連載　プロマネの現場から
第１回　システム構築ソリューションとシステム開発の現場の間にて

蒼海　憲治（大手 SI 企業・金融系プロジェクトマネージャ）

私はＳＩベンダーにおいて、金融系分野の業務システム構築を担うプロジェクト・マネージャ（以下、プロマネ）です。そして、ここ数年は複数のシステム開発プロジェクトを統括するプロマネとして、プロジェクトを推進しています。

システム構築ソリューションにおいては、対象とするソリューション分野の知見でいかに差別化できるか、ということと、提案したソリューションを確実に開発するための高いエンジニアリング力が必要とされます。
前者の業務ソリューションとしてのイノベーション、新規ソリューションへの取り組みについてはＳＩベンダー各社によって得意分野・課題認識が多様であり、またの機会に述べたいと思いますので、今回は後者のシステム構築のためのエンジニアリング力についての現状の課題とその取り組みについて考えたいと思います。

先日、１４年ぶりに、エドワード・ヨードン氏の「ソフトウェア管理の落とし穴―アメリカの事例に学ぶ」を読み直しました。１９９２年刊の本書は、１９８０年代を通して、アメリカ人のＩＴ技術者は、品質面では日本、コスト面ではインドをはじめとする諸外国のエンジニアに職を奪われてしまうのではないか、という危機感・・問題意識から書かれています。ヨードン氏曰く、アメリカ人プログラマーは、「とうの昔に絶滅したドードー鳥」だ、と。

２００８年現在、日本がアメリカ人のＩＴ技術者にとってかわるということは、杞憂に終わったわけですが、「日本」という言葉が、中国・インド・ブラジル・東欧等のＩＴ技術者に置き換わったと考えると、この予測が誤りというよりは、少し時期尚早な警告だったのだと思います。そして、かつてチャレンジャーであったはずの日本の立場が、当時の「アメリカ」にそっくりそのまま置き換わった感があります。

絶滅したドードー鳥にたとえられつつも、システム開発の現場においては、対象とするシステムの難易度が増し続けています。具体的には、
・増大するステークホルダーと、その要求の高度化・多様化
・開発環境やシステム稼動環境のオープン化
・ビジネスのライフサイクルの短期化とあわせた開発工期の短期化
・社会インフラとしてのシステムの安全性・信頼性の要求
・システム統合・システム間連携による対象システムの高度化・複雑化
そして、
・低コストかつ圧倒的な要員供給源であるオフショアの出現
を課題と受け止め、要求事項に基づいたスコープで、納期・品質・コストを満たしつついかに円滑にプロジェクトを推進していくか、日夜、汗と知恵を絞る日々が続いています。

システム開発には、狼男を一発で倒すような「銀の弾丸」はない、と言われますが、たった１発では無理でも１０数発の弾丸を適切な場所に撃ち込めば倒すことができ

る・・理想のプロジェクトの実現に近づける、とヨードン氏はいいます。１９９２年当時の効果的な弾丸の候補として、１１の項目が挙げられていました。

1. 優れたプログラミング言語
2. 優れた人材
3. 自動化ツール
4. ＪＡＤ：共同アプリケーション設計
5. ＲＡＤ：迅速アプリケーション開発
6. プロトタイピング
7. 構造化技法
8. 情報工学
9. オブジェクト指向方法論
10. ソフトウェアの再利用
11. ソフトウェアのリエンジニアリング

この実装レベルでの弾丸に加えて、プロジェクトを円滑に推進するための取り組みとして、「ピープルウェア」「ソフトウェア・プロセス」「ソフトウェア方法論」「ＣＡＳＥツール」「ソフトウェア・メトリクス」「ソフトウェアの品質保証」などの観点が、十分条件として述べられています。

改めて一覧を眺めてみると、前者の道具立ては様変わりしたものの・・たとえば、「優れたプログラミング言語」では、Ｗｅｂの到来とＪａｖａの普及、また「ＪＡＤ・ＲＡＤ、プロトタイピング」の狙いであるＸＰ等のスパイラルアップ的な開発方法の登場、「再利用」の観点でのＳＯＡ等を位置づけるのであれば、挙げられている項目については現在でも、ほぼそのまま通用するように思えます。

また、プロマネの立場からすると、後者の「ピープルウェア」「ソフトウェア・プロセス」の観点の重要性はますます高まってきています。これに加えて、要求工学やリスク管理の観点を入れた上で、これらの取り組みが全て上手くできているのであれば、実プロジェクトでバーストした不採算案件の９割方は解消するはずと思われるのではないでしょうか。

でも、これは不思議ではなく、ここで挙げられている生産性向上施策・品質向上施策は、そもそも一斉に導入することはできず、組織的な活動として、３年の中期計画から１０年以上の長期計画として継続して取り組むべき対策であり、組織的なシステムエンジニアリング強化の取組みが重要である、ということを示しています。

ところで、膨大・広範なソフトウェアエンジニアリングの世界を体系化しようとする取り組みとして、ＳＷＥＢＯＫ（ソフトウェアエンジニアリング基礎知識体系）やＰＭＢＯＫ（プロジェクトマネジメント知識体系）、ＥＡフレームワークの知識体系等がガイドというかたちで整備されつつあります。

これらの成果は、既に一部取り込まれつつありますが、普及・適用の動きはこれから加速していくと思います。ただし、いずれの取り組みにおいても、現在の組織の実力・身の丈にあった取り組みでなければ実効性がありません。過去数十年にわたるソフトウェア工学上の蓄積とともに、自社・自プロジェクトの実力を踏まえ、「プロセス標準」「プロセス改善」「メトリクス」「ソフトウェア工場」等を取り込み、システム構築ソリューションの高度化を進めていく必要があります。

今後、プロマネの現場における課題認識とその具体的な取り組みについて、ご紹介させていただければと思います。

連載　プロマネの現場から
第２回　プロジェクトメンバー幸福のための「プロセス改善」

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

４年ほど前、約３０億円規模のソフトウェア開発プロジェクトをカットオーバーした直後、ソフトウェア開発プロセスの能力成熟度モデルのアセスメントを受審しました。
プロジェクトそのものは、納期面・品質面・収益面とも申し分なく、また、プロジェクトの開始に先立って、各種標準（工程標準、成果物標準、設計標準、ツール標準）・共通部品等を整備し、プロジェクト実行計画書・品質保証計画書を策定し、それに基づいたＰＤＣＡを回していたこともあり、プロジェクトのメンバーともども余裕を持って審査を受けることになりました。

そもそもの能力成熟度モデルについてですが、ソフトウェア開発を行う組織がプロセス改善を行うためのガイドラインであり、カーネギー・メロン大学の下部組織であるソフトウェア工学研究所（ＳＥＩ）が考案したＣＭＭＩ（Capability Maturity Model Integration：能力成熟度モデル統合)等いくつかの種類があります。私たちが受審したのは、ＣＭＭＩの作成者も参画して策定された「ISO/IEC 15504」・・通称、ＳＰＩＣＥ（Software Process Improvement and Capability dEtermination）というモデルに基づくものでした。

ＳＰＩＣＥモデルにおいては、ソフトウェア開発を行う組織のプロセスの成熟度を、６段階のレベルで評定します。
レベル０　不完全なプロセス（Incomplete process）
レベル１　実施されたプロセス（Performed process）
レベル２　管理されたプロセス（Managed process）
レベル３　確立されたプロセス（Established process）
レベル４　予測可能なプロセス（Predictable process）
レベル５　最適化しているプロセス（Optimizing process）

レベル２以上の状態になって、ようやくプロセスの状況が管理者にとってプロジェクトが「見える化」になることがわかります。
そこで、レベル２相当と認定されるために必要な主要プロセス・・「プロジェクト管理」「品質保証」「構成管理」等の「マネジメント系プロセス」とともに、実開発メンバーにとって、現場の日々の作業に直結する要求分析・設計・構築・ソフトウェアテスト・・といった「エンジニアリング系プロセス」を審査の対象としました。

ところで、アセスメントの結果は、どうだったでしょうか？

実はちょっと残念なもの・・・いえ、プロマネにとっては極めて手厳しいものでした。
要求分析・設計・構築・ソフトウェアテスト・・といった「エンジニアリング系プロセス」は高く、ルールに則って開発されており、結果として、高品質・納期遵守を達成している、ということで概ねレベル２の評定でした。
一方、プロマネの本業であるはずの「マネジメント系プロセス」については低く、プロジェクトが円滑に推進しているのはプロセスによるものではなく、プロセス以外の方法（人的スキルや能力）で行われている可能性が高い、との指摘でした。当時、プロマ

ネとしては、マネジメントがエンジニアリングの足を引っ張っている、という事実を突きつけられたようで、衝撃であるとともに、プロジェクトメンバーに対して、とても申し訳なく思ったことを覚えています。

その後、「プロセス以外の方法（人的スキルや能力）」で押さえられているという暗黙知を、プロセス標準のかたち等に明文化し、極力、形式知とする努力を続けるとともに、新規メンバーへのプロジェクトの導入教育の実施、また、プロセス標準改訂毎の再教育の実施を続けています。また、１年に一度のペースで、アセスメントの再審査を受けて、改善指摘を受けつつ、改善活動を続けています。

アセスメントを受けてわかったこととして、

１．改善活動は、漸進的にしか進まない
　１レベル上げるのに１～２年かかるといわれますが、先に進むと新しい景色・・新しい課題が見え、それに取り組むというサイクルを回す必要があります。

２．いきなりレベル３以上の組織を作ることはできない
　ベスト・プラクティスは参考にはなりますが、身の丈に合わない理想の標準プロセスを持ってきても、現場はそれではまわりません。標準プロセスからのテーラリングと、現場のプロセスのボトムアップの２つの方向が必要になります。

３．途中のレベルを抜かすことはできない
　たとえば、レベル１からレベル４へというような飛躍はいきなりできません。漸進的アプローチと同様に、組織風土の変化をともなうからです。

