ソフトウェア開発における情報共有の課題と効果に関する研究

Research on the Problem in Information-sharing and its Efficiency in software development

主査

石田　厚子（株式会社日立製作所）

副主査

高橋　秀敏（東京海上火災保険株式会社）　多田　幸翁（日本電気テレコムシステム株式会社）

研究員

足立　真弘（株式会社ジャステック）　小泉　幸恵（ＴＩＳ株式会社）

石井　英明（株式会社インテック）　沢本　哲夫（株式会社デンソー）

上原　佳広（オリンパスシステムズ株式会社）　十日市　勉（株式会社タイコーシステムエンジニアリング）

内山　英俊（池上通信機株式会社）　中井　一人（株式会社松下ソフトリサーチ）

菊地　淳（ＮＴＴコムウェア株式会社）　松崎　昌史（日本ノーベル株式会社）

木田　昌平（株式会社セゾン情報システムズ）　李　智炯（三星ＳＤＳ株式会社）

研究概要

第７分科会においては、これまでナレッジマネジメントに代表される知識共有について議論を行なってきた。その結果、知識共有は多くの場合、企業活動においてプラスの成果をもたらしていることがわかった。これはソフトウェア開発においてもあてはまる。

しかしながら、ソフトウェア開発では、知識に限らず開発工程の各段階において、開発者間の情報共有が十分に行なわれているとはいいがたい。なぜであろうか？

ソフトウェアは目に見えないものであるが故に、品質管理が難しいとされる。同様に情報共有を行なったとしても、どの程度効果があるかは目にみることはできない。われわれはここに解を求めた。すなわち、目に見える形で効果を表すことができれば情報共有も可能になるとの仮定を設け、開発時の情報共有を進めた場合にどの程度生産性が向上するかの検証を行なった。また、一般に共有化が難しいとされる個人・組織が有しているノウハウの共有がもたらすソフトウェア開発の効率化について考察した。

Abstract

The 7th committee has discussed the information sharing, the typical knowledge management in the past.　As a result, we found that it brought us plus achievement in many business cases. And it also did in Software development.

Software is considered to be the thing out of control in quality as it is invisible. As such, it is hard to visualize how effective it is even if it really shared information. We executed the experimentation to verify how much it improved its productivity when we shared the information during the development process. It was done under the condition that we can visualize the information sharing process if we can shift something invisible to visible.

Moreover, we studied the efficiency in software development coming from sharing the know-how of the individual or the organization which we believe it is hard to share.

１．研究課題の選定理由と背景

近年、情報技術の進歩やシステムの複雑化・多様化に加え、情報システム構築のスピードアップも要求され、ソフトウェアの肥大化、テスト項目の増大、短期間でのシステム開発などのため、ソフトウェア開発自体も容易ではなくなってきている。

このような状況の中で、高品質なソフトウェアを迅速に開発するためには、ソフトウェア開発上の情報共有や再利用なくして現代のソフトウェア開発は行えなくなりつつある。

当分科会メンバの各社でも、それぞれ情報共有や再利用に関する試みを行い、一部、運用が継続しているものがあるが、成功とまで言えない状況である。

これらの状況を踏まえ、当分科会では各社の情報共有や再利用に関する課題をもとに、「高品質ソフトウェア開発のための知識共有と再利用」をテーマに掲げ、研究活動を開始することとした。実際の議論の中では、特に「ソフトウェア開発時の情報共有」に焦点を当て、活動を進めていくこととし、また併せて「ソフトウェア開発時のノウハウ共有」に関しても議論を進めてみることとした。

２．本年度活動の目標

当分科会では、前述の研究課題の選定理由と背景を踏まえ、次の内容を中心とした研究を行うことを活動目標とした。

「ソフトウェア開発時の情報共有」

「ソフトウェア開発時のノウハウ共有」

３．活動内容

前述の様に、実際のソフトウェア開発の現場では情報共有や再利用の必要性を感じているが、満足のいく効果を得られていない状況である。

各社とも、「良い情報が集まらない」「継続されない」などの状況であり、活動開始当初、各社の情報共有について意見を出し合い、課題の整理を行った。その結果、情報共有を考える上では、大きく２つの課題に類別されると考えた。第一は、情報の収集、加工・分類、公開などの「情報共有の仕組みに関する課題」、第二に、インセンティブ、人および組織などの「情報を扱う人に関する課題」である。これらの課題を踏まえ、情報共有を成功させるための要因について検討を行った。

