

# ソフトウェア開発における見える化活動

Make-Visible Activity in Software Development

藤田国和 \*  
Kunikazu Fujita

稲葉 徹 \*  
Toru Inaba

山口正毅 \*\*  
Masaki Yamaguchi

池上雅雄 \*\*  
Masao Ikegami

\* ソフト・アライアンスグループ ソフトウェアプロダクト事業部 第一開発部

\*\* Active-V 推進室

ソフトウェア開発は本来創造的で楽しいものである。しかし開発現場からは楽しくなくなってきたという声が聞こえてくることもあった。PFU のあるソフトウェア開発部門では、メンバーがやりがいを持って、自発的に考え、行動する、本来の楽しいソフトウェア開発を目指して「見える化」活動に取り組んだ。その成果をまとめるとともに、今後の PFU の取組みを紹介する。

Software development itself is a creative and enjoyable activity. However, it is becoming less of an enjoyment-that is what is being heard sometimes in the software development scene. Some software development sections in PFU started "Make-Visible" activity-which is enjoyable by nature-for realizing the development of software in which each member of the section is highly motivated to think and take action on their own initiative. This paper summarizes the result of such activity, and also introduces PFU's approach to this matter in the future.

## 1 まえがき

ソフトウェア開発は、より短納期で、より高品質で、より低コストでの開発が求められるようになってきている。これらの要求により開発現場へのプレッシャーもますます大きくなってきており、以下のような様々な問題が発生している。

- 1) 残業が多い
- 2) 予想外の作業が発生し、進捗が遅れる
- 3) 特定の人に負荷が集中する
- 4) 頻繁にスケジュールを見直さなくてはならない
- 5) 作業の優先度が頻繁に変わる
- 6) 新しい技術を身に付ける時間がとれない

これらは発生する問題の一例であるが、このような問題は、特定のプロジェクトだけで発生するのではなく、ほとんどのプロジェクトで発生している。これら問題を解決するために、これまで能力成熟度モデル (CMM : Capability Maturity Model の略) などを導入し、プロセス改善活動に取り組んできたが、このような問題はなかなかなくならなかった。また、これら問題は現場担当者を疲弊させ、モチベーションの低下を

招く要因となっており、本来創造的で楽しいはずのソフトウェア開発が、楽しくなくなってきたという声が聞かれることもあった。

そこで筆者らはこの状況を打破すべく、改善活動に取り組んだ。本稿ではその改善活動の柱となった「見える化」について紹介する。

## 2 現状分析と改善策

### 2.1 ソフトウェア開発をとりまく現状

#### (1) ソフトウェア開発の中心は人

ソフトウェア開発は知識労働であり、人に大きく依存する。メンバー全員が知恵、工夫、創造力を発揮しないと良いものではない。そして良いものを創るためには、メンバーそれぞれがモチベーションを高くもって、やりがいを感じる事が非常に重要である。

しかし、人が中心で作業を進めるため、問題が発生すると、これを人と切り離して考えることが難しい。そのため、問題の原因を突き止めようとすると、どうしても「あいつが悪い」という風に、原因が人にあるように

感じられてしまい、何かあればだれかが責められる。こういった人を責め合うところからは何も生まれないし、メンバーのモチベーションを高く保つということも難しい。実際、筆者らの職場でも人を責め合う状況があり、解決したいポイントの一つでもあった。

## (2) ソフトウェア開発は不確実性との戦い

ソフトウェア開発の作業は、工場の製造現場で見られるような繰返し型作業とは異なり、全く同じことは二度と繰り返すことがない非繰返し型の作業である。もし同じ作業が繰返しできると仮定した場合、その回数と作業時間の分布は図-1のようになると言われている。

例えば、部品をネジで固定するような繰返し型の作業の場合、平均1分でできるとすれば、グラフの頂点は1分となり、ばらつきはせいぜい±数秒の範囲となるだろう。一方、見知らぬ土地へ車で移動するような非繰返し型の作業の場合、仮に何度も経験できるとすれば、ある時間に到着する回数がもっとも多くなり、そこがグラフの頂点となる。数回に1回はそれよりも早く到着することもあるだろう。しかし、「渋滞に巻き込まれる」、「道に迷う」、「工事による通行止めで迂回が必要」、確率は小さいだろうが「事故に巻き込まれる」などの可能性があり、こういった事象が発生すれば、移動時間は延び、どれだけ時間をかければ確実に到着するということも言えない。これは、非繰返し型作業の場合、不確実性が非常に大きいためである。