４．レベルが低い段階は、新技術の導入より先にすべきことがある
　統合開発環境やＣＡＳＥツール等は、上手く導入できれば業務プロセス見直しの良い契機になると考えられます。しかし、有効に機能させるためには、構成管理プロセスやベースラインの考え方の整備が先に必要となります。

「レベル１からレベル２への移行が一番困難である」といったのは、成熟度モデルの提唱者であるワッツ・ハンフリー氏ですが、それは、混沌とした組織文化から、近代的なマネジメントの文化へという価値観の転換が必要であるためだと思います。
レベル１からレベル５を目指す取り組みは、１０年のスパンでプランすることを覚悟する必要があるため、経営のコミットが必須となります。

ところで、現場のプロマネがほしいのは、ＣＭＭＩやＩＳＯのソフトウェア開発プロセスの成熟度のレベルの高い評定ではなくて、「予測可能性」・・あとどれだけの工期と工数で、どの程度の品質のソフトウェアができあがるのか？ということだ、と思います。
工期・工数・品質の指標は、開発プロセスの品質・生産性とリンクすることを踏まえると、「最下層のあえぎ」と評される、成熟度レベルが１未満のプロジェクトチームを救う重要な方策は、開発プロセスの改善です。
プロジェクトメンバーの幸福のためには、プロジェクトの現場が３Ｋから３Ｔ（楽しい、給料高い、定時に帰れる）へ変身する必要がありますが、そのための１つの有効な弾丸が、全員参画型のプロセス改善活動であると思います。

そのため、現場任せで、プロジェクトが円滑に回るまではプロセス改善に手をつけないという事態を避けるためにも、プロセスが不十分なうちは、組織側からのコミットと

少し手厚いフォロー、人のアサインが必要になります。その上で、プロセスの成熟度にあわせて、冗長なプロセスや人のアサインをスリム化するという順序を間違えないことが大切です。

そして、高度化を目指す対象プロセスも、自組織やプロジェクトの課題を踏まえて選ぶ必要があります。たとえば、見積り精度を高度化したいのであれば「見積り」プロセスを選び、また、レビューやインスペクションの方法を改善したければ「ピア・レビュー」プロセスに取り組むということから始めることも、プロジェクト現場のモチベーションと業務への効果の観点から有効です。

最後に、プロマネ及びプロジェクトメンバーは、アセッシー・・・アセスメントを受ける立場なのですが、組織としてのプロセス改善のためには、アセスメントを実施するアセッサーの協力・参画も必須であり、いかに一体となったプロセス改善活動とすることができるかが成功の鍵であると思います。

連載　プロマネの現場から
第３回　ソフトウェアにおけるオフショア開発への取り組み

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

ソフトウェアにおけるオフショア開発の取り組み・・既に先行して実践されている方が多いかと思いますが、担当業種がセキュリティを重要視する金融機関ということもあり、３年ほど前から中国等とのオフショア開発に取り組んでいます。

「日経ソリューションビジネス」の２００８年3月１５日号において、インドのＩＴサービス企業の日本進出が加速しているとし、インフォシスが、日本ユニシスと提携したニュースとともに、タタ・コンサルタンシー・サービシズ（ＴＣＳ）が、「技術力」と「圧倒的な開発ボリューム」を理由に受託開発を受注したこと、この契約の際、「価格」提示がなかったことが注目すべき点である、と報じています。

今後、ウィプロが、２０１０年までに、社員を倍増し、１５万人超体制とするように、インド全体で、現在の１３０万人から、２０１５年には３０２万人、中国全体で、現在の９０万人から、２０１５年には３２４万人、インドと中国あわせて、６２６万人のＩＴ技術者になる見通し、とのレポートもあります。
日本のＩＴ技術者は、５０万人とも１００万ともいわれますが、地理的な距離・言語の壁があるものの、ネットワーク・コストが限りなくゼロに、またオフショア先技術者の語学や業務に対する学習意欲を考えると、ついに黒船が到来した感があります。

また、従来からのオフショアの流れについては、エドワード・ヨードン氏の「アウトソース」において、ほぼ全ての業務・職種・スキルにおいて、自国内でしかできないという安住の地はなくなった様子が描かれており、２１世紀は知識労働者の業務がオフショア先へアウトソーシングされる時代であること、もちろん、ＩＴ技術者はその例外でないことと説明されています。
ただし、オフショア・メリットが、日米で異なる点は注意が必要だと思います。
日米ともに共通するメリットは、「コスト削減」「要員調達」ですが、これに加えて、米国の場合、もともと業務レベルが高くないという事情があるため、「品質」「生産性」のメリットがあるといいます。たとえば、コールセンターやヘルプデスクの導入初期時の事例をみると、日本の場合、窓口応対等の極め細やかなサービスが優れているため、一時的にマイナス評価となってしまうことを考慮する必要があると思います。

したがって、従来からのオフショアの流れとともに、今回のオフショア企業の日本国内への進出を踏まえると、プロジェクトの現場は、

１．国内における低コスト・高調達力のパートナー出現への対応
２．オフショア先への業務委託によるメリット享受への対応
の大きく２つの対応をする必要があります。

１つ目の低コスト・高調達力のパートナー出現に対しては、将来的には、ＳＩベンダーとして、またＩＴ技術者として、オフショア先の企業・技術者に代替されない努力として、

（１）要件定義や基本設計の上工程はもちろんのこと、さらに上流の工程、イノベーション創出のプロセスでいえば「コンセプト」「モデル」「デザイン」等へシフトするとともに、
（２）高品質・高生産性のプロセスを確立、さらなる高度化、に取り組む必要があります。

２つ目のオフショア先への業務委託においては、まずオフショアの目標・・オフショアによって、何をメリットにしたいかを決める必要があります。コストメリットを採るのか？、大規模な要員調達を採るのか？、特殊技術（たとえば、ＣＯＢＯＬ技術者）の要員確保なのか？
次に、目標を明確にした後、プロジェクトの立ち上げにあたっては、オフショア先の企業を育てるつもりで、過去長年にわたり国内で培ってきた組織的なエンジニアリング力を使って、一緒になって品質システム・開発標準・手順等の整備・作成すること。それに基づいて、人材教育・訓練を行うことが必要になります。

オフショア先との連携については、アウトソーシング（外部調達）型と、インソーシング（社内調達）型がありますが、いずれの場合を採るにせよ、オフショア先へは業務丸投げではなく、日本人マネージャによる委託先プロジェクトの適正なマネジメントにより、オフショア開発のＰＤＣＡを確実に回す必要があります。

技術面では、開発基盤となるフレームワークや共通プログラム、ワークフロー・プロセスの共有のしかけを準備・提供すること。
人の面としては、ブリッジＳＥという案件の開発工程という「点」で連携するレベルから、ブリッジＰＭという領域全体及びシステム開発ライフサイクル全体という「面」で連携するレベルでの対応を図るレベルまで、提供できる人材の質・量を踏まえて決定する必要があります。そして、ブリッジＳＥ・ブリッジＰＭを日本側・オフショア先双方で育成していくことになります。

「大連は燃えている」の中で何徳倫氏は、オフショア成功のための鍵は、日本人のＳＥを直接オフショア先の現地業務委託企業へ派遣せよ、と提案されており、そのメリットを４点挙げられています。

１.日本人のプロジェクト担当ＳＥは、プロジェクトの設計段階から参加しているので、システムの構成と業務知識を熟知し、現場での指導が十分できる。ブリッジＳＥを通じての無駄なやり取りが減少することによって、開発の品質を高め、コストも下がる。
２．日本人ＳＥを現地へ派遣することにより、現場サイドの技術者同士の交流とコミュニケーションを深めることができる。
３．現在、オフショア開発を行っている企業の中で、多くの経営者は、もっとたくさんのプロジェクトをオフショア開発に出したいが、現場サイドの担当者が、自分の責任だけ考えて、なかなか出せない状態だということを良く聞く。この問題を解決するため、その担当者たちをオフショア先へ派遣することによって考え方を転換させるという有効な方法がある。
４．日中間の若い技術者同士の国際交流を行うことで、日本人の若い技術者に国際感覚を味わわせて、やる気をどんどん出させ、会社のグローバル化のプラスにする。

項番１から３については、足元のオフショア開発の成功のために必要ですが、さらに項番３及び４で指摘されている意識改革・国際感覚の醸成は、近い将来に必要となるであろう日本人ＩＴ技術者の海外でのソフトウェア構築ビジネスを展開するための布石ともなると考えています。

これまで様々な理由づけをしつつ、国内事情を優先し、プロジェクトチームの組成においては、ほぼ無条件に国内パートナーを優先してアサインしていました。しかし、国内パートナーは、長年のつきあいの中で慣れ親しんでおり、プロセスとして明文化できない領域を担保するというメリットがある一方、ややもすると馴れ合いの関係になっているというデメリットもあると思います。
当初はピークカット時の要員調達だけを理由に導入した場合であっても、オフショアの規模が拡大するに伴い、国内パートナーとの棲み分け等の調整も入るとともに、ＩＴ業界全体の懸案事項の一つであった多重委託契約の商慣習にもメスが入ることになります。

自分たちの得意技を磨くこととオフショア先との協業を図ること・・その結果、マクロな観点でみると、ＩＴ業界における世界規模での人材の最適配置が進むことになります。正直、まだまだ胸中、不安と期待が入り混じっているのですが、このプロセスを楽しみながらプロジェクトを推進できる心境になりたいと思っています。

連載　プロマネの現場から
第 4 回　システム構築ソリューションにおける創造プロセス

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

システム開発のプロジェクトにおいて、プロマネ及びＳＥとして、常に忘れてはならないのは、自分たちの役割・ミッションです。

馬場史郎氏は「信頼されるＳＥの条件」（日経ＢＰ社）の中で、「ＳＥとは何か、どんな仕事か」と問われた場合、「ＳＥは社会システム変革の担い手である」と断言しろ、と言い切られています。