また、情報共有に関し、大成功には至っていないが、一部、継続して運用されている例もあり、各社の実際の情報共有事例をまとめてみた。共有する情報の対象として、トラブル情報、顧客情報、個人が持っている技術情報・ノウハウなど、様々な情報が存在するが、共有効果のある情報、共有が比較的容易な情報の分析を行った結果、当分科会では、「開発工程に関わる情報」について、掘り下げて議論を進めることとした。その中で、情報共有が進まない理由のひとつと考えられる「情報共有効果の把握が難しい」という点を打破するために、これらの情報を共有した場合の効果についても言及することとした。さらに、知識マスターやナレッジマネージャの必要性についても話し合った。

活動の後半では、共有効果はあると思われるが、実現が最も難しい「ノウハウ」の共有について検討を行い、今後の情報共有の取り組みへの方向性を示すこととした。

４．研究成果と考察

４．１　情報共有の課題

情報共有を成功させるためには、如何なる仕組みを構築すれば良いのか。如何なる仕組みならば、メンバーが容易に利用できるのか、利用しようと考えるのか。

以上のような情報共有を実現する際の課題について考察する。

先ず、情報共有を成功させている三星ＳＤＳ株式会社の「ARISAM」というシステムについて分析した。

その結果、当初成功させるためのキーワードは、「トップダウン」、「強制」、「インセンティブ」であると認識した。

しかし、考察を進めたところ、「ARISAM」の成功は韓国の状況と関連が深いと思われ、日本の状況においては必ずしもうまくいくとは限らないと判断した。そこで、日本での情報共有事例について考察することにした。

４．１．１　課題の提起

まず、各社で実施されている情報共有事例を出し合い、これらについて分析し共有がなぜ行いにくいのかをブレーンストーミング的に議論し、情報共有の課題を提起した。

その結果、前章でも記述したように、課題は大別して２つのカテゴリーに分類できた。

（１）仕組みに関する課題

（１－１）情報の収集および登録

情報共有を行うためには、先ず情報を収集し登録する必要がある。情報の収集および登録に関する課題は以下のようになった。

・無条件に収集しても質が悪く、価値がないものばかりが多くなる。
・ボランティアのみに頼っていると、有益な情報は集まりにくい。
・所有している情報を非公開とすることで自身の存在価値が高まると考える人がいる。
・Give&Take になっていない。通常は提供する人は提供専門で利用する人は利用専門になっていることが多い。結局、一部の人を頼りにシステムが成り立っている。

（１－２）情報の加工および分類

情報が登録されただけでは利用しにくいので利用できるように加工することが必要である。

情報加工に関する課題は、以下のようになった。

・通常の業務を進める際に発生する情報は難なく収集できるが、そのままでは利用しにくい。
・加工を行うには、内容を理解していなければできない。そのためにはＡＲＩＳＡＭにおける知識マスターおよびナレッジマネージャ*1 に当る人が必要であるが、日本では業務として切り出すのが難しい。
・必要な情報を探索する手段として全文検索だけでは利用しにくい。分類が必要であるが、それにコストがかかるとメンテナンスされにくくなり、結局使われなくなる。
・情報を登録する人が加工・分類まで行うことが考えられるが、忙しい時には登録がされなくなってしまう。

（１－３）情報の公開

情報が登録、加工されても利用されやすいように公開する必要がある。利用環境も含めた公開に関する課題は以下に示す状況である。

・イントラネット環境でＷｅｂ参照できる仕組みは必要であるが、Ｗｅｂサイト構築ツールを利用すればあまり問題ではない。
・モバイルでＷｅｂが利用できる様にするためには、インターネット接続またはリモートアクセス環境が必要である。

（２）人に関する課題

（２－１）インセンティブ

「登録する人」、「作る人」、「使う人」に与えるインセンティブに関する課題は以下のようになった。

・日本の風土ではインセンティブを与えても効果が継続しない面がある。
・インセンティブと言うよりは、ペナルティーを課して強制する方向になりがちである。
・技術提供や利用に報いるためにはインセンティブが必要だと思われるが、インセンティブが存在しても失敗したケースも報告されている。