ソフトウェア開発の場合、不確実性の要因としては、

- 1) 時間の流れとともに発生する要求の変化
- 2) 認識違いや考慮不足による後工程での手戻り
- 3) 過去の製品の保守作業などによる割り込み

など様々であるが、ある作業で不確実性による問題が発生すれば、その作業の期間や工数はすぐに2倍、3倍と膨れ上がる。

こういった現実の中で、我々ソフトウェア開発者やプロジェクトマネージャーは、納期遵守や品質確保を行

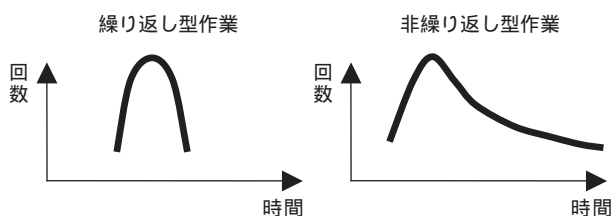


図 1 作業における時間と回数の関係  
(Fig.1-Relationship between time and number of times regarding development work)

わなくてはならない。つまり、ソフトウェア開発をマネジメントすることは不確実性との戦いといっても過言ではない。

## (3) ソフトウェアは見えない

ソフトウェア開発というのは、そもそも見えないものを作るという作業である。家のような建築物や道路など、目で見えるものは、どの程度出来上がっているか、実際に見ることができるので分かりやすい。一方、ソフトウェアは目に見えないので、どの程度出来上がっているかというのは分かりづらい。

また、ソフトウェア自身が目に見えないものなので、どんなものを作っているのか、関係者間での認識違いといったことも発生しやすく、後になって「思っていたものと違う」、「これでは使えない」といった手戻りも発生しやすい。

## 2.2 従来のプロジェクトの進め方

このような現状の中、筆者らの職場では、「プロジェクトの納期を守るためには、プロジェクトの様々な作業一つ一つに期限を設定し、その期限を守らなければならない」という考え方でプロジェクトを進めてきた。この考えは筆者らの職場では従来から当然のように受け入れられてきた。しかし、筆者らが検討を重ねた結果、この考えが元で様々な問題が発生するという結論に達した。以下にこの考え方の問題点を示す。

- 1) ソフトウェア開発には不確実性がつきものであり、一つ一つの作業の期限を守るためには、作業時間に安全余裕を含めなくてはならない。しかし、プロジェクトとしては短期間での開発が求められており、各作業における安全余裕は、不確実性による問題に対して十分に確保することはできない。このため不確実性による問題が発生すれば期限に間に合わない作業が出てくる。
- 2) 遅れが発生すれば、担当者は残業や休日出勤を行い、遅れを回復しようとする。
- 3) 担当でリカバリーできない場合は、他の人による応援という行動がとられる。しかし、ほとんどの場合、応援する人も作業を抱えているため、作業の掛け持ち状態となる。掛け持ち作業は遅れを生む要因であり、遅れが発生する可能性が高まる。
- 4) 2), 3) 項のような行動は後続の作業にも影響が及ぶため、プロジェクトが進むに従って、遅れが発生する作業が増える。また、現場担当者の負荷も大きくなっていく。

5) 担当者の努力だけでは、遅れの回復が難しい状況になると、作業の優先順位の見直しやスケジュールの見直しなどが行われる。頻繁な見直しは現場の混乱を招く要因となる。

6) 作業一つ一つの期限を守るという考え方でプロジェクトを進めると、プロジェクトマネージャーは作業一つ一つについて期限が守られているか厳しく監視することになる。これにより、現場担当者へのプレッシャーは必要以上に大きくなる。また不確実性による問題が発生し、作業に遅れがあれば、「なぜ事前に想定できなかったのか」、「なぜ回避できなかったのか」、「もっとうまく対処できたのではないか」と担当者を責める場面が見られる。