過去そして現在のＩＴ技術者が営々と積み重ねてきた取り組みの結果が、社会インフラとしてのＩＴ基盤・サービスとなっているのは間違いありません。その社会システム変革の担い手としてのＳＥは、顧客に対して、様々なソリューションを提供する必要があります。

ソリューションが、問題解決という意味であることから、システム構築ソリューションとは、システム構築によって、顧客の経営戦略上・業務改革上の様々な課題を情報技術（ハードウェア、ソフトウェア、パッケージソフト、スクラッチ技術など）を用いて解決することになります。

したがって、システム構築ソリューションを提供するＳＩベンダー及び、そこでのプロマネ及びＳＥは、顧客の言われたことを聞いて、そのままプロダクトを導入する存在、またはアプリケーションプログラムを開発するという存在ではなく、顧客の業務を熟知の上、顧客が明示的にできない要求事項を形式知とし、顧客ビジネス・業務改革のグランドデザインの中でのシステム化をプランニングする提案をすること。そして、その中で適切なハードウェア・ミドルウェア・業務パッケージを選定し、カスタマイズ及びスクラッチによる開発というトータルなインテグレーションを提供する主体であるべきです。

システム構築ソリューションの取り組みの現状を考えるにあたって、そもそものソリューションが創出されるプロセスを振り返ってみます。

２０００年のアメリカズ・カップのテクニカル・ディレクターを務められた宮田秀明氏が、イノベーションのための創造プロセスをこう述べています。

「哲学」→「ビジョン」→「コンセプト」→「モデル」→「デザイン」→「シミュレーション・変更とフィードバック・検証」→「意思決定」「テスト」→「実行」（＊）

ビジョンからコンセプトを作りモデル化していく。そして、実行まで終えると、再びビジョンに戻すというスパイラルを回していくことで大きな価値を生み出す。

ところが、多くの会社・組織は、コンセプトやモデルを前例や他社事例から採り、自らが考えようとしていない。「デザイン」「シミュレーション・変更とフィードバック・検証」から入り、小さな改善をコツコツと行うことに集中する。その成果は上がることは確かですが、コンセプトから得られる価値に比べると、とっても小さな価値となってしまいます。コンセプトやモデルが前例や他社事例・・借り物であることが多いとの指摘は、耳が痛いです。

デザインは、「モデルを具体化する」作業であり、「シミュレーション・変更とフィードバック・検証」によって、デザインの実現性を確実にします。たとえコンセプトやモデルがよくても、デザインができなかったり、コンセプトから逸脱するようなデザインになってしまったら、そのプロジェクトは失敗です。

技術者に対しては、シミュレーション以降のことが仕事のすべてだと、錯覚してしまわないようにと釘をさされ、理想のサイクルを回すことで、大きな価値を生むマインドを失わないこと。自分のプロジェクトが、この創造プロセスのどの工程から回っているかを、常に意識することが大切であると指摘されています。正直この指摘、目からウロコでした。

システム構築ソリューションにおいても、ＩＴ戦略立案工程において、「ビジョン」を練り、システム化の企画工程において、「コンセプト」づくりを行う。
基本構想・要件検討工程で、「モデル」「デザイン」「シミュレーション・変更とフィードバック・検証」を実施した上で、以降の設計・開発の工程を実施していきます。

ソリューション＝顧客の問題解決という原点から、システム構築ソリューションを捉えなおすと、
・顧客自身が、何が問題かわからずに困っているような場合は、「ビジョン」「コンセプト」レベルからコンサルティングに入り、システム構築につなげること
・顧客自身が、問題の原因はわかっているけれども、どう解決したらよいかわからないで困っているような場合は、自社の得意な情報技術（ハードウェア、ソフトウェア、パッケージソフト、スクラッチ技術など）を組み合わせ、「モデル」「デザイン」レベルからソリューション提案を行う
ということになると思います。
実際のプロジェクトにおいては、ＩＴ技術をベースとしたインプリメンテーションを中心に、スキル及び体力の過半を割かれることが多いと認識していますが、顧客のビジネスモデルや業務を理解した上でのアプローチを常に心がける必要があります。
また、たとえ開発工程からの参画の場合であっても、プロマネは「コンセプト」レベルの意識を持ってプロジェクトに望むことが大切である、と思います。

最後に、宮田氏の「イノベーションのための創造プロセス」において、一番目のプロセスは、「哲学」であるといわれています。以下の工程においては、経験と論理と直観を駆使して、意思決定を図っていく必要があります。当然ながら、プロジェクトの推進途上においては、この経験と論理と直観に基づく科学的な意思決定ができない場合が生じます。
その際、拠りどころになるのは、「哲学」と「ビジョン」である、と。そして、この「哲学力」は技術の習得以上に難しく、困難なプロジェクトの実戦経験を通して鍛えられると指摘されています。
本メールマガジンの連載「情報システムの本質に迫る」において、芳賀正憲氏が「情報と情報システム」の根本から遡って議論を展開されていますが、ここで論じられている知育面を理解した上で、実戦での体育面を通して経験を積み、それを再び、より進化した知育として表出することがＩＴ技術者の成長にとって必要ではないか、と考えています。

（＊）宮田秀明「理系の経営学」、「仕事のやり方間違えてます　―　成功を手にする「理系思考」10 の法則」

連載　プロマネの現場から
第５回　ＳＥとしての学ぶ姿勢と成長可能性について考える

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

初めてのプロジェクトを前に少し不安を覚えながらも、希望に夢を膨らませている新人たちが、プロジェクト現場に配属される今日この頃・・。若い新人たちの今後の活躍に期待する一方、新人たちの指導員やコーチャーとなった２年目・３年目のメンバーが、緊張感を持って対応しているのが伝わってきて、とても嬉しくなります。

研修を終えたばかりの新人たちに話しをするのは、これで勉強は終わりではない、ということです。もちろん、これからは業務を通してのＯＪＴが中心となりますが、ＯＦＦ－ＪＴや自己研鑽を継続していくことが大切であること・・そして、それは若手だけのことではなく、ＳＥを続けていく間は一生必要であることを自戒をこめて話します。

システム構築ソリューションのカバー範囲の大きさを考えると、ＳＥやプロマネは自分自身の役割や職域の幅を広げるため、学び続ける必要があります。

馬場史郎氏は「信頼されるＳＥの条件」（日経ＢＰ社）の中で、「ＳＥの５年サイクルの危機」というものがある、といいます。

２０数歳から仕事を始め、最初の５年間ぐらいは日に日に力をつけ成長していくものの、２８歳前後で何%かのＳＥは伸びなくなる。つまり、進級できずに落第を続ける状態である。さらに、その５年後、３３歳前後になると、「成長できない落第ＳＥ」はもっと増える。こうしたＳＥの成長の節目は、３８歳、４３歳、４８歳と５年サイクルで来る。

「２８歳あるいは３３歳ごろ」に落第するＳＥのキャリアの特徴は、
①若い時、付加価値のある仕事をあまり経験していない
②ビジネスマンとして厳しく指導され、鍛えられていない
また、「３８歳、４３歳あるいは４８歳ごろ」に落第するＳＥのキャリアの特徴は、
③新しいＩＴについて行けない
④ＳＥとしての夢が持てない
という共通点が見られたといいます。

若手から中堅のＳＥは、プロダクト中心の仕事やプログラミングの仕事ばかりするのではなく、アプリケーションの開発やシステム基盤全体の構築なども担当するようにする。さらに、提案書作り、業務分析、要件定義、システム設計、プロジェクト管理といった付加価値の大きい仕事を率先してする。また、対人関係の処理能力、表現力、交渉力、リーダーシップなどを持つようにする。

中堅以降になったＳＥは、製品や技術の細かい知識が必要な仕事から、経験と応用力が要求される仕事へと、比重を移していく。要件定義や設計など上流工程の仕事、対人関係の処理能力やマネジメント能力が要求されるプロジェクト管理の仕事、非常に深い業種・業務知識に基づくコンサルタントの仕事、あるいはライン管理職を目指す。

若さにまかせてＩＴを追求することはもちろん重要ですが、同じ若い時に、顧客や仕事の進め方について多種多様な経験を積むことにより、ＳＥにとって普遍的なものの重要さを知り、それを身につけるよう心がけることが大切なのだと思います。

ＩＴ企業におけるエンジニアの年代別スキル調査を実施された、ＩＴイノベーションの林衛氏が、アンケート調査により、「向上心」について面白い結果が出たことを紹介されていました。
その調査結果によると、入社し仕事を始めたばかりの２５－２９歳のレンジが一番意欲が高いのですが、３０－３４歳でがくんと落ち込み、以降、５４歳まで落ちたままの意欲は横ばいとなる。そして、定年を意識した５５歳以上で再び向上する。
この中だるみの現象の原因は、本人の問題であるだけでなく、組織として、社員に対して「仕事の多様性」を教えられていないからだ、とコメントされたことを覚えています。

そうはいいながら、組織が変わるまで受身でいるわけにはいきません。自己成長を考える必要がありますが、人間の成長可能性についての考え方を２つ紹介します。

ブライアン・トレーシー氏の「フォーカル・ポイント」に、１０００パーセント成長する方法が書かれています。
曰く、毎日０．１パーセントずつ成長して、１０倍成長しよう！
毎日１パーセントの１０分の１ずつ進歩すれば、１週間でおよそ１パーセントの２分の１の進歩が見込める。
１週間で１パーセントの２分の１ずつ進歩すれば、１ヶ月でおよそ２パーセントの進歩が見込める。
１ヶ月で２パーセントずつ進歩すれば、１年でおよそ２６パーセント分、生産性・成績の進歩が見込める。つまり、こつこつと人間性の育成と学習に励めば、誰でも１年で２６パーセントずつ、生産性・成績を高められるということだ。１年で２６パーセントの進歩を１年また１年と積み上げていけば、２年７ヶ月で、生産性・成績は２倍に達する。
１日に１０００分の１（１パーセントの１０分の１）ずつ進歩し、１年で２６パーセント進歩していけば、１０年後、あなたの生産性・成績は１００４パーセントにまで増えている。
複利のマジックには誰しも驚かされることですが、時間を味方につけて成長するという指摘はもっともだと思います。

