロジェクト管理

本章では、ユーザ・ベンダにWin-Winの開発管理形態を実現するための枠組みの2つのポイント、ユーザ・ベンダ協調型プロジェクト管理サイクルと、このサイクルを実現するための管理プロセスの構築手順について説明する。前述した法的紛争の事例や現場アンケートからも、ユーザのプロジェクト管理における責任や役割の重要性は明確であるが、既存のソフトウェア開発の定量的管理の規格やガイドライン［CMMI2007］［PMBOK2004］は、いずれも開発組織の視点で作成され、ユーザは管理の対象の一部として扱われている。紹介する枠組みは、双方にとって透明性の高い管理プロセスを実現し、問題の早期発見・解決を支援する。

表1　ソフトウェア構築に関する法的諸問題の分析

紛争の型	法的責任	ソフトウェアタグの意義	ソフトウェアタグの技術的課題	今後の課題
未完成または欠陥により使用不可	ユーザまたはベンダの債務不履行責任	・ユーザに進捗を見せ、相互に管理を行う。 ・開発が順調に進んでいることをユーザに知らせる。 ・開発が順調に進んでいない場合の軌道修正を容易にする。	・ユーザの要望（要求定義）がベンダに伝わっているかどうかを明らかにすること。 ・実績が予定通り進んでいること（予実管理）を行えるようにすること。	・実証実験により、基準値、標準曲線を作成し、ユーザの視認性を高めること（例：人間ドックにおける基準値）。
使用可ではあるが、完成後のバグ発生または障害発生	ベンダの不法行為責任または債務不履行責任	・事故があった場合のトレースを容易にする。 ・早期復旧を容易にする。	・システムがダウンする前に、どこにバグまたは欠陥があるかのおおよその当たりをつけること。	・システムがダウンした場合の原因、修正履歴を残し、障害の発生率を下げることが出来るようにすること。

3.1 協調型プロジェクト管理サイクル

図6はPMBOK［PMBOK2004］に基づく従来のプロジェクト管理サイクルである。ユーザは、サイクル外に位置付けられ、ベンダから提供されるデータに対してチェックや要望を出す。これに対して、図7の我々が提案する協調型プロジェクト管理サイクルでは、計画・監視・コントロールのプロセスに関して、両者の合意・相互確認・各組織での合意に基づいた適切な是正処置を行うことを要求する。

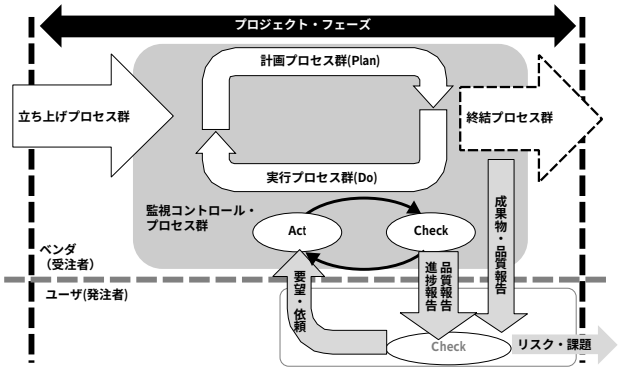

図6　従来のプロジェクト管理サイクル

3.2 管理プロセスの構築手順

次に、合意形成プロセスについて、我々はCMMI［CMMI2007］のプロセス領域「定量的プロジェクト管理」を参考に以下のような定量的プロジェクト管理プロセスの構築手順を提案する（図8）。

① プロジェクト管理目標を設定し、両者で合意
② ユーザ・ベンダ別に、目標に関わる作業（サブプロセス）を定義
③ 作業でユーザ・ベンダ間のコミュニケーションや共有される（べき）データを明確化
④ ②③に基づき、定量的に管理するフェーズを構成
⑤ 管理フェーズごとに定量的管理に関する目標、尺度、データの収集と格納方法、分析方法等で測定モデルを作成。このステップについては、前号で紹介したタグデータの事前選定と計測計画立案ツール「タグ・プランナー」を適用可能