（２－２）人および組織

これまでの全ての課題に共通して言及できることは結局「人」である。情報を扱う「人」に関する考察を行わないで情報共有の仕組みを作ることはできない。

・重要な要因は「人」である。「人」を考えて仕組み作りをする必要がある。

４．１．２　成功させるための要素の考察

提起した課題に基づき、成功させるための要素について考察した。

考察の過程で課題提起されなかった要素が現われ、それを新たに追加し分類を見直した。

（１）仕組みに関する要素

（１－１）情報の収集、登録および活用

・収集した情報で何を行いたいのかが重要である。
・最初は参照するが、徐々に参照しなくなるのは、情報が更新されないからである。極力登録を行い、有効な情報があることを認識させる必要がある。
・如何なるものでも登録した方が良い。少々質が落ちても構わない。
・量より質という場合もある。質の良いものもそれなりの割合で含まれるようにする。
・部分的に質の良いものを半強制的に登録することも必要である。
・多くの登録を行い自然淘汰することにより、良いものを残していけばよい。
・情報を登録することが業務プロセスに組み込まれていることが理想である。組み込まれていなければ、強制的に登録する必要がある。
・以前に何処かに記述したことを改めて登録しようとはしない。プロジェクト終了時の報告書など、有益な情報が記載されたものは相当数存在するので、自動的に登録できるとよい。

情報の収集および登録に関する議論は未だ入り口に過ぎない。結局、電子的に何処か

に記述した情報をどのように活用するかが重要である。

（１－２）人数および規模

・WWWが成り立っている仕組みを考えると、圧倒的に参照するだけの人が多いが、有用な情報も多いので、役立っている。

・大きくするなら徹底的に大きくした方が良い。

・大きければ大きいなりに、小さければ小さいなりの利点があるに違いない。

当初は、小さい組織でないと成功は難しいと考えていたのであるが、人数及び規模そのものは無関係であるという考えが有力になった。

（２）人に関する要素

（２－１）インセンティブ

・最初の立ち上げ時にはインセンティブが必要かもしれないが、長期間継続させるための要素としては疑問である。

・WWWが成り立っている仕組みを考えると、インセンティブに関係なく情報を発信する人が存在する。そのような人が存在しなければ育成する必要がある。

当初は、インセンティブは必要不可欠な要素と考えていたのであるが、無関係であるという考えが有力になった。

（２－２）動機付け、人および組織

・情報共有に対してアグレッシブな人ばかりとは限らない。いわゆる「２－２－６の法則」に於けるアグレッシブな上位２０％の人だけでは厳しい。中間の６０％の人を極力アグレッシブに動機付けさせるような仕組みが必要である。

・上に立つ人が、そのようなシステムが重要だと認識することが大切である。

・動機付けに関することは、結局は「人」の問題である。

「人」の問題は、決して避けて通ることができない難しい問題である。

（２－３）効果

・効果がある程度目に見えないと説得力が無い。

・効果を定量的に評価することが難しい。

・効果が目に見えれば、メンバーは参加するものである。

・長期間継続させるためには目に見える効果が必要である。

結局、情報共有による効果をどのように把握するかが最も重要な問題である。

考察の結果、動機付け、つまりは人および組織の問題が根底に存在し、情報共有の効果を定量的に評価することが難しいため動機付けを行うのが難しく、共有が進まないのであると結論付けた。

次節では、共有による効果の定量的評価に関して検討する。

４．２　開発工程と成果物の共有とその効果

４．２．１　開発工程と成果物の共有

共有化による生産性向上の効果をもたらすものは、工程内で生成される成果物の他に、開発工程の周辺にある成果物も該当しうる。前者は設計書・ソフトモジュール等であり、後者は顧客情報や工数情報等である。

開発工程内で生成される成果物を共有化するためには、工程を定め、成果物を明確にする必要がある。数社を調査したところ、ソフトウェア開発プロセスモデルはウォーターフォール型を採用しており、開発工程は会社によってまちまちであったが、それぞれ定められた工程にしたがって開発していた。

開発プロジェクトや開発チームを、業務別で構成するか技術領域別で構成するかによって共有したい情報は異なっている。前者は、顧客や業務に関する情報共有に適している反面、他プロジェクトにおける同等の機能の共有が困難である面を持つ。業務別構成のチームではチーム間の連携を保つことが難しく、類似機能を有するプロジェクトが存在した場合であっても、共有を行う為には予め同様な機能分割を実施しておく必要があり、実践を妨げている。後者は、技術ドメインごとにチーム編成されることから、個々の技術分野（例えば、無線通信技術，ハードウェアドライバ組み込み技術等）における技術共有を推進することができるという特徴を持つ。