では、この０．１パーセントの成長をするための方法は・・ということへのヒントですが、「仮説思考」と「仮想演習」を挙げたいと思います。

「仮説思考」とは、ある事象が発生する前または発生したがその真の原因が判明するまえにその答え＝仮説を想定することです。限られた時間の中で、すべての問題や原因をしらみつぶしに検証していくことが不可能である中で、極めて有力なアプローチ方法です。人生が有限である以上、すべての人に必要なスキルになると思います。

「仮想演習」とは、畑村洋太郎氏の「決定版・失敗学の法則」において紹介されている、自己の課題を明確にした上で、その課題をいかに解決すればいいのかを思考することです。「仮説思考」で想定した課題に引き続いての思考法であると考えています。

畑村氏、この「仮想演習」が自己成長につながるため、大いに実践することを勧められています。

仮想演習によって、「つねに自分の周囲を観察し、自分の守備範囲以外の課題を４つ５つと設定してシミュレートしている人は、少なくとも人の５倍、成長することができます。もし、「仮想演習」を日々繰り返すことにより、様々な物事に対する対処能力を、５年で５倍の成長が見込めるとすると、切磋琢磨し続けることにより、２５歳から６０歳になったとき、２５歳の自分に比べて、なんと！５の７乗＝７８１２５倍も成長することができる。

５年で５倍の前提を置く・・という前提の妥当性はありますが、５年で２倍でも、２の７乗＝１２８倍です。自分よりも、よくできる人・・っていうと、イメージ的に１０倍ぐらいかな～、と思っていたら、大間違い。

１００倍から１０００倍以上も差がついている可能性があること。たとえ４０歳からでも、目標を７０歳にすれば、６乗するチャンスがあります。

冒頭の馬場氏は、「ＳＥのキャリアは、年功序列で偉くなることではなく、年齢とともに成長することである。」と述べられていますが、一日も早く課題設定して、それに向けて徹底的に仮想演習に取り組むことで、個人はもちろんですが、組織としても成長し続けるようにしていきたいと思っています。

連載　プロマネの現場から
第 6 回　トラブルを通して人も組織も成長する

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

システムをリリースしエンドユーザの利用が始まって以降、プロジェクト・メンバーや営業担当、お客様より、「本番障害です！」「トラブル発生！」「お客様からのクレームが・・」という言葉が飛び込んできた瞬間、びくっと全身に力が入り、身体がこわばります。「待ってました！」という気持ちもいくぶんありますが、大きな息を吸い込んでゆっくりと吐き出しながら、まずは頭と心と身体が冷静になるように努めます。

多種多様な経験を積むための１つの有力な方法は、障害・トラブル・バグといった不具合発生時のトラブル・シューティングにあると思っています。致命的な障害・トラブルは絶対に避けなければなりませんが、そうでなければ、障害・トラブル・バグへの対処をいかに上手くするか、そのコントロールにかかっています。そして、障害・トラブルの対処を通して、個人レベル・組織レベル双方での絶好の改善チャンス・・個人レベルでは、能力発揮・能力育成のチャンス、組織レベルでは、プロセス改善のチャンスと捉えることが大切です。

システム構築プロジェクトにおいて、障害・バグが発生した場合、「障害管理表（障害記録票）」を起票しますが、その中には、「障害内容・現象」「直接原因」「暫定対策」「根本原因」「類似調査」「恒久対策」「再発防止策」等の記述欄が設けられています（付帯情報として、「原因分類」「影響度」等もあります）。少し簡略化した様式だと、「原因」と「対策」が各々１つずつしかないケースもあります。
ところで、「障害内容・現象」「直接原因」「暫定対策」まではどのプロジェクトでも記述されると思いますが、「根本原因」以降の対応をどこまで真面目に取り組むか・・ここに、個人レベル・組織レベルでの改善がかかっています。

個人レベルでの改善チャンスについては、柴田芳樹氏の著書「プログラマー現役続行」の中で、デバッグはエンジニアの総合力であると示されています。
柴田氏曰く、「デバッグには、論理的に推進する力だけでなく、その推論を支える膨大な知識が必要です。初心者や見習いレベルの場合には、その両方が欠けています。論理的思考は日々の指導で、知識は継続した学習で、それぞれ身につけていきます」と。
具体的には、まず、障害・バグの現状・・どのような現象が発生しているのか、その発生頻度はどの程度なのかを把握します。
次に、その原因を特定するための仮説を立てます。この仮説を立てるために、設計書やソースコードを調べたり、再現させたりという活動も含まれます。設計書やソースコード上で、原因と思われる箇所が見つかった場合には、それが原因でバグが発生していることを論理的に必ず説明できるようになること。原因と推定される箇所が複数発見された場合、そのどれが本当の原因かをじっくりと考え論理的に説明できる必要があります。
自分で把握・整理した原因と対応方法を、第三者に説明することによって、論理的な矛盾点を指摘されたり、もっと良い方法の指摘を受けること。相手に理解してもらえない場合、相手の理解レベルを考慮して、説明の抽象度を上げたり、別の観点から説明する工夫が必要になります。

この「仮説・検証」と「説明」の努力を通して、論理的思考力をベースとする能力が高まります。

次に、組織レベルでの改善チャンスについてです。
Ｅ．Ｗ．ダイクストラ氏が「プログラムテスティングは、バグが存在することを示すためには使えるが、バグが存在しないことを示すためには使えない」というように、システム開発、プログラムにバグ・障害は避けられないとも思われます。しかし、類似した障害・バグが複数発生し、プロジェクトの進行に伴っても収束しない、同じように散見されるという場合、その直接原因がミス・うっかり・・であった場合でも、個人レベルではなく組織レベルの問題と捉えて対応するべきです。根本原因は、要員・体制面であったり、プロセス面の不備であるケースが大半であり、そこに手を入れた対策を採らない限り、再発防止につながりません。
事故や災害について調査した結果確立した法則として、「ハインリッヒの法則」があります。この法則では、１件の重大災害の裏には、２９件のかすり傷程度の軽微な災害があり、さらにその後ろには、ヒヤリとしたりハッとして冷や汗が流れるような事例が３００件潜んでいます。３２９件の事象を軽視・黙認した結果が、１件の重大事故となります。３２９件のヒヤリハットを見逃し続けた責任もさることながら、問題から学べなかったことこそ問題なのだと思います。
日常のヒヤリハットを意識しつつ、組織的な改善のためのベースラインとして「プロセス標準」を持ち、再発防止策を取り込む漸進的な「プロセス改善」のＰＤＣＡを回るようにすることで「学習する組織」になれると思います。

「学習する組織」になるような組織風土をつくるために。
失敗学の畑村洋太郎氏によると、失敗を隠した結果、判明した場合、「ツケの大きさはだいたい、予兆や事実を無視したり、隠したりして「得をしたつもりになっている」金額の３００倍くらいになる」といわれます。
また、トラブルから学ぶことのできる組織とできない組織について、スコラ・コンサルトの柴田昌治氏は、こう述べられています。
「問題はあること自体が問題ではなく､解決されていっていないことが問題」である。失敗は「あってはならない」という精神論で、問題発生の都度、犯人探しが始まる組織では、現場は問題を隠し、問題解決のＰＤＣＡは回らない。人というものにはどんなに努力しても「失敗」はつきものという現実論に立った上で、「めざすものを共有する」「厳しく向き合う（協力し合う）」「当事者として自分の役割を認識する」というこの３つの進化に必要な価値観を共有することで、「学習する組織」にできるといいます。

プロマネの役割の大きな部分は、プロジェクトの推進とともに、プロジェクトの推進母体である自分たちのプロジェクトチームを「学習する組織」となるようにすることです。プロジェクト・メンバーとともに、このプロセスに取り組むことがプロマネの醍醐味であると思っています。

連載　プロマネの現場から
第 7 回　リスクマインドを高める法

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

企業の屋台骨を揺るがすような社会的な事件が相次いでいることを見るたびに、プロジェクトにおいても失敗を回避するためのリスク・マネジメントの必要性を改めて思います。そして、リスクをいかにして回避するか、また、リスクをミニマム化するかを考えます。しかし、その一方で、「リスクのないプロジェクトには価値がない」ともいわれるように、リスクを積極的に取り、そのリスクを適切にコントロールするためにも、リスク・マネジメントが重要になっています。

プロジェクト・マネジメントにおいては、ＰＭＢＯＫにおける９つの知識エリアの１つとして、リスク・マネジメントが位置づけられています。プロジェクト開始にあたって、「リスク・マネジメント計画」を策定し、その計画にしたがって、「リスク識別」をし、「定性的リスク分析」「定量的リスク分析」を行った上で、優先度の高いリスクに対して「リスク対応計画」を策定し、対応・実施します。プロジェクトの全プロセスにおいて把握されるリスク情報は「リスク管理簿」に一元的に集約・管理されます。そして、「リスクの監視のコントロール」において、リスク・マネジメント計画から対応実施までのプロセスが適切に回っているかを確認し、必要があれば是正措置をとります。

このリスク・マネジメントのプロセスを構築することは必要なことですが、他のマネジメント領域と同様に、トップダウンの指揮・命令だけでは上手く機能しません。上手く機能させるためには、プロジェクト・メンバー一人一人のリスクマインドを向上させ、把握したリスクを情報共有できる組織風土をつくる必要があります。