7) 人は時間が与えられると、与えられた分を使い切ってしまう生き物である。いわゆるパーキンソンの法則である。もし、不確実性による問題が発生しなかった場合、作業は期限前に完了するはずである。しかし、この人間の特性により、期限通りとなるように作業を行ってしまう。つまり作業期間に含まれる安全余裕はその作業で使い切ってしまう、プロジェクト全体を守るという形にはならない。

このように「プロジェクトの納期を守るためには、プロジェクトの様々な作業一つ一つに期限を設定し、その期限を守らなければならない」という考え方では様々な問題を引き起こす。改善活動を進めるには、この部分最適の考え方から全体最適の考え方へのパラダイムシフトが必要である。

### 2.3 改善策

まず、今回の改善活動の目的を以下に示す。

- 1) メンバーが自発的に考え、行動する風土、改善し続ける風土作り
- 2) ソフトウェア開発における不確実性のマネジメント

これら目的を達成するためには、プロジェクトの状況や各自の状況が見えるということが、とても重要である。見えることで、状況がリアルタイムに把握でき、メンバーが自ら問題に気づき、自分たちで解決策を考え行動できる。また状況が見えることで不確実性による問題が発生した場合、その影響が把握でき、適切な判断ができる。

そこで、筆者らはソフトウェア開発における「見える化」に取り組んだ。

## 3 取り組み事例

導入した代表的な施策について説明する。

### 3.1 プロジェクト状況の見える化

プロジェクトの状況を「見える化」するために、クリティカルチェーンプロジェクトマネジメント（以降、CCPM）を採用した。CCPMはTOC（制約条件の理論）を元に考えられたプロジェクトマネジメントの手法である。詳細は参考文献 参1、参2を参照されたい。

#### (1) CCPM

CCPMは個々の作業の期限を守ることも、プロジェクト全体の期限を守ることを重要視したプロジェクトマネジメントの手法である。

CCPMでは各作業の安全余裕を取り除き、プロジェクトの納期を守る上で必要な箇所に集め、バッファという形で配置する。このバッファを監視することで、プロジェクトの状況や、不確実性による問題発生時の影響が把握できるようになる。図-2にCCPMによるスケジュールのイメージを従来手法によるスケジュールのイメージと対比して示す。

#### (2) CCPMにおける見積り

CCPMでは、各作業に含まれる安全余裕を除いて見積もる。これは「何のトラブル・割り込みもなく、その作業だけに集中できて順調に進んだ場合、最短でどれくらいの期間がかかるか」という見積りである。筆者らはこれを「50%見積り」と呼んでいる。従来の「作業の期限を守る」という考え方では「見積り=約束」なので、安全余裕を取り除くことができない。ここで「作業の期限を守る」から「何かあれば作業が遅れてもかまわない」という大きなパラダイムシフトが必要なのである。

しかし、従来からの考え方を変えることは難しく、抵抗も大きい。筆者らは現状を変える必要があること、従来の考えでは様々な問題が発生することを粘り強くメンバーに説明し、理解を促すよう努めた。

#### (3) CCPMにおける進捗管理

CCPMの進捗管理では毎日担当者に「今やっている作業があと何日かかるか」をヒアリングし、その結果からバッファの消費状況を把握する。筆者らが採用したプロジェクト管理用のソフトウェア、(株)ビーイングの「BeingProject-CCPM」ではバッファの消費状況を図-3のような形で色分けして出力することが可能である。そして色の説明のように対策検討、実施を行うのである。

このようにバッファの消費状況を「見える化」することで、だれでもプロジェクトの状況が一目で把握できるようにした。

### 3.2 各自の作業状況の見える化

各自の作業の状況を「見える化」するために、ソフトウェアかんばん<sup>参3)</sup>というツールを導入した。

ソフトウェアかんばんは、図 - 4 のように Todo, Doing, Done に分割されたフォーマットを使い、「やるべきこと」、「やっていること」、「完了したこと」を付箋紙に書いて壁に貼り出し、各自の作業を見えるようにするツールである。これによりだれがどのくらいの作業を持っていて、それらがどのような状況か把握可能となる。

### 3.3 ふりかえり会<sup>参3)</sup>

新たな改善案の創出、有効な改善案の定着や我々に

あった方法へのカスタマイズを狙い、1 ~ 2 週に 1 回、定期的なふりかえり会を実施することにした。定期的なふりかえり会を行うことで、以下のような効果も期待できる。