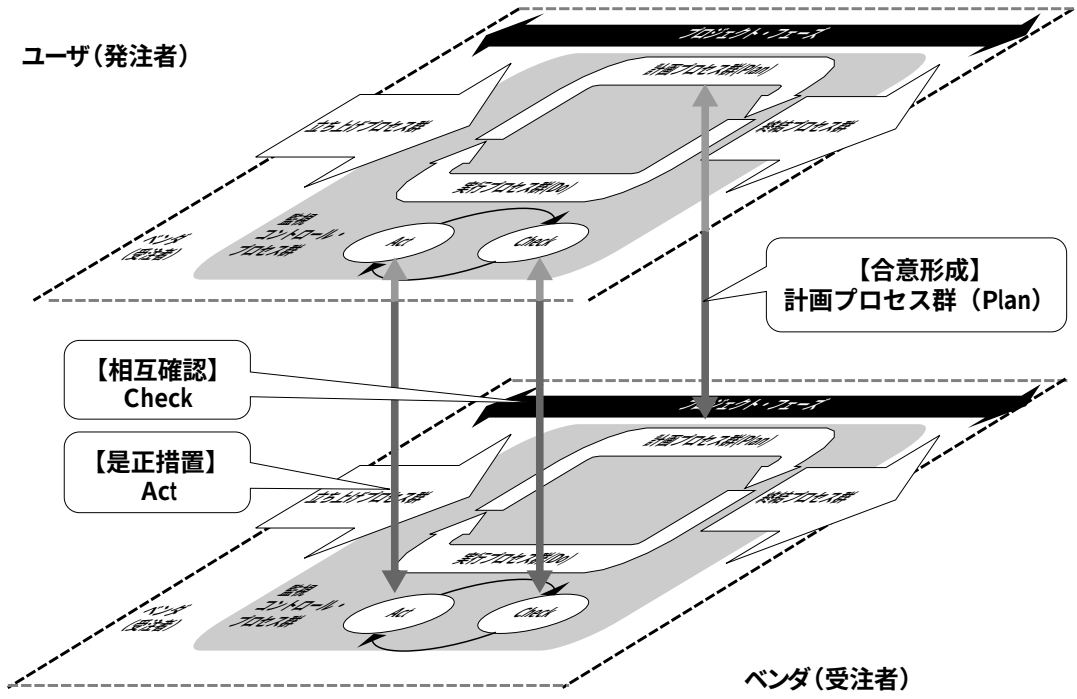

図7　提案する協調型プロジェクト管理サイクル

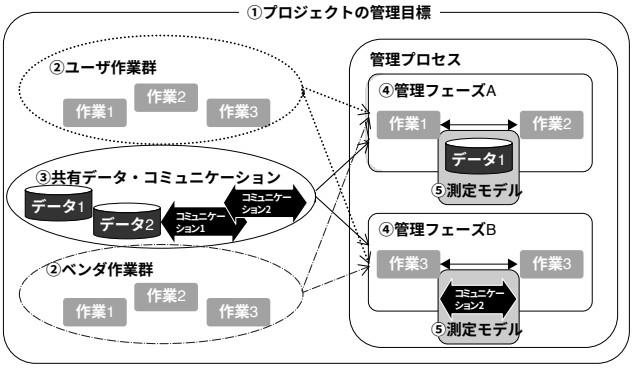

図8　定量的プロジェクト管理プロセスの構築手順

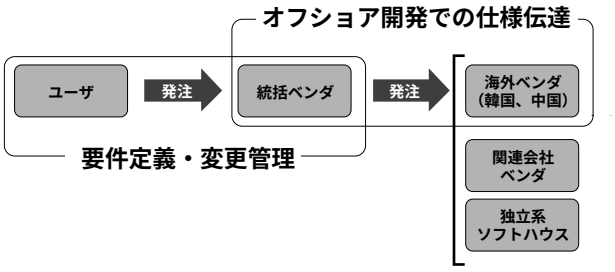

図9　実プロジェクトでの検証

3.3　実プロジェクト開発情報に基づく検証

上記の枠組みについて、数社から実プロジェクトのデータをご提供頂き、定量的データ分析とその結果に基づいた開発関係者へのヒアリングを繰り返し、管理プロセスの構築と実用性の検証を行った（詳細は、[松村2009-1][松村2009-2]）。図9は検証に用いた2つのプロジェクト体制（プロジェクトは異なる）と、おのおののプロジェクトの管理目標を示す。対象プロジェクトでは、データ共有やコミュニケーションは十分に行われていて、既に定量的な管理に活用出来る基盤があり、実際に第3者（StagEスタッフ）でも既存データの定量的分析からプロジェクトの状況を把握出来ることを確認した。また、検証過程で、実プロジェクトへの適用のための幾つかの要件が抽出され、次章のモデリング言語へも反映された。

4. ソフトウェア開発データ分析モデリング言語

4.1　要件と構造

ソフトウェア開発データ分析モデリング言語SDAML※3とは、ソフトウェア開発における定量的データ計測・分析の活動を記述するための記法である［角田2009］。SDAMLの利点は以下の3つである。

・データ分析方法のノウハウの共有が容易になる
・データ分析の目的と方法を明確に示すことが出来るため、ユーザとベンダ間でのデータ共有に関する合意がしやすくなる
・テンプレートの機能を果たすため、全くテンプレートが無い状態に比べ、記述するコストが低い

SDAMLの役割は、直感的にはソフトウェアの仕様記述法であるUML※4の役割に、プログラム設計パターンのカタログであるデザインパターンの役割を加えたものであると考えると分かりやすい。