プロジェクト・メンバーのリスクマインドの向上を考える上で、一番重要な鍵は「当事者意識」にある、と思っています。この「当事者意識」を考えるにあたって、社会心理学における実験、ダーリーとラタネによる「傍観者問題」に関するテーマで行った一連の実験が非常に興味深く示唆されるところが多いため、紹介します。

この実験に先立つ、１９６４年３月１３日午前３時、ニューヨークで２８歳の女性が自宅前で暴漢に襲われ殺害されるという「キティ・ジェノヴィーズ事件」が発生します。この事件が衝撃的だったのは、住宅街の真ん中での惨劇に対して、女性の叫び声を聞いた３８名の人々が自宅から事件を目撃していたにもかかわらず、誰一人、警察に通報せず、また助けにも出なかったということでした。当時の世論は当初、この３８名を告発するという論調だったようですが、ダーリーとラタネの二人は、「多くの人が気づいたからこそ、誰も行動を起こさなかったのではないか」と考え、実験を行いました。
大きく２つの実験がニューヨーク大学の学生を対象になされたのですが、１つ目の実験では、学生生活への適応調査が目的だとして、複数の学生がそれぞれマイクだけがある個室に入れられます。このマイクを通して、学生が順番に学生生活の問題点をスピーチするのですが、途中で、一人の学生がてんかんの持病があると言った後、てんかん発作が始まります。発作の様子は、スピーカーを通じて６分間続くという設定でした。学生の人数を変えて実験をした結果、被験者一人しか聞いていない時には８５％の確率

で、学生の救出に向かうにもかかわらず、自分の他に４名の学生がいる場合、行動を起こしたのは３１％のみでした。
　２つ目の実験は、換気口のある部屋に偽学生を複数人用意し、そこに被験者の学生を加えます。彼らに学生生活に対するアンケートを書かせている最中に、換気口から目に見える煙を流し出し、部屋は人の顔がかすみセキが出るほどになります。偽学生は落ち着いてアンケートを書き続けます。煙が出始めた後、被験者一人のみのときは７５％が通報するにもかかわらず、グループでいる時は３８％しか行動を起こしませんでした。

　この２つの実験を通してわかることは、人は誰でも「冷淡な傍観者」になる可能性があること。そして、その理由は、
「責任の分散」・・「私がやらなくても誰かがやるだろう」
「評価懸念」・・「自分だけの早とちり？」「自分だけが行動して何でもなかった時に恥をかくことを気にする」
「多数の無知」・・「他の人も行動しないのだから必要ないのでは？」
という３つにある、と評されています。

　また、この実験結果の恐ろしいところは、人はたとえ自分が被害者となっていても、その時点では、その事実に気がつかないことがあるという点だと思います。

　プロジェクトにおいても、納期遅延や品質問題を生じてバーストする過程で同様なことが起こっています。プロジェクトのＱＣＤを揺るがす事件が、多数のプロジェクト・メンバーの目の前で起こっているにもかかわらず、「そんなに大切なことなら誰かが何かをするはずだ」という「冷淡な傍観者」としての判断の積み重ねがバーストを招いていることも一因にあると思います。

　こうした事態を防ぐにはどうすればよいか。実験を通した教訓として、ダーリーとラタネの処方箋はこうでした。
　１．人を救いたいならば、まず危険を察知すべし
　２．それは助けが必要な出来事なのだと考えるべし
　３．自分には責任があると考えるべし
　４．取るべき行動を決めるべし
　５．そして行動に移すべし

　「煙の実験」のフィルムを見て、このステップを学んだ学生は、学ばなかった学生より、人を助ける確率が高まるという結果が出たように、リスクマインドを高めるためには、まずこういう事実があることを認識し、「当事者意識」を持つことの大切さから始めるべきでは、と考えます。
　また、組織の側からは、「評価懸念」という壁、言いだしっぺが損をするという壁を破ること。他のメンバーに「協力すると損」という価値観から「協力しないと損」という信頼ベース・正直ベースの文化へ転換を図ることが必要になります。
　「問題はあること自体が問題ではなく、解決されていっていないことが問題」であると捉え、リスクを早期に発見すること、発見するしくみを作ること、発見したらすぐに解決にとりかかる組織風土を作ること、そして、再発しないしかけを考え、作ることが、マネージャ層の役割になります。

こうした、個人面、組織面の両方が上手く機能することにより、リスクマインドを高めることによって、リスク・マネジメントのプロセスが十分に機能しはじめます。

（参考文献）ローレン・スレイター「心は実験できるか　２０世紀心理学実験物語」

連載　プロマネの現場から
第８回　タイム・マネジメントとコミットメント

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

ここ数年、システム開発の現場においても他の職場と同様に、深夜残業・休日出勤の原則禁止や早帰りデーの設定等という時短の職場が増えてきています。その一方、プロジェクトの立上げ局面や、バースト気味のプロジェクトにおいては、「原則」の規定を越えて、依然として勤務時間のピークが続くことが往々にしてあります。特に、プロジェクトの立上げ局面においては、立上げ局面として求められる成果物の作成はもちろんですが、その後のプロジェクトの円滑な推進のための前準備である体制構築、標準策定、開発環境の構築などの段取りが必要となります。ところが、各事項について合意すべき対象である顧客や自社・協力会社などのステークホルダーの了解がなかなか得られず、時間のみが過ぎてしまうということが往々にしてあります。「時間切れ」は「負け！」と言い聞かせて叱咤激励するのですが、その頑張りどころのベースとなるのが、マイル・ストーンを踏まえたスケジュールであり、それに基づくタイム・マネジメントとなります。とりわけ、先の見通しが見えにくい段階こそ必要だと痛感しています。

そこで今回は、プロジェクト管理における様々な側面の中で、タイム・マネジメントについて考えてみたいと思います。

スケジュールやそれに基づく進捗管理を適切に行うためのプロジェクト・マネジメントの知識エリアとして、ＰＭＢＯＫにおいては、プロジェクト・タイム・マネジメントがあります。プロジェクト・タイム・マネジメントでは、プロジェクトを所定の時期に確実に完了させるために必要なプロセスとして、６つのプロセスを規定しています。

１．アクティビティ定義・・プロジェクトにおける成果物を生み出すのに必要なアクティビティを洗い出す
２．アクティビティ順序設定・・各アクティビティ間の相互依存関係を明確にし、文書化する
３．アクティビティ資源見積り・・各アクティビティ作業にどのような資源（リソース）がどの程度必要かを見積もる
４．アクティビティ所要期間見積り・・個々のアクティビティを完了するために必要な作業時間を見積もる
５．スケジュール作成・・アクティビティ定義、アクティビティ順序設定、及び資源に対する要求事項を分析し、プロジェクト・スケジュールを作成する
６．スケジュール・コントロール・・プロジェクト・スケジュールの変更をコントロール

対象とする成果物を作り出すための作業を、アクティビティとして洗い出します。最下層のアクティビティの粒度は、普通の人が１週間働けば、進捗が目に見えるようにするため、最大４日間以内とするように分解します。複数のアクティビティ間の作業順序を明確にし、クリティカル・パスやボトルネックを把握します。過去の実績値やプロトタイピングによる試行によって得た作業時間を踏まえて、アクティビティ毎の作業時間を見積もります。そして、スケジュールを作成します。表記法としては、ガント・チャートやマイルストーン・チャート、ネットワーク図等により記述します。

このスケジュールを基に、現場ベースでは、毎日、進捗把握を行い、遅れ・進みの状況をウォッチします。遅れのリカバリーに対して、プロジェクト・メンバーの増強やリ

スケジュールを必要とする場合、週次または月次ベースでの社内上長や顧客等へのエスカレーション等を図ります。

　このタイム・マネジメントの実効性をあげるための個人レベルの活動を２つ紹介したいと思います。

１つ目は、作業時間の「記録」をつけること。
　つい最近、ドラッカー氏の本を手にとった際、次の言葉が目に飛び込んできました。

「成果をあげる者は、仕事からスタートしない。時間からスタートする。計画からもスタートしない。何に時間がとられているかを明らかにすることからスタートする。」（＊１）

　作業時間の実績を把握することの重要性・・何にどれだけ時間を使っていることがわかって初めて、時間のリストラも、新しい業務にその時間を当てることもできるということだと思います。
　この作業時間の実績把握について、個人のプラクティスに着目して、ソフトウェア開発作業を行うための指針として、パーソナルソフトウェアプロセス（ＰＳＰ：Personal Software Process）（＊２）というものがあります。
　このＰＳＰ、元々は学生や新人を対象としているところがあるのですが、ソフトウェア技術者自身が、徹底して自分の実施した作業に関する作業時間の「記録」を取ることを求めている点が参考になります。
　当初の計画で設定した見積りの値は妥当だったのか否か。自分の作業効率が良かったのか、悪かったのか。悪かった場合、どこがまずかったのか・・等々、これらの分析や反省を行うためには「記録」が必要となります。また、作業にかかった時間とともに、「中断時間」を計測することも薦められています。作業と作業の間のロスや待ち時間がわかれば、作業の品質と効率を上げることができるようになります。
　時間をおいて、「記録」を眺めてみると、「忙しい」と思っていた状態が、段取りが不十分であったため対応が後手に回った「気ぜわしい」ことであることがわかること。
　また、「記録」の開始直後としばらく経過後では、自分自身の対象作業への習熟度合いが高まり、また段取り＝プロセスが良くなるため、品質も効率も良くなっていきます。そして、しばらくすると、生産性が安定するという過程を通して、タイム・マネジメントへの感度が上がっていくようになります。