- 1) チャレンジすることは自分たちで決めるので、納得して取り組むことができ、確実に実行される。
- 2) フィードバックが速く行われるので、失敗を恐れずに「とにかくやってみよう」という意識になりやすい。

3) 目的の共有やチームビルディングにも効果あり。

ふりかえり会とは KPT と呼ばれる図 - 5 のように Keep, Problem, Try に分割したフォーマットを使い、「続けたいこと」、「問題」、「チャレンジすること」について議論する場である。KPT はとても単純なフォーマットであるが、これをフレームワークとして議論することで、非常に効果的に議論ができる。

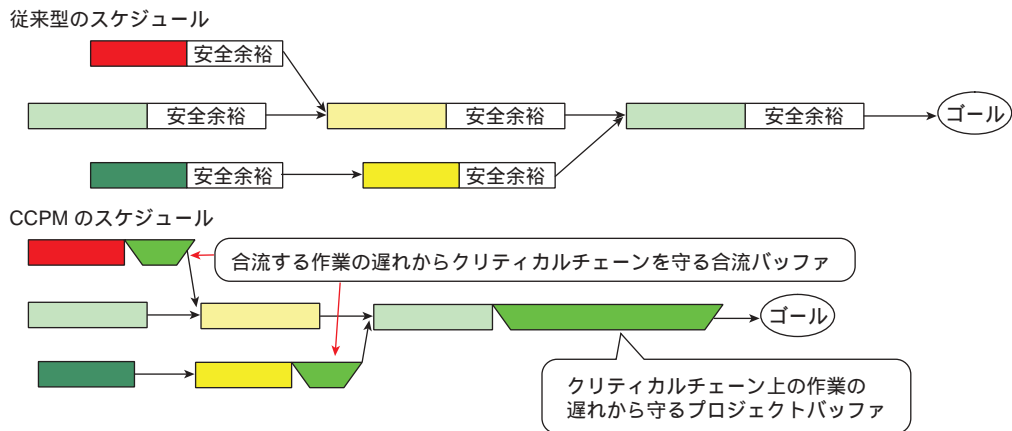
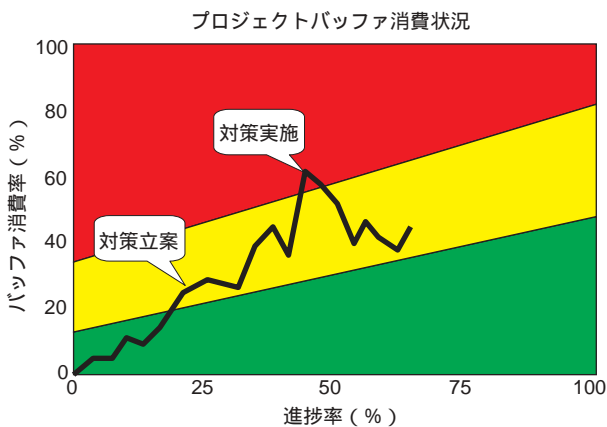


図 2 CCPM のスケジュールのイメージ  
( Fig.2-Concept of the CCPM schedule )



緑：問題なし．特に何もしない．  
 黄：注意．対策を検討．ただし実施はしない．  
 赤：危険．考えておいた対策をすみやかに実施．

図 3 バッファ消費状況  
( Fig.3-Buffer consumption status )



図 4 ソフトウェアかんばん  
( Fig.4-Software development information sign )

### 3.4 朝会<sup>参3)</sup>

朝会とは、その日の仕事開始時にチームメンバーがソフトウェアかんばんの前に集まり、15分間程度で「昨日やったこと」、「今日やること」、「問題点」について話し合う場である。

筆者らはCCPMを採用しているので、さらに「今担当している作業はあと何日かかる」ということについても報告することとした。

## 4 評価

今回の取り組みに対する評価を以下に示す。

### (1) 見える化

プロジェクトの状況や各自の状況の見える化により、状況や異常が把握できるようになったことはもちろんである。さらに様々な情報を壁に貼り出すことで以下のような効果もあった。

- 1) シートの前で対話が行われるようになり、コミュニケーションの活性化にもつながった。
- 2) 問題発生時に原因が人にあると考えがちであったが、問題そのものに目が向くようになった。
- 3) 作業や開発プロセスについての自分たちの良いところ、まずいところなど、様々な気づきを得ることができた。