表１において、業務系・技術系それぞれの成果物例をあげている。

表１　開発工程別成果物一覧

工程	作成されるべき成果物	実ドキュメント名	
		業務系	技術系
基本設計	システム化計画書	開発計画書	仮スケジュール計画（要員確保のみ）
	開発計画書		
	要求定義書		
	基本設計書	システム分析書	要望確認メモ
	システムテスト計画書	ST計画書	計画書
	基本設計レビュー議事録	SAレビュー議事録	見積書、契約書
機能設計	外部設計書	UI設計書	外部仕様書
	内部設計書		内部設計概要メモ
	結合テスト計画書	IT計画書	
	機能設計レビュー議事録	UIレビュー議事録	レビュー議事録
	–	UI仕様変更連絡票	レビューコメント票
構造設計	プログラム設計書	SS設計書	詳細設計概要メモ
	構造設計レビュー議事録	SSレビュー議事録	レビュー議事録
プログラミング	ソースプログラムリスト		
	モジュール		
単体テスト	単体テスト項目チェック表		単体テスト仕様書
	単体テスト報告書		単体テスト成績書
結合テスト	結合テスト項目チェック表	IT項目チェック表	結合テスト仕様書
	結合テスト報告書	IT報告書	結合テスト成績書
		IT障害記述票	
		IT障害管理票	
システムテスト	システムテスト項目チェック表	ST項目チェック表	システムテスト仕様書
	システムテスト報告書	ST報告書	システムテスト成績書
		ST障害記述票	
		ST障害管理票	
保守	–	–	–

業務系の例（付表１）では、パッケージソフト開発の例で検証した。

パッケージソフトの場合、不特定の顧客が使用するため汎用性を高める必要がある。また、個別のユーザニーズに加え市場動向や他社動向といった要件も考慮しなければならない。したがって、これらを明確にしている上流工程の成果物ほど共有度合が高くなっている。但し、共有度合が高いのは開発プロジェクト内の場合で、開発プロジェクト間では必

ずしも高くはない。これは、開発するパッケージソフトの種類により製品の設計思想が異なるため、上流工程の成果物の再利用が難しいことが考えられる。

技術系の例（付表２）では、ＷＥＢアプリケーション開発と組み込みソフトウェア開発の例で検証した。技術系の例の場合、全体的な共有度は高くなっている。これは設計→プログラミング→テストを繰り返すスパイラル方式を一部取り入れているため、結果として共有化が進んでいると考えられる。しかしながら、見積書・契約書といった設計工程以前の段階についての共有化は進んでいない。組織が同じ技術を持つ集団として構成されているため、開発するソフトウェアが利用されている業務分野が異なることが多くなることに起因していると考えられる。

４．２．２　共有化の効果

工程内の成果物の共有効果は、数％から８０％まで幅広い。これは、共有される成果物をどれだけ標準化し、再利用率を上げることができるかによって変わる。

業務系の例の場合、共有化が進んでおらず、かつ全体工数に占める割合の多いプログラム工程以下の工程で共有化を進めることで、ソフトウェア開発の生産性向上が期待できる。モジュールの部品はもとより、テスト工程においてもテスト項目やテスト環境を再利用することにより各工程で必要とされる工数の削減が可能になる。また、他のパッケージソフトで発生した障害をテスト項目に取り入れることで、品質向上にも寄与することになる。したがって、これらを共有化するには共有化されるに足るレベルにあることが要求される。開発プロジェクト単位での作成ではなく、常に再利用を意識した成果物の作成が求められる。また、ブラッシュアップも欠かすことなく進めていく必要がある。このサイクルを繰り返していくことで品質・生産性向上を図ることができる。

技術系の例の場合、ＷＥＢアプリケーション開発と組み込みソフトウェア開発とで工数削減可能な工程がやや異なっている。ＷＥＢアプリケーションの場合、機能設計工程が最大の削減時間を持つ工程となっている。新しいプラットフォーム・言語での開発の場合、既存の設計文書の再利用は難しい。従来の設計技法や設計手順と異なることが多いからである。この点、経験を積むことで技術・ノウハウが蓄積され、同様の開発プロジェクトへのフィードバックにより工数削減が可能になる。その効果が一番高いのが機能設計文書であるということができる。一方、組み込みソフトウェア開発の場合、単体テスト工程の削減時間が一番大きいものとなっている。結合テスト工程以降がないので、実際にはテスト工程全般ということができる。これは同種のハードウェアへの組込用ソフトウェアであれば、テスト工程の再利用割合が大きいことを示している。