SDAMLは、データ計測・分析プロセスを記述するために、次の8つの要件を満たすように定義されている。

（R1）　データ分析の目的と、目的を果たすために着目すべきデータ項目を記述出来ること
（R2）　各データ項目の計測方法を記述出来ること
（R3）　各データ項目の分析方法を記述出来ること
（R4）　分析結果に基づいた、実施すべき対応策を記述出来ること
（R5）　データ計測・分析・共有の手順やタイミングを記

※3　SDAML：Software development Data Analysis Modeling Language
※4　UML：Unified Modeling Language

述出来ること

（R6） ユーザ、ベンダそれぞれの分析目的を個別に記述出来ること

（R7） プロジェクト進行中、及びプロジェクト完了時のデータ分析方法を区別して記述出来ること

（R8） データ分析モデルの適用条件を記述出来ること

（R1）～（R5）は、一般的なデータ計測・分析プロセスの活動を記述するための要件である。（R1）はデータ収集計画立案活動（PDCA※5サイクルのPlan）を記述するための要件、（R2）はデータ計測活動（PDCAサイクルのDo）を記述するための要件、（R3）はデータ分析活動（PDCAサイクルのCheck）、（R4）は対応策実施活動（PDCAサイクルのAct）を記述するための要件、（R5）はデータ計測・分析プロセスのフロー（PDCAサイクルのフロー）を記述するための要件である。

（R6）はユーザとベンダでデータを共有する場合に必要となる要件である。ユーザとベンダのデータ分析目的は完全に一致しない場合がある。例えば、「ソフトウェアの出荷後の欠陥数を抑える」はユーザ、ベンダ共通の分析目的となるが、「テスト効率を高める」は、（請負開発の場合）ベンダのみの分析目的としかならず、それぞれを明確に区別して記述する必要がある。