### (2) プロジェクト期間と残業時間

今回の取り組みは、試行プロジェクトで評価を行い、その後、本格導入という形で進めた。プロジェクト期間は、試行プロジェクトで従来の2/3、本格導入では従来の4/5という期間短縮を行うことができた。また残業時間についても前回のプロジェクトと比較して平均で

25%減という結果となり、効率化の効果があった。

### (3) 各作業の見積りと実績

今回の開発では、各作業は「何のトラブル・割り込みもなく、その作業だけに集中できて順調に進んだ場合、最短でどれくらいの期間がかかるか」という値で見積りを行った。このため割り込みやトラブルがあれば実績は見積り期間より長くなる。見積りと実績がどれくらいのばらつきがあったかを図-6に示す。

このような見積りを行った場合、約1/3のタスクだけが見積り期間内に完了し、それ以外のタスクは何らかの不確実性による問題が発生し、期間が延びることになり、最高で7倍に延びるものもあった。従来の考え方で、各作業に安全余裕をもたせ各作業の期限を守ろうとすると、作業ごとに非常に多くの安全余裕が必要となる。このことからCCPMのバッファ管理は非常に有効だと言える。

また、クリティカルチェーン上の作業について、見積り期間と実績期間を比較すると、平均して見積りより4割程度の遅れが発生していた。このことから、必要なバッファの量は、CCPMのガイドライン通りクリティカルチェーンの半分程度が妥当であると言える。

### (4) 「あと何日」の進捗管理

実際に作業を行っている人が「あと何日」を毎日見積もることになる。この質問に答えるためには、作業の段取りや、どれくらいのリスクがあるかといったことを考えなくてはならない。これらはソフトウェア技術者としてとても大切なスキルであり、これを毎日聞くというのはとても良いトレーニングとなった。しかも、この質問に答えるために、また周りの人に理解してもらうために、担当者は作業のブレークダウンを行い、作業詳細の「見える化」を自主的に始めた。1~2時間の単位にプ

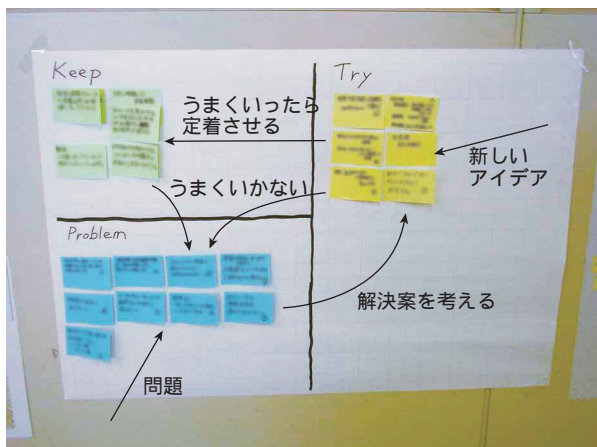


図5 KPT  
(Fig.5-KPT)

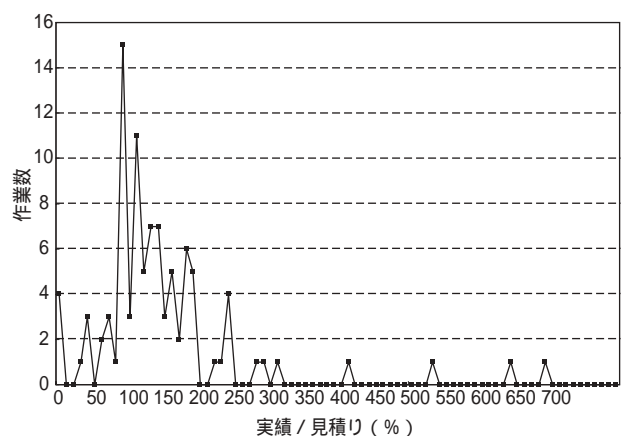


図6 見積りと実績のばらつき  
(Fig.6-Differences between estimate and achieved result)