業務系・技術系共通でいえることは、工程単位で成果物共有効果が異なることである。開発プロジェクト単位でも異なることが考えられるので、効果の大きいものから共有化を進めていくことが望ましい。

４．３　ソフトウェア開発時のノウハウ共有

４．１及び４．２節ではソフトウェア開発時の成果物に焦点を当てて共有化がもたらす効果について考察を行った。

これらの議論を重ねる中でたびたび話題にあがった事項として成果物として固定化されていない、いわゆるノウハウの共有がある。

本章では、このノウハウの共有についての断片的な議論を整理し、ノウハウの共有を実際に行う時のヒントとしたい。

ノウハウを共有したいという要望は議論の中ではかなり多くあった。特に、設計やプログラミングの方法、円滑なプロジェクト運営に関する技術的なノウハウなどである。

これらのノウハウが何故共有されにくいのかを考えると以下の原因が考えられる。

・ノウハウは成果物などの中に直接記述されることがない。

・ノウハウ共有の仕組み（加工、蓄積、参照）を構築、運営しにくい。

・ノウハウを提供する側のメリットが明確でない。

当分科会メンバが所属する会社においての状況は、ARISAM を運用している三星ＳＤＳ株式会社以外では、ノウハウの加工や蓄積を行うための時間は公的に確保されていない。すなわち、ノウハウを共用するためにはボランティアとしての活動が必要となり、したがって持続性が無い。また、ノウハウの共有・蓄積・参照の仕組みを構築するためには、その効果を事前に定量化できないという難問が立ち塞がっている。

しかし、各社ともノウハウの共有に関する試みが行われたり、現在行いたいと考えている事実も明らかになった。つまり、ノウハウが本当に共有できるならば、新たな効果が生まれてくると考えているのである。

共有したいノウハウとしては前述したように設計のノウハウやプロジェクト管理のノウハウが最も望まれているが、これらをソフトウェア開発という面から見ると開発プロセス関連（進め方）と設計アイデア関連（作り方）という２種類に大別できると思われた。

以下では、当分科会の議論の中で紹介された内容を中心に整理を行うことにする。

４．３．１　開発プロセスに関するノウハウ共有

最近はソフトウェア開発プロジェクトを可視化するために開発プロセスとプロセスモデルを考えていく傾向にある。これは、ソフトウェアが見えにくいものであるため、ソフトウェアを作り出すプロセスに着目し、良いプロセスに基づいて開発されたソフトウェアは品質も生産性も高くなり、開発プロセスを改善することで品質や生産性をさらに向上させるという考え方である。

ノウハウの共有という面から開発プロセスを考えると、個人の資質や能力あるいは経験などに依存する個人差が大きく、優れた開発の進め方や考え方を共有できていない状況にある。

開発プロセスのノウハウ共有としては、①開発プロセス自体の共有と②共有が行いやすい開発プロセスということが浮かび上がってきた。

①の代表的な考えはＣＭＭなどに代表されるプロセスフレームワークの共有であり、②の流れとしては最近話題になっているエクストリーム・プログラミング（extreme Programming：ＸＰ）に代表される開発者視点で軽量級のプロセスモデルである。①についてはＣＭＭなどの議論に譲り、②についてノウハウの共有の面からＸＰを例に考える。

ＸＰとは４つ価値と１２（あるいは１４）のプラクティスから構成される開発方法論（あるいは開発哲学）である。その開発スタイルは、２人組になった開発者が日々分析、設計、実装、テストを繰り返し、極限にまで反復単位を短くして、フィードバックを得ながら開発を進めるものである。

４つの価値のうちのコミュニケーションとシンプル、１２のプラクティスのうちの計画ゲーム、メタファー、ペアプログラミング、共同所有、オンサイトのユーザ、オープンワークスペースがノウハウの共有に関係する。

ＸＰは開発者間やユーザと開発者間のコミュニケーションを重視しており、そのために全てをシンプルにすることをうたっている。コミュニケーションすることにより開発情報だけでなくノウハウも共有される。