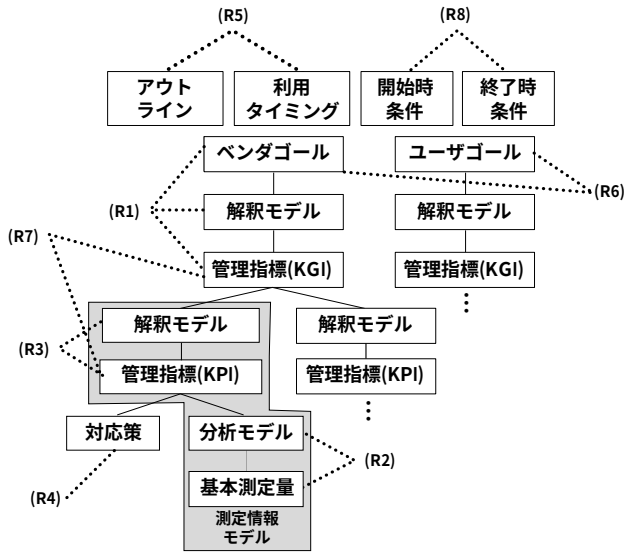

図10　SDAMLの構造と要件との関係

（R7）はユーザとベンダでデータを共有し、かつプロジェクト進行中とプロジェクト完了時の両方でデータ分析を行う場合に必要となる要件である。例えば、「高い品質のソフトウェアを開発する」ことを目的として、「出荷後欠陥密度」に着目して分析しても、プロジェクト進行中に目的が達成出来そうかを判断することが出来ない。逆に、同じ目的で、プロジェクト進行中に「結合テスト欠陥密度」と「システムテスト欠陥密度」に着目して分析することを決めても、最終的に目的が達成されたかを判断する基準を決めていなければ、ユーザとベンダで目的が達成されたかどうかの認識が一致しなくなる可能性がある。

（R8）はデータ分析モデルをカタログ化する際に必要となる要件である。過去のプロジェクトで作成されたデータ分析モデルを、新たなプロジェクトに適用する際、誤った評価をすること避けるためには、「どのようなプロジェクトに適用出来るのか」、「プロジェクト実施中にどういったことが起こると正しい評価が行えないのか」等が明らかになっている必要がある。

SDAMLは、ISO/IEC 15939で定義されている測定情報モデル［ISO/IEC 15939］をベースにした記述法であり、測定情報モデルと同様に、階層構造を持ったモデルである。測定情報モデルとは、データの計測から分析方法までを階層構造により表したモデルである。図10にSDAMLの構造を示す。図では各要素と要件との対応関係、及び測定情報モデルに該当する部分を示している。

次節では、SDAMLの構成要素について説明する（誌面の都合上、主要な構成要素のみ説明する）。

4.2　主要構成要素

（1）アウトライン、利用タイミング

アウトライン、利用タイミングは、「データ計測・分析・共有の手順やタイミング（R5）」を記述するための要素である。アウトラインでは、利用シナリオ（分析モ

※5　PDCA：Plan Do Check Action

デル）の適用場面、ユーザ/ベンダの要求、データ計測、分析手順を具体的に記述する。要求分析におけるユースケースと類似した内容を記述すると考えるとよい。利用タイミングは、データの計測、データのユーザへの受け渡し、データ分析のタイミングを示した図であり、UMLのアクティビティ図を使って表記する。

(2) 基本測定量、分析モデル、管理指標、解釈モデル

基本測定量、分析モデル、管理指標、解釈モデルは「各データ項目の計測方法（R2)」と「各データ項目の分析方法（R3)」を記述するための要素であり、ISO/IEC 15939で定義されている測定情報モデルに基づいている。各要素の役割は以下の通りである。

- 基本測定量：測定対象から直接計測される数値
- 分析モデル：基本測定量に基づいて管理指標を計算する方法を示したモデル
- 管理指標：分析モデルに基本測定量を与えることにより求められる数値
- 解釈モデル：管理指標の分析方法を示したモデル

(3) ユーザゴール、ベンダゴール

ゴールは、「データ分析の目的と、目的を果たすために着目すべきデータ項目（R1)」を記述するための要素であり、GQMパラダイム［BASILI1984］のゴールの概念に基づいている。GQMパラダイムとは、データ収集の目標設定からデータ収集のメトリクス決定までをモデル化したものであり、ゴールとは、計測の目標、計測対象、計測理由等を明確にした文である。ゴールと管理指標を対応付けて記述する。

