はじめに

昨年、『ITプロジェクトの「見える化」下流工程編』を発行してから、約1年が経過しました。この間に、同書はソフトウェア業界やソフトウェア研究者から大きな反響を得ることができました。

同書は、「現場のありのままを分かりやすく表した事例」としてまとめられており、「ソフトウェア開発現場での改善に役立つ」、「見えない現場を見えるようにするための秘訣がよく理解できた」などの賛同をいただいてきました。

一方、「下流工程も重要だが、上流工程をどのように見える化すべきかを教えてもらいたい」、「上流工程での品質をどのように測ったらよいのか」、「プロジェクトの成否を決定的にしてしまう上流工程を、どのような考え方で進めればよいか」など多くの意見が寄せられました。

そこで、『ITプロジェクトの「見える化」上流工程編』では、プロジェクトの成否を決定付けてしまう要因とはそもそも何であるのかを研究しました。そして、それらの要因を見えるようにするためには、何をどう測ればよいのか、その測定結果から見えたことをどのように判断すべきか、という実践的な視点から、プロジェクトの不調の要因をいち早く検出できるようにしました。不調の要因が分からず悩んでいるプロジェクト・マネージャが、的確な対策を早期に実施できるように構成しています。

ITプロジェクトにおいて、絶対に死守しなければならないコスト・納期・品質の3要素を十分把握したうえで、しっかりした計画を立案し、プロジェクトを強力に推進していくことが重要です。今後も絶え間なく発展していくソフトウェア開発技術への追従と、プロジェクト・マネジメントのプロセス改善が継続的に必要です。私たち、ソフトウェア・エンジニアリング・センター（SEC）および「プロジェクト見える化」部会は、ソフトウェア・エンジニアリングの発展に貢献していきたいと考えております。

これを契機に、本書と下流工程編がますます活用されていくことを期待しています。

2007年4月
プロジェクト見える化部会一同

ITプロジェクトの「見える化」 上流工程編

第1章

「見える化」の目標

1.1 ITプロジェクトの実情

近年、ビジネス環境が急激に変化し、それを支える情報システムの開発においても無理な納期・コストが要求される事態が発生している。

しかし、ITプロジェクトには「手戻り」が発生するリスクが大きく、その程度によってはプロジェクトに重大な危機をもたらす。そもそもITプロジェクトの推進主体は人間であるため、メンバー間の意志疎通にズレが生じたり、コミュニケーション不足や常識（文化）の違いからくる誤解・誤認などが生じたりしやすい。これらに起因して、常に手戻りが発生する可能性がある。

では、現実のITプロジェクトでは、こうした問題の本質を正しくとらえているだろうか。例えば要件定義の工程において、ユーザー/顧客から指摘される問題は、「業務内容を理解していない」、「提案がない」などプロジェクトを推進するシステム・エンジニアの未熟さに起因すると考えられることが多い。プロジェクトが抱えている問題の一面を示していることは間違いないが、それがすべてでないことを、読者の方々も察していることだろう。

もっと重要で本質的な問題は、上流工程、特に要件定義における「見える化」がどの程度実現できているか、という点にある。家の建築を建設会社に依頼するときのことを考えてみれば分かる。どのような家を建築するかは、顧客が建築模型や間取り図、サンプル（モデルルームや部品）など、具体的に見えるものに基づいて重要な判断をしている。

一方、情報システムを構築する場合はどうか。家の建築のように具体的に見えるものに基づいていないため、要求内容の認識にズレが生じやすく、厳密性に欠け、曖昧な仕様となってしまうケースが多い。

そのため、ITの専門家ではない顧客は、正しい仕様かどうかの判断が難しく、設計書の承認を遅らせたり、後でも

簡単に直せると誤解し、安易な判断をしたりする。その結果、後になって仕様変更が頻発することがしばしば起こる。

こうした問題を未然に防ぐために、本書はITプロジェクトの要件定義工程において、プロジェクトの「危機」の兆候を「見える化」するエンジニアリング的アプローチを提案する。

対象読者

本書が想定する読者は、①情報システム開発の受注側プロジェクト・マネージャおよびプロジェクト・メンバー、②受注側のPMO（Project Management Office）や品質管理部門の担当者である。