また、コミュニケーションを円滑にするためにペアプログラミングなどのプラクティスが存在し、ノウハウの共有が円滑に行える工夫がなされている。

当分科会のあるメンバの職場では開発者が全てオープンスペースにおり、その結果として聞きたいことがすぐ聞けるという発言がなされた。

ＸＰでは、前述した３つの阻害要件のうち、「仕組みづくり」と「ノウハウ提供のメリット」の対応が行える。

４．３．２　設計アイデアに関するノウハウ共有

４．３．１節では、開発プロセスについて記述したが、ソフトウェア開発の分析・設計・アーキテクチャなどの決定の際に他のシステムで使われたアイデアを利用したいという要望も議論した。

しかし、アイデアそのものは成果物に記述されることは少なく、アイデアを適用した結果のみが成果物に記述されるため、類似システムである場合を除くとアイデアの再利用はなされず共有がされにくい。

前述のＸＰでは、同一プロジェクトやチーム内の設計アイデアも共有されるが、プロジェクトやチーム間での共有は行われない。

最近、ソフトウェア開発の技法として「パターン」という考え方がでてきており、分析・設計・アーキテクチャに関するアイデア共有の可能性が出てきた。

パターンは、過去に有用さが実証されている分析時や設計時に現れる解（アイデア）を、「問題」「文脈」「解決」の組にして提示し、名前を付けたものであり、一種の「設計カタログ」として利用できる。

たとえば、分析においては概念モデル構築時には、「アナリシスパターン」という分析レベルのパターンがまとめられている。

また、設計モデルの構築時には、「デザインパターン」というソフトウェア構造レベルの設計パターンが存在し、アーキテクチャ決定の際には「アーキテクチャパターン」も存在する。

パターンは、前述したように「問題」「文脈」「解決」という形式でアイデアを整理し、それを利用する時のトレードオフや他のパターンとの関係なども記述されている。

パターンを知ることで過去何度となく繰り返された分析・設計時の発明・発見（設計アイデア）のエッセンスを知識として得ることを可能にしている。

さらに、独自のパターンを見つけだし、組織に合ったパターンのカタログ（これをパターンランゲージという）を作成することにより、より効果的なアイデアの共有が可能になる。

パターンを利用することにより、前述した３つの阻害要件のうち、「ノウハウの成果物

への記述」と「仕組みづくり」の対応が行える。

５．まとめと今後の課題

今回われわれは、ソフトウェア開発における情報共有のもたらす効果について研究を行なった。情報共有によって、当初は開発工程全般に対してある程度平均的な効果があるものと考えていたが、調査分析の結果開発工程ごとに効果に違いがあることが判明した。また、その結果の度合いは、会社組織の業務構成にも依存することを導くことができた。このことは、無目的に情報共有を進めても生産性向上どころか、逆に情報共有のための余分な工数を要する可能性さえある。情報共有にあたっては、どの情報を共有化すれば効果があがるかを見極めて進めるべきであろう。

今回は仮定上の話に留まっており、サンプル数も少ないものであった。したがって、実際の情報共有の効果としては、開発プロジェクトや組織のあり方によりわれわれが考察した結果とは大きく異なることが考えられる。今後、情報共有を進めるにあたり、共有前と共有後でどの程度生産性が向上したか、客観的な数値をもって検証を行なっていきたい。また、目に見える成果物とは別の、ノウハウの共有に伴う生産性向上についての具体的な数値の検証も行なっていきたい。

６．参考文献

１）ケント・ベック　エクスストリーム・プログラミング入門
（株）ピアソン・エデュケーション　２０００年
２）日本ＸＰユーザグループ　extreme Programming テスト技法
（株）翔泳社　２００１年
３）鈴木純一他　ソフトウェアパターン再考　（株）日科技連出版社　２０００年
４）中谷多哉子他編　ソフトウェアパターン　共立出版（株）　２００１年

*1 いずれも、ARISAM で使用されている用語。ARISAM の詳細は第 16 年度ソフトウェア品質管理研究会第 7 分科会報告書を参照。

知識マスター：システム（ＡＲＩＳＡＭ）に登録された知識の価格（サイバー貨幣）を決定する。登録された知識の中で優秀な知識については利用者に推奨する。

ナレッジマネージャ：ＡＲＩＳＡＭにおいて利用者の利用現況や 知識マスターの活動現況を管理しながらコンテンツ別に発展の方向を提示する。