ブレークダウンができると、ほぼ予定通りに作業が進むということもわかった。逆にうまくブレークダウンできない作業は不確実性が高いということである。

#### (5) バッファ管理

従来の進め方だと、すべての作業の進捗が重要であったが、実際、マネージャーがすべての作業に集中することは難しく、重要なことを見失いがちであった。CCPM では、個々の作業ではなく、バッファで進捗を管理するので、プロジェクトにとって大切なところに意識を集中させやすかった。また、不確実性による問題が発生した場合でも、バッファの消費状況を見ることで、対策の必要性を判断することができた。最高で見積りの7倍かかるタスクもあったが、こういったタスクがあった場合、従来型であれば、他の作業との影響を判断することができなかつただろうし、スケジュールの頻繁な見直しが発生したであろう。

#### (6) 改善意識、問題意識

各自の作業やプロジェクトの状況を見える化したこと、定期的なふりかえり会の実施により、うまくいっているところはどこか、うまくいっていないところはどこか、ということをもみんなが自分たちで考え、「さらにうまくいくためには？」ということをも、みんなで話し合い、知恵を出し合って進める雰囲気が出てきた。また、「ソフトウェアかんぱん」についても、自分たちでわかりやすいように改良していった。このように常に問題意識を持ち、自分たちで改善を進めていく雰囲気が出てきたのは非常に大きな成果であると考えている。

また、メンバーから「CMM によるプロセス改善活動の意義が理解できてきた」との声も聞かれ、CMM によるプロセス改善活動も活性化してきた。これも改善意識向上の結果であり、今後はさらなる相乗効果が期待できると考えている。

#### (7) 現場の声

これらの取り組みを行ったメンバーの感想を以下に示す。

- 1) メンバー達が自ら動きだしている。以前はどうしても自分の守備範囲内で仕事をしようとしていた
- 2) 言いたいことが言える状態でありがたい
- 3) 自発的行動ができてきた
- 4) 元気にやれている
- 5) モチベーションがあがってきた
- 6) 作業漏れがなくなり、頭の中のものが見えてきた
- 7) 新しいやり方が楽しい
- 8) 次期開発でも活かせるようにしたい

これらの感想から、本活動により、モチベーション向上や自発的行動についても効果があったといえる。

#### (8) 課題

これまで述べたように、本活動において多くの成果が得られたと考えているが、一方でさらに改善すべき課題もある。以下にその課題を示す。

##### 1) マンネリ化防止

ある程度活動が定着してくると、どうしてもマンネリ化といった現象が現れてくる。

##### 2) 手段が目的に

「見える化」はあくまで目的達成のためのツールであるが、「見える化」自体が目的になりがちである。

##### 3) 人材育成

本活動を継続・拡大していくためには推進者の役割が非常に重要である。また、メンバーにおいてもコミュニケーション能力といったヒューマンスキルがとても重要である。

今後はこれら課題に取り組み、更なる発展を目指していく。

## 5 今後の展開

今回の試行プロジェクトによる「見える化」活動の成果を受けて、他部門への展開を開始した。「見える化」活動の推進は、Active-V 推進室という専任組織が支援を行っていく。

#### (1) 組織横断的な人材の活性化活動

開発業務は知識労働であり、人に大きく依存するという観点から、PFU では従来から人材の活性化や育成に「知識創造」、「夢やアイデアの実現」活動などを通じて取り組んでいる。これらの活動は Active-V 推進室が、現場の開発部門と密接に連携を取りながら組織横断的に支援や推進を行っている。この従来活動に加え「見える化」活動も人材の活性化手段として推進を開始した。

全社の立場でこれら活動を多面的に統合展開することで、人材や組織の活性化はもちろん、社員が自発的かつ継続的に成長する元気な職場づくりや企業風土づくりを目指している。

#### (2) 活用組織の拡大と高度化

「見える化」活動による社員の活性化を推進拡大するため、様々な活動を企画展開している。

具体的には、講演会や活用事例発表会の開催、ホームページの開設運営（導入意義、効果、活用事例紹介など）、メールマガジンの発行、利用相談会や利用説明会

の開催などの取組である。

さらに、「見える化」活動は継続的改善を目指した取組であることから、その活用形態や運営方法も利用者自身の創意工夫により日々進化していく。このため、活用チーム間のネットワークを充実強化することで実践チーム間の情報共有や連携を強化し、全体的な活用の高度化を推進していく。そして、活動のマンネリ化防止や「見える化」自体が目的となりがちな活動の方向修正を適時図っていく仕組みづくりを実現する。