　プロジェクト成熟度モデルにおいてレベル１未満の組織においては、実プロジェクト全体に適用することはちょっと無理があるように感じる方がいるかもしれませんが、作業工程毎にパイロット・チームのメンバーに意識して作業をしてもらうだけでも、効果的だと思います。
　というのも、以前初めてプロジェクトの進捗管理に、ＥＶＭ（Earned Value Management）を適用した際に、実績記録を取り、その結果を分析して驚いたことがあるからです。
　それは、各工程の計画段階において詳細にアクティビティをブレイクダウンしＷＢＳを策定していたと思っていたにもかかわらず、プロジェクトの在籍要員のトータルの作業時間を足し合わせたトータル所要作業時間と、ＷＢＳのアクティビティのトータル作業実績時間との間に、２割から３割以上の開きがあったことでした。

つまり、２割から３割以上の時間が、ＷＢＳに記載されておらず、プロジェクト・メンバーは、ＷＢＳにないアクティビティを一生懸命実行しているということになるのでした。
その主な原因は、
１．プロジェクトマネージャやチームリーダのマネジメント工数の抜け・漏れ
２．実作業者以外による成果物のレビューアーの作業工数の抜け・漏れ
３．他チームや顧客間の調整時間、また調整による待ち時間やロス
が、ＷＢＳとして管理されていないことでした。・・・そして、恥ずかしながら、これらはＥＶＭ以前の問題です。
ただ、この事実がわかったことにより、結果として、マネジメントのアクティビティやレビューのアクティビティのＷＢＳへの反映、待ち時間やロス等の発生事由の実績への記述とリカバリー対策のフォロー等のアクションへつなげる、という工夫につながりました。

２つ目は、コミットメントの必要性です。
コミットメントとは「約束、誓約、関与」・・目標に対する決意表明です。
責任を引き受けると、何か起こった時の対応が早くなります。人のせい、環境のせいにする習慣がついていると、それは誰かがやるはずだ・・と、対応がワンテンポ遅れ、即応していればボヤですんでいたことが火事になってしまうことも起こりえます。
プロジェクトの推進上においても、顧客や上司等のステークホルダーに対する様々な約束・・コミットメントが要求されます。
ＰＳＰにおいては、このコミットメントの中でも、ソフトウェア技術者自身が、自分に対して行うコミットメントが一番重要である、といいます。ソフトウェア技術者自身が、「何を行うか」「それが行われたことを判定するための基準」「誰が行うか」「いつまでに行うか」「見返りとして与えられる報酬またはその他の対価」「誰がこの報酬または対価を提供するか」を明確にした上で、スケジュール案やその代替案を考え、コミットするべき相手に対して正式に合意します。

プロジェクトにおいて、ソフトウェア技術者自身は様々なコミットメントをすることになります。コミットメントに合意し、責任を引き受けた上で、そのコミットメントを適切に管理するマインドを持つことで、プロジェクト・メンバー一人一人のタイム・マネジメントへの感度が上がっていくと思います。

（＊１）ピーター・ドラッカー「経営者の条件」
（＊２）ワッツ・ハンフリー「パーソナルソフトウェアプロセス入門」

連載　プロマネの現場から
第 9 回　不況下におけるプロジェクト・マインド

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

１２月に入ってから、ソニーがグローバルで１万６千名のリストラを発表し、製造業を中心に数万人単位の派遣社員の契約打ち切り、また自動車業界で世界一のトヨタが大幅減益し来年度が赤字となる見通し、さらには内定者のお断りをする企業が出るといった厳しいニュースを立て続けに目にすることが多くなりました。

システム投資そのものが企業戦略の一環であるため、当然ながら、好況・不況によって、コスト削減要求や新規プロジェクトの成立そのものが大きく左右されます。

各企業とも、年末から年初にかけて、来年度予算の枠取りや予算案の策定検討が始まっているところが多いかと思いますが、その予算の前提となる案件が不透明になってきています。今年度の来年度実施予定の案件が延期となり、また現在実施中の案件が中断となった、という話を聞くことが多くなりました。

このような状況下で、プロジェクトマネージャやエンジニアはどうすべきか。本稿では、ドラスティックな回答を与えることはできないのですが、不況下におけるプロジェクトとそこでのエンジニアのマインドについて少し考えてみたいと思います。

売上げが伸びない中での収益改善のために、コスト構造の見直し・コスト削減の要求が社内外から強くなってきています。売上の規模では、前年比１割程度の削減であっても、現場のマネージャへの要求は、あくまでも感覚としてですが・・５割以上の厳しい締め付けがきている、というのが実感です。もっとも、実際の数値目標に落とすと、５％～１０％程度になるかもしれないため、自分でもちょっと過剰反応気味と思いつつ。

ところで、なぜそう感じるかというと、大きく２つ理由があります。

１つは、ソフトウェア開発プロジェクトの場合、人件費がプロジェクトコストに占める割合が８割を越すケースもあり、下方硬直性のある単価の削減に直結する見直しが、要求する側のマネージャ側にも、まだまだ心理的な抵抗感があります。もう１つは、コストが固定費主体で、そのコスト構造が変わらない場合、仮に収益が売上の１０％であったとすると、極端の話、売上の１割減は、そのまま収益の１００％減となってしまう背に腹は変えられない事情があるためです。

したがって、コストの大幅見直し、コスト構造の再編ということに直結してきます。

一方、個別プロジェクトにおいては、景気の良し悪しに関係なく、プロジェクトの目標である対象システムを、所期の納期・品質・コストで完遂することが大切である、と思っている担当者が多いのではないかと思います。

また、既にプロジェクトチームの要員・体制は、ハード組織から、有期のプロジェクトに対応したマトリックス組織となっていて、プロジェクトに必要な増員メンバーは、外部のコンサルタントやパートナー各社、またオフショア先との連携によって推進する、変動費が高い体制になっているかもしれません。

生産性についても、ツールや標準化等の整備による向上がある一方、対象システムの複雑化･高度化により相殺されている面があるため、大幅な改善の手段が見えていない。つまり、個別のプロジェクトにおいては、既に実施できるコスト削減策や生産性向上に取り組んでおり、プロジェクト目標である品質やスコープが切り下がらないかぎり、ドラスティックな改善が厳しい状況にあるのではないかと思います。

新規案件や戦略案件等が絞られる不況下におけるプロジェクトにおいて最も強いのは、保守・リテンション領域であると考えています。企業側としても、事業基盤のインフラとしてのＩＴシステムを、継続して維持・運営し、将来にわたっての戦略的な展開を行うためには、保守・リテンションは必須です。ただし、どの企業においても、潤沢に保守・リテンションの予算枠があるわけではないため、最低限必要な工数の議論をすることになります。

最低保守工数、限界保守工数という言い方は、一般化した言葉ではありませんが、保守枠の議論をする際に、使うことがあります。明確な定義があるわけではありませんが、最低保守工数は、ある領域を保守し、今後のレベルアップ案件に対応するためのシステム知見・業務知見を担保するために確保が必要な最低要員の数になります。また、限界保守工数は、ある領域の保守範囲の、定常稼動の監視等に必要な最低要員のみを残す工数であり、当該領域におけるレベルアップ案件や新規案件がきた場合に、そもそもの見積り等を含め対応できるかどうかをベンダーとして保証できない工数になります。
そして、この最低保守工数と限界保守工数との間が、プロセス改善や生産性向上、継続的なシステム運用の見直しの源泉になっていると考えています。

保守・リテンション領域は、システム知見・業務知見のベースカーゴであり、また、それゆえ、保守・リテンション領域は営業の最前線であると思います。いったん保守の領域を失うと、新しい提案やシステム構築の際に、既存の保守・リテンションのメンバーが行ったであればできたであろう負荷の２～３倍以上の負荷を要することがあるし、また、そもそも提案そのものができないという機会損失につながります。

そこで、豊富なシステム知見・業務知見を有する保守・リテンション領域の位置づけを再認識した上で、既得権益に甘んじることなく、もう一段二段の努力をする価値があります。

ところで、不況下のプロジェクトを考えるとき、１０年前の教訓を思い出します。バブル崩壊後の平成５年頃から数年間、新人の採用の大幅抑制またはゼロとした企業もありました。

その後の１０年を振り返ったいま、仕事の少ない時期でも、無理をして採用した新人がその後大きく成長し活躍しています。その結果、新人を獲得した部署は組織として大きく広がり、苦労して教えた先輩社員は十分に報われたと思います。苦しいときに逃げずに耐え投資したところこそ、花開いたのだと思っています。

だからこそ、会社側・マネージャのマインドとしては、不景気こそ人を採れ、良い人を採れ！　足元の社員・メンバーを抱えることさえ困難になっていることを認識しつつ、大手ではない中堅以下の企業こそ、本来そうあるべきなのだと思います。

一方、エンジニア側は、担当業務や部署の違いはあるかと思いますが、少数精鋭のマインドを持つべきでは、と思っています。少数精鋭とは、精鋭だから少数でよい・・の意ではなく、少数だからこそ精鋭化すると解釈すべきでは、と考えています。作業負荷のバランスはマネージャの役割として丁寧にフォローする必要はありますが、担当者に対しては、これまで複数のメンバーで分担していた領域やタスクをトータルに見ることのできるチャンスが得られるという意識付けをもたせることも大切だと思います。

万一、不稼動となった場合でも、新しい技術の習得とともに、これまでのプロジェクトでのさまざまな体験を、システム知識面・業務知識面・プロジェクト管理面等のノウハウとすること。組織としての教訓として蓄積する努力が必要だと思います。たとえば、既にマーケティング部門やソリューション企画部門等が主体で検討しているかもしれませんが、ＳａａＳやｘＳＰ化への知恵出しを行うこともできるのではと思っています。

また、現場ベースのみではコスト構造の見直しが困難であるといいましたが、企業として、オフショア先を含めた戦略パートナーとの分担の見直し、ソフトウェア開発プロセスにおける従来の委託範囲を超えた切り出し範囲の見直しについて取り組み始める必要があります。

新規ソリューションやサービスの充実・提供とともに、既存の保守・リテンションを核としたシステム知見・業務知見の強みを活かしたプロジェクト及び組織体制・構造の高度化への取り組みを継続する必要があります。