ゴールにはユーザゴールとベンダゴールの2種類が存在する。これらは、「ユーザ、ベンダそれぞれの分析目的を個別に記述（R6)」するための要素である。

(4) KGI※6、KPI※7

管理指標にはKGIとKPIの2種類が存在し、ゴール、

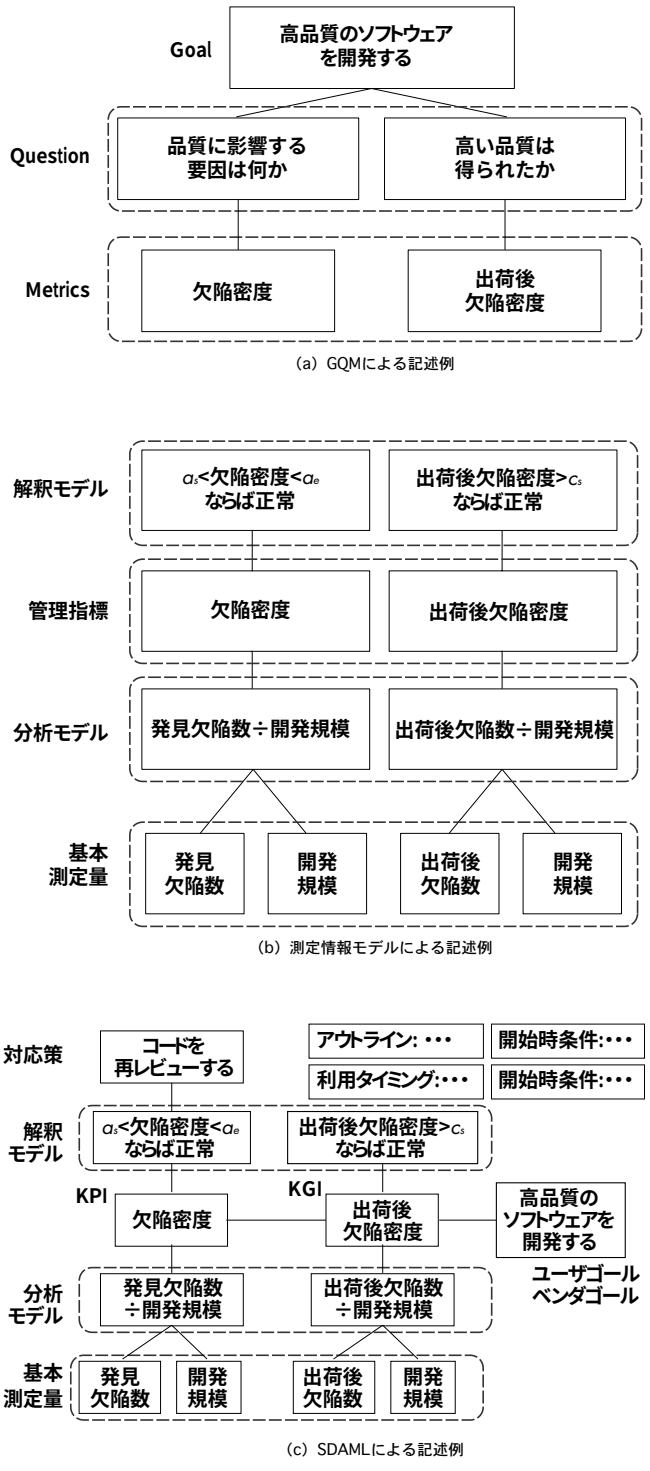

図11　モデル記述例

※6　KGI：Key Goal Indicator，重要目標達成指標
※7　KPI：Key Performance Indicator，重要業績評価指標

KGI、KPIは階層構造となっている。これにより、「プロジェクト進行中、及びプロジェクト完了時のデータ分析方法 （R7）」を記述出来るようになっている。KGIとKPIはビジネスマネジメントの分野で用いられている概念であり、KGIはゴールを達成したか否かを判断するための指標、KPIはプロジェクト進行中にプロセスを評価するための指標である。KPIが目標値をクリアするようにプロジェクトを遂行することにより、KGIも目標値をクリア出来るような関係となる。なお、（「テスト工程の進捗を把握する」等）進捗を把握することが目的のゴール場合は、KGIを設定しない。それ以外では、まずゴールに対して1つのKGIを設定し、KGIに対して1つ以上のKPIを設定する。