### (3) 知識創造活動との連携

PFU では「知空間」と名付けられたネットワーク上のバーチャルな情報交流の場を運用している<sup>参4)</sup>。

「知空間」では、組織横断的な知識共有や知識創造活動を 2002 年 12 月から実践活用している。

しかしながら「知空間」では相手の顔が見えないバーチャルな場であることから、文書化された形式知の流通はできても、直接会話や共通体験を通じて伝播されるケースが多い暗黙知の共有や流通展開は進みづらいという課題があった。さらに、コミュニケーションにおいて重要な感情の伝達が IT システムでは十分に行えない。そのため、単に情報の透明性を確保しただけではコミュニケーションを活性化したことにはならない。

これに対して、「見える化」はその実践を通じてチームメンバー間の直接対話が促進される効果がある。また、直接コミュニケーションする場を通じた Face to Face の対話により問題や課題の解決に向けたメンバー間の知の結集を促進できる効果も期待できる。つまり、知識共有や知識創造という観点からみると、「見える化」と「知空間」は相互に連携補完しあう関係にあると言える。

したがって、PFU では「見える化」と「知空間」の連携実践モデルの構築を目指した活動展開も視野に入れて実現を目指している。

### (4) 北陸先端科学技術大学院大学殿との連携

2005 年度から北陸先端科学技術大学院大学の知識科学研究科殿（以下、JAIST と略す）<sup>参5)</sup>と PFU 間では、知識創造活動と「見える化」活動の利用拡大や高度化に関する共同研究を開始している。

2006 年度の JAIST 殿との共同研究テーマは、「見える化」と知識創造活動の連携提案と実践評価であり、学習する組織構築へ向けた PFU 独自のユニークな方策の研究提案と実践を目指している。

## 6 むすび

今回の取り組みを通じて分かったことは、ソフトウェア開発においては、技術やツールに目がいきがちであるが、それらを使うのは人間であり、やはり人が重要であるということであり、人が納得感ややりがいを持って仕事に取り組むことが大切であるということである。

これはソフトウェア以外の開発や、業務全般に言えることであり、実際に「見える化」の手法は様々な業務での活用が始まり、効果を発揮しつつある。

今後はこれまでの取り組みをさらに発展させ、社員が自発的に行動し、協調することで、社員の総和以上の力を発揮する組織作りに取り組んでいく所存である。

最後に、今回の取り組みに際し、PFU でのセミナー開催や、アドバイスなど、多大なご協力をいただいた、富士通 (株) の和田 憲明氏、株式会社永和システムマネジメントの天野 勝氏、また、JAIST 殿と PFU 間の共同研究に関連して講演会や事例発表会など、各種社内プロモーション活動を通じて学術的な示唆を数多くいただいた JAIST 知識科学研究科の近藤 修司教授、田口 剛史氏に感謝したい。

### 参考文献

- 参 1) エリヤフ・ゴールドラット：クリティカルチェーン なぜ、プロジェクトは予定どおりに進まないのか？、ダイヤモンド社、1 版、東京、(2003)。
- 参 2) 岸良：CD-ROM 付 目標を突破する実践プロジェクトマネジメント、中経出版、1 版、東京、(2005)。
- 参 3) オブジェクト倶楽部 プロジェクトファシリテーション資料  
プロジェクトファシリテーション プレゼンテーション  
<http://www.objectclub.jp/download/files/pf/ProjectFacilitation20051124.pdf>  
プロジェクトファシリテーション価値と原則編  
<http://www.objectclub.jp/download/files/pf/ProjectFacilitation.pdf>  
プロジェクトファシリテーション実践編：朝会ガイド  
<http://www.objectclub.jp/download/files/pf/MorningMeetingGuide.pdf>  
プロジェクトファシリテーション実践編：ふりかえりガイド  
<http://www.objectclub.jp/download/files/pf/RetrospectiveMeetingGuide.pdf>
- 参 4) 山口、吉田：ナレッジマネジメントの導入と実践、*PFU Tech.Rev.*, 15, 1, pp.70-76 (2004)。
- 参 5) 北陸先端科学技術大学院大学 知識科学研究科 近藤修司研究室  
知識科学を基盤に人間力を向上する「成功の宣言文」  
成功の宣言文 - 全国版  
<http://www.success-poem.com/>