連載　プロマネの現場から
第 10 回　変化の必要性と２つのチェンジ

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

昨年１年間繰り広げられたアメリカの大統領選挙を通じて最も印象に残ったのは、やはり新大統領となったバラク・オバマの言葉でした。

それは、「Change！ チェンジ（変化しよう）」と「Yes, we can！イエス・ウィ・キャン（私たちにはできる）」の２つの言葉。「チェンジ」とは、直接的には、８年続いた共和党政権からの交代、原油など資源を求める世界戦略の見直し、イラク駐留からの撤退、サブプライム問題を生んだ金融システムへの規制・・等々の変化を求めたのでしょうが、その根本には、１００年に一度の不況に対応するために、未曾有の危機を脱しようという決意があり、国民に対して変化する覚悟を訴えたこと、また、その変化への対応が「イエス・ウィ・キャン」私たちにはできる、と繰り返し繰り返し訴え、それが伝わった選挙結果だったと思っています。

また、このオバマの演説を聴きながら、国は異なれども、ＩＴ業界等という大上段に構えずとも、個人やチームとして、「チェンジ」することを恐れない覚悟を持つことを鼓舞されました。

一方、ＩＴ業界がドッグイヤーと呼ばれて久しいですが、ジャック・アタリ「２１世紀の歴史」（＊１）の未来予測によると、２１世紀における技術革新のスピードと社会の変化のスピードもそれに負けていないことがわかります。

「技術革新のペースは加速する。食品や衣服の開発から生産・商品化されるまでのサイクルは、１ヶ月から４日に短縮される。自動車や家電製品のサイクルは、すでに５年から２年にまで短縮されたが、まもなく６ヶ月になる。同じことが薬品では、７年から４年になる。ブランドの寿命もますます短くなる。

よって、すでに強固な経営基盤をもち、グローバル化した企業だけが、めまぐるしく変化する経営環境のもとで生き残ることになる。・・」

また、この変化のスピードを踏まえて、知識習得のスピードも加速する、といいます。

「・・現在よりもさらに知識の獲得競争が主要な活動となり、企業はさらなるイノベーション競争をうながされ、そうした競争によって常に検証されることになる。

最新の職業知識が非常に重要となり、人々は自らの＜就労可能性＞を維持するための勉強を強いられ続ける」

「・・必要な知識は、すでに７年ごとに倍増しているが、２０３０年には７２日ごとに倍増する。知識を得て学び、＜就労可能＞な状態であり続けるために必要な時間も同様に増加する。」

７２日ごとに倍増・・の信憑性がわかりませんが、自分自身のチェンジなくして変化への対応はできません。

ところで、「チェンジ」という言葉については、昨年１２月にお亡くなりになった加藤周一さんの追悼番組（＊２）の中で、４０年前の１９６８年にも同じような場面があったことが語られていました。
それは、フランスのパリの５月革命の標語であった「Changer la vie! (暮らしを変えよう)」という言葉であり、この当時の精神を表していた、と。
同時期に、チェコスロバキアに起こった自由を求める市民たちによる「プラハの春」は、
ソ連軍の圧倒的な戦車部隊によって破壊されたのですが、その時の様子を加藤さんはこう表現しています。
「１９６８年の夏、小雨に濡れたプラハの街頭に相対していたのは、圧倒的で無力な戦車と、無力で圧倒的な言葉であった」。そして、戦車＝軍事力が徹底して言葉を弾圧したのは、言葉を軽んじていたからではなく、言葉が圧倒的であることを知っていたからである、と。

１９６８年当時は、圧倒的な軍事力と自由を求める圧倒的な言葉との対立でした。
そして、今回のサブプライムローン問題に端を発する世界金融危機とその結果の不況は、世界が一体となった圧倒的な金融資本及びそれを支えた金融システムと、個人・企業、さらにいえば国家との対立と見ることができるのでは、と思います。そして、現在のＩＴ技術やＩＴをベースとしたシステムは、意図せずして、前者に対して圧倒的に寄与してしまったかのようにみえます。そうであるのであれば、今後は、ＩＴ技術やＩＴシステムの使い方によって、社会をより良くするために後者の個人・企業・国家に対してより寄与をすることが必要である、と願望も込めて思います。

年初にあたって、ＳＥの役割は「社会システムの変革の担い手である」といわれることを改めて思い、その矜持を持って足元の業務に取り組みたいと思います。

（＊１）ジャック・アタリ「21 世紀の歴史――未来の人類から見た世界」
（＊２）ＮＨＫ　ＥＴＶ特集「加藤周一　１９６８年を語る～“ 言葉と戦車 ”ふたたび～」
２００８年１２月１４日放映
加藤周一「言葉と戦車」１９６９年刊

連載　プロマネの現場から
第 11 回　システムエンジニア観・再考　～３Ｋから新３Ｋへ

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

昨年来の景気の波の落ち込みは依然継続中・・景気の「気」は、多分に「気分」を映しているといわれますが、こういう時節こそ、目線を上にあげねばと思う日々です。

気分を左右するものとして、ＩＴ技術者に対してなされる３Ｋという「きつい、帰れない、給料が安い」というような揶揄された言葉や、さらには、「化粧ののりが悪い」なんて悪乗りも含んだ４２Ｋ（しにく・・死肉）という語呂合わせだか言葉遊びのような言葉があります。こういった物言いを聞くたびに思うのは、事実認識の妥当性に疑問を感じるとともに、言霊というものの存在の有無とは別に、予言の自己成就の作用を考えれば、状況は何も良くならないのではと思っています。

４２Ｋの中には、「気が休まらない」「キリがない」というソフトウェア開発の特性にあたるようなものもあり一理あると思ったりするものの・・その結果が「過労死」までつながっているのを見ると、そこにはマネジメントの姿がみえず、憮然とします。また、プロジェクト・メンバーが全員、全くの受身で、何も準備せず工夫もなしに、大規模化・複雑化・高度化しているプロジェクトに臨むのであれば、プロジェクトがバーストし、３Ｋのような状況が頻発するのも、ある意味、必然ではないか、と思います。

そこで、今回は、３Ｋなどという言い方を拒否することが大切であること、そして、それに変わる新しい言葉・・新３Ｋを考えてみたいと思います。

アメリカの心理学者ウィリアム・ジェームズの言葉に、
「意識が変われば行動が変わる。
行動が変われば習慣が変わる。
習慣が変われば人格が変わる。
人格が変われば運命が変わる。」
というものがあります。同意の表現として、ヒンズーの教えやフレデリィック・アミエル等いくつかあるようですが、そのことがかえって、人生における真実を表しているからなのでは、と思っています。

意識が、行動をつくり、
行動が、習慣をつくり、
習慣が、人格をつくり、
人格が、運命を決する

それでは、「意識」は何がつくるのか？

これについては、自己暗示について研究したエミール・クーエの法則、

「意志力と想像力（イメージ）が相反した場合は、常に想像力（イメージ）が勝つ」

という、「努力逆転の法則」がヒントになるのではと考えています。

目の前に、幅が６０ｃｍ、長さが５ｍの木の板があったとして、これが床の上にそのままおいてある場合、板の上をやすやすと歩いて渡ることができます。しかし、この板が、空中２０メートルの建物の間に橋渡しされている場合は、万一足を踏み外したらど

うしよう、突風が吹いたらどうしよう、と想像し始めると足がすくんでしまい、そこを無理強いして歩いても足を踏み外してしまうことになる、ということを示しています。

　このことからは、

　イメージが、意識をつくる

ということが言えるのではと思います。

　それでは、さらに、「イメージ」は何がつくるのか？

　最近のＮＬＰ（神経言語プログラミング）等でわかるのは、冒頭にいった言霊・・ではありませんが、使う「言葉」によって、私たちのイメージや意識が大きく左右される、ということでした。

　つまり、言葉が、イメージをつくる、と。

　以上をまとめてみると、

　言葉が、イメージをつくり
　イメージが、意識をつくり
　意識が、行動をつくり、
　行動が、習慣をつくり、
　習慣が、人格をつくり、
　人格が、運命を決する

　だから、否定的な言葉はできるだけ使わず、常日頃から肯定的・良い言葉を使うべきではないか、と思っています。

　そこで、システムエンジニアにとっての必要とされる言葉・良い言葉探しをしてみました。新３Ｋの候補ということで、Ｋで始まる言葉を少し挙げてみると・・

　好奇心、興味、教育、考える、概念・コンセプチュアル、仮説、行動、感性、開発、改善、改革、工夫、向上、解決、開花、果敢、決断、決定、コントロール、協力、協業、協創・コラボレート、協働、共同、交流、コミュニケーション、カバー、矜持、貢献、快適、快感、快挙、格好いい、稼ぐ、希望、期待、感謝　・・・

　ところで、これらの中から、システムエンジニアにとっての相応しい言葉を選ぶにあたって、ロシアの研究者・A.P.Erchov の「プログラミングに必要な才能」（＊）についての要件を振り返ってみます。

・第一級の数学者の論理性
・エジソンのような工学の才能
・銀行員の正確さ
・推理作家の発想力

・ビジネスマンの実務性
・協同作業をいとわず経営的な関心も理解する性向

当然ながら、こんなスーパーマンみたいな人はいないのですが、そもそもプログラミングには、こういう能力を要求されているのだ、ということを示しています。こういう難度の高い仕事にいかに取り組むか、その取り組みそのものがソフトウェアエンジニアリング
であること、そこには様々な創意工夫があり、チャレンジがふんだんにある世界なのだ、といえます。
その上で、システムエンジニアにとっての３つのＫを選んでみると、

１．好奇心・興味
コンピュータエンジニアリング、コンピュータサイエンスから、プロジェクトマネジメントにわたる広範な知識と、プロジェクトが対象とする業務知識・ドメイン知識を修得し続けるマインドが必要とされます。
また、仕事を通してスキル・能力が向上することによって、より高度な仕事にチャレンジできるようになることが報酬になると思います。