4.3 SDAMLと従来法によるモデル記述例

GQM、測定情報モデル、SDAMLそれぞれを用いてモデルを記述した例を図11に示す。GQMは、測定情報モデル、SDAMLに比べ、記述出来る情報量が少ないことが分かる。SDAMLと測定情報モデルで記述出来る情報量の差は大きくはない。しかし、測定情報モデルの場合、ユーザゴール・ベンダゴール、KGI、KPIの関係を表すことが出来ないため、管理指標を何のために利用するのかが非常に分かりにくい。更に、測定情報モデルでは、開始時条件、終了時条件を記述出来ないため、データ分析モデルをカタログ化することが出来ない。

このように、SDAMLを用いてモデルを記述することにより、データ分析の目的と方法を明確に示すことが出来る。また、テンプレートの機能を果たすため、GQMや測定情報モデルを用いるよりも、モデル記述が容易となる。更に、データ分析モデルをカタログ化することが出来る。

5. おわりに

本技術解説では、文部科学省StagEプロジェクトの活動とこれまでに得られた主な成果を3回にわたって紹介してきた。2007年8月から5年計画の始まった同プロジェクトは、ちょうど折り返し地点を迎えたことになる。先日開催したプロジェクト主催の研究会での議論等を拝聴していると、開発データの共有に対するユーザ・ベンダの関心は、プロジェクト開始当初に比べて高まっており、ソフトウェアタグの具体的な利用イメージも明確になりつつある。ソフトウェア開発管理力に自信のあるユーザ・ベンダにとって、開発データの共有によるデメリットは少なく、その一方で、メリットを拡大する余地があると考えられ始めているのかもしれない。

現在のところ、同様の取り組みは海外では見られない。IPA/SECによるプロジェクトベンチマーキングや経済産業省によるソフトウェアメトリックス高度化の取り組み等と連携をとり、そして何より、ユーザ・ベンダ企業との連携の輪を広げることで、日本独自のソフトウェア技術としてソフトウェアタグの研究開発を推進していく予定である。なお、ソフトウェアタグ規格を始めとして、本技術解説で取り上げた取り組みの詳細やその他の研究成果についてはStagEプロジェクトのウェブページ［STAGE Web］にて公開中である。参照いただければ幸甚である。最後に、技術解説の機会を頂いた、SEC journal編集委員会に心から感謝致します。

参考文献

［AMBLER1984］ Ambler, S.W.：Process Patterns：Building Large-Scale Systems Using Object Technology, Cambridge University Press, 1998
［BASILI1984］ Basili, V. and Weiss, D.：A Methodology for Collecting Valid Software Engineering Data, IEEE Trans. On Software Eng., vol.10, No.3, pp.728-738, 1984
［CMMI2007］ CMMI 成果物チーム：開発のためのCMMI®（CMMI-DEV）1.2版 公式日本語翻訳版，技術報告書 CMU/SEI-2006-TR-008，2007
［ISO/IEC 15939］ International Organization for Standardization：ISO/IEC 15939:2002, Software Engineering - Software Measurement Process, International Organization for Standardization, Geneva, Switzerland, 2002
［PMBOK2004］ Project Management Institute：プロジェクトマネジメント知識体系ガイド（第3版）PMBOKガイド，2004
［STAGE Web］ http://www.stage-project.jp/
［角田2009］ 角田雅照，松村知子，松本健一：ソフトウェア開発データ分析モデリング言語の提案，ソフトウェアエンジニアリングシンポジウム2009 併設ワークショップ 「ソフトウェア開発マネジメントのための測定と分析」，2009年9月
［松村2009-1］ 松村知子，大平雅雄，森崎修司，松本健一：オフショア開発におけるユーザ・ベンダ間コミュニケーション情報の分析による仕様伝達の評価，奈良先端科学技術大学院大学情報科学研究科テクニカルレポート，NAIST-IS-TR2009005，2009年10月
［松村2009-2］ 松村知子，松本健一：ユーザとベンダ間の協調による要求品質確保のための定量化事例，奈良先端科学技術大学院大学情報科学研究科テクニカルレポート，NAIST-IS-TR2009006，2009年11月