２．改善・工夫
プロジェクトの成果物の設計・開発とともに、その成果物の設計・開発プロセスそのものの構築をすること。そして、個人レベルから組織レベルにわたる、その継続的な改善活動をすること。新規・先端的な技術面でのチャレンジや、様々な技術の、大規模・複雑化した実プロジェクトへの適用時においては現場での工夫が必須となります。

３．感謝
大規模・複雑化した実プロジェクトに立ち向かおうとする時、そのプロジェクトに必要なスキル・能力が非常に多岐にわたっていることに思いをいたすと、プロジェクトの推進が、ステークホルダーとしていかに沢山の人々や組織の協力や支援の上に成り立っているかに気づきます。協業・協働・協創を通して、プロジェクトの仲間や顧客とともに、成長していくこと。このことに対する感謝なくして、プロジェクトはないと思っています。

以上を踏まえて、システムエンジニアは、新３Ｋとして、常に「好奇心」「改善」「感謝」を持って行動指針とすべきではないか、と思います。

最後に、この新３Ｋが、ユーフェミズム（Euphemism）・・事実の隠蔽のための婉曲表現となっていないかどうか、プロジェクトマネージャは、胸に手を当てて考えてみる必要があります。現実にプロジェクトを苦痛に感じているメンバーがいるのであれば、そのことを真摯に受け止めること。その原因が、プロジェクトに必要な知識やスキル不足にあるのか、要員や体制の不足・不備にあるのか、プロセスが不十分であるのか、等々を明らかにした上で、組織的な取り組みを行うことになりますが、また別稿にて紹介できればと思います。

（＊）鶴保征城・駒谷昇一「ずっと受けたかったソフトウェアエンジニアリングの授業１」より引用

連載　プロマネの現場から
第 12 回　システム構築における「トリアージ」

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

３月も末、春を間近に迎え、数年前から国家的なリスクの１つとなっているパンデミックの危機ですが、今年は無事に回避することができたのでしょうか。

先日見た映画「感染列島」は、このパンデミックの世界を見事にシミュレートしています。２０１１年のお正月、パンデミックが日本を襲います。感染から一週間足らずで、数千人が亡くなります。ライフラインは破綻・・新種のウィルスはなかなか発見されず、発見された後も、感染経路は特定されない。さらに、感染源が分かってからも、治療法の確立までに時間がかかります。全国民の３分の２が罹患し、１千万単位の人が死に至る・・映画のため、WHOの予測よりも誇張されているきらいはありますが、万一実現した場合、悪夢といえばこれほどの悪夢はないかもしれません。

最近のセミナーのお知らせ等を見ていると、パンデミックを取り上げ、各企業においてのパンデミック対策のためのビジネス・コンティンジェンシー・マネジメントのあり方やそのマネジメント・プランを考えるものもあり、関心が高まっていることを示しています。

ところで、この映画「感染列島」の描く悲惨なパンデミックの世界の中で、特に印象に残ったのは、病院に大挙して押し寄せる患者に対して、圧倒的に少ない医師・看護師・医療機器・施設等のリソース・・この状況下で、誰を治療し、誰を治療せずにおくか、という「トリアージ」が実践される姿でした。

この「トリアージ」、映画鑑賞後ずっと心に引っかかっていたため、高橋章子編「救急看護師・救急救命士のためのトリアージ」（＊１）を手に取りました。

「トリアージ」という言葉の由来は、フランス語の trier（分別する、選り分ける）によります。そこから、「傷病者など治療を受ける必要のある人々の、診療や看護を受ける順番などを決定する診療前の一つの過程である」と定義されます。緊急度と重症度の判断基準を基に、限られた医療資源を最大限に活用して、傷病者の救命に最大効果を上げるための技術・システムである、といいます。

「トリアージ」を行う目的には３つあります。

（１）治療不要な軽症傷病者を除外すること
（２）救命不可能な患者に時間や医療資源を費やさないこと
（３）緊急性の高い患者を選別し、搬送・治療の優先順位を決めること

３つ目のトリアージの判定・優先順位づけの結果は、後工程の医師や看護師に伝えるため、トリアージタッグという色別の識別表によって表されます。

そして、トリアージタッグの色は、４つにわかれています。

赤　危機的状態で直ちに処置が必要
黄　数時間処置を遅らせても生命に影響なし、歩けない
緑　軽度外傷、歩行可能、通院加療で可能
黒　生命兆候がない

赤・黄・緑・黒・・の紛れのない４つの色分け。万一、自分の肉親が災害に遭い、その身体にどのトリアージタッグがつけられているか、を知る瞬間を想像するだけで、胸がつぶれる思いがします。

そして、このトリアージの判定とタッグづけは一度実施すれば終わりになるのではありません。

Ｓｔｅｐ１　現場のトリアージ
search & rescue により、災害現場の救助・救急隊員が行う
Ｓｔｅｐ２　救護所のトリアージ
急搬送により救命の見込める患者の識別、
緊急治療を要する患者の識別
Ｓｔｅｐ３　搬送のトリアージ
搬送前の再評価、１医療機関に集中しないための効果的分散化
Ｓｔｅｐ４　病院のトリアージ
各種制約下での根治的治療のためのトリアージ、
後方搬送に備えてのトリアージ
というように、各ステップ毎の状況と傷病者の容体変化を踏まえた上で、適切な「トリアージ」を行う必要がある、となっています。

それでは、システム構築においてはどうでしょうか？

「トリアージ」が行われる現場という点で連想されるのは、実際、救急・救命医療の現場と類似している、本番障害の現場であり、次に、バーストしたプロジェクト、それに対処するための「リスクマネジメント」だと思います。
そして、「リスクマネジメント」プロセスにおいても、定量的・定性的なリスク分析により、重要度をつけて、優先度の高いリスクに対して「リスク対応計画」を策定し、対応・実施することが必要とされています。
しかし、救命・救急現場における「トリアージ」の判定と「トリアージタッグ」による厳然たる識別を思うと、システム構築の現場が「トリアージ」から学ぶべきことは、徹底した「優先順位づけ」の大切さである、と思います。

システム構築における「優先順位づけ」を考えさせられる特徴的な例として、システム開発の要求事項抽出における「２４２３の法則」があります。
ベンダー側の立場からみた要求事項の数を見た場合、
・ＲＦＰ等の契約段階では、規模・金額は「２」の要求が出ているとします
・ところが、要求分析段階を進めるうちに顧客の要望は膨らみ、「４」となります
・当然、ベンダー側は追加費用を要求しますが、顧客側は「２」しか払わないといい、いったんは要求仕様の絞込み等喧々諤々の調整が行われます
・最終的に、顧客側は追加予算を組み、「３」の費用を支払うことになります。
顧客側は当初予算が「２」から「３」へ増えたことを不満に思い、ベンダー側は「３」の費用で「４」の作業量をさせたれたことに不満を持つというものです。
要求事項が「２４２３の法則」にしたがってしまうのは、
・顧客にとって、業界常識・自社の常識・業務の常識であり、説明するまでもなくわかっているはず、と他意なく説明をはしょってしまう暗黙知が、ベンダー側にとっての理解の壁になることがあること。
・また、業務知見やシステム知見のあるといわれる有識者が最初からすべてを提示できるわけではないこと。
・そして、前の２つの理由以上に、まだ上流工程であるため要求事項をすべて詳細に出す必要がないと顧客・ベンダー双方が思っていること
が、大きな理由であると思います。

「２４２３の法則」によって肥大化するのは、「機能」や「スコープ」だけではなく、「品質」「納期」「コスト」に対する要求も同様です。これらの要求は互いにトレードオフの関係にあるため、すべてが最優先のプロジェクトは失敗する・・いや、大失敗する、といわれますが、その通りだと思います。

そのため、システム構築の現場においても、「要求のトリアージ」という考え方が必要になる、と考えます。
実は、システム構築においての「トリアージ」の重要性について明快に指摘されている先人がいます。それはエドワード・ヨードン氏で、ヨードン氏曰く、
「チームがたった一つ言葉を覚えるとすれば、それは「トリアージ」だ」（＊２）、と。
「ソフトウェア・トリアージ」を行うためには、自分が構築しようとしているシステムの「目的」を明確にすること。優先すべきは、機能なのか？　品質なのか？　コストなのか？。また、どの機能を優先すべきなのか、を問え。そして、各々工程における選別のための基準を持つ必要がある、と。

ところで、優先順位づけの大切さは、誰しも理解していることだと思います。その一方で、それが様々なステークホルダーからの言い分を聞いているうちに、なかなか実現できないのも現場での悩みであると思います。

救急救命の現場の「トリアージ原則」においては、
　生命　は　四肢　に優先し、
　四肢　は　機能　に優先し、
　機能　は　美容　に優先する
つまり、
　生命　＞　四肢　＞　機能　＞　美容
の原則があること。極端な場合、トリアージ担当の医師や看護師らは気道確保と止血以外の医療は行わない、といわれます。この厳然たる優先順位づけを見て、正直驚くとともに、いままで、ここまでシビアな優先順位づけを考えたことがあったか、と自問自答すると、決してそうではないことに気づかされます。日頃から、トリアージの基準を整理し、実戦を通してたえず改善を進めていくことが大切である、と思います。

最後に、パンデミックの世界で私たちはどう生きるか？
映画のラストでの主人公の医師の答えはこうでした。

「たとえ明日、地球が滅びるとも、
今日君は、りんごの樹を植える」

システム構築の現場においても、かくありたいと思います。

（＊１）高橋章子編「救急看護師・救急救命士のためのトリアージ」（エマージェンシー・ケア　２００８年夏季増刊）
（＊２）エドワード・ヨードン「デスマーチ　第２版」