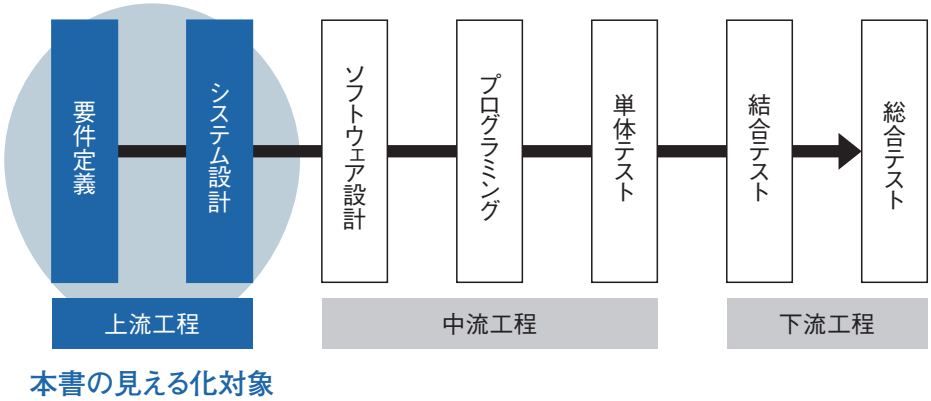
また、ITプロジェクトの上流工程では、情報システム開発の発注者もプロジェクトの成否に大きな影響を及ぼす。このため、発注側の責任者、PMO、品質管理部門も対象読者とする。

本書の狙い

本書の狙いは、ITプロジェクトに関与する人々に、「このような兆候が表れている場合は危険信号であり、後工程でプロジェクトが不調となる可能性がある」という“気付き”を与えることにある。そのため、上流工程のプロジェクトが直面している問題や実際の事例などを紹介しつつ、プロジェクトのステークホルダーに示唆を与えることも考慮している。

具体的には、「プロジェクト計画の内容がどのように実行されているかを問題とする」のであって、「プロジェクト計画書の書き方を説明する」のではない。特に、「本来はそうすべきかもしれ

図表1-1●本書の見える化の対象



ないが、現実的には様々な事情から、理想通りにはできない」というプロジェクト・マネジメントの現状を十分に考慮して、現場で実践的に使える「見える化」の方法を紹介する。

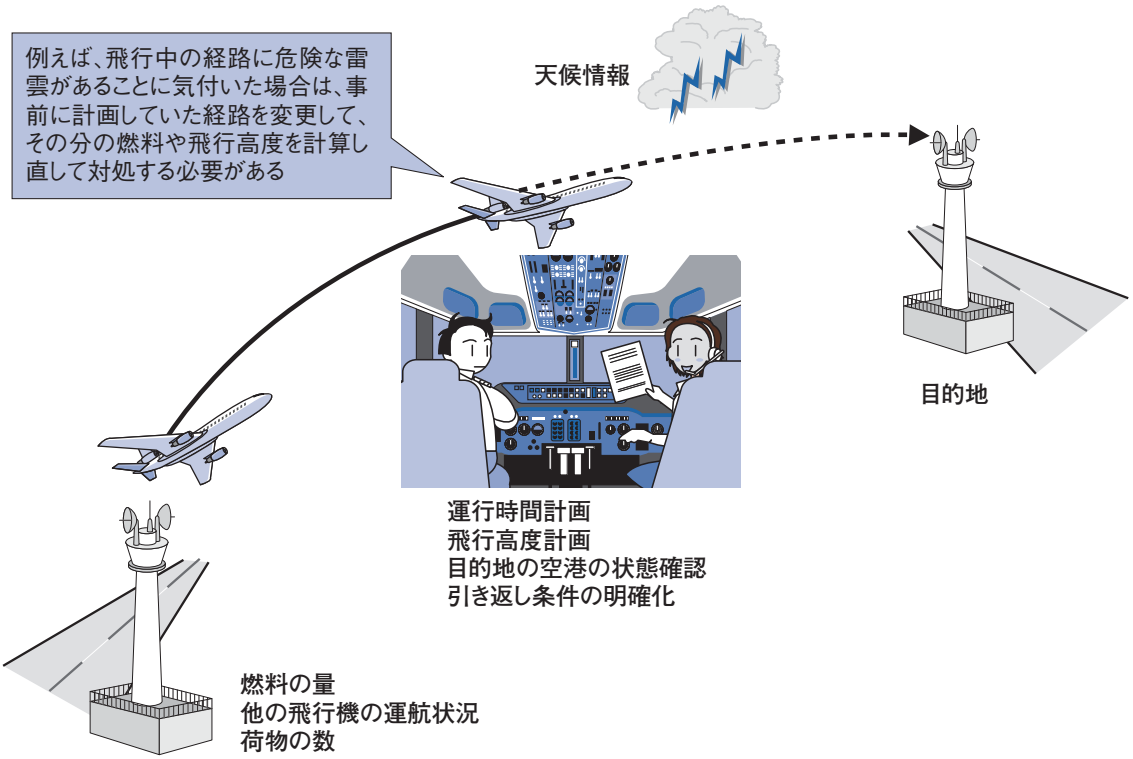
対象とする開発工程の範囲

本書が対象とするITプロジェクトの「上流工程」とは、**図表1-1**に示すように、要件定義工程からシステム設計工程までとする。結合テスト工程以降の見える化については、『ITプロジェクトの「見える化」下流工程編』を参照してもらいたい。

1.2 上流工程の「見える化」の目標

ITプロジェクトの上流工程では、要件の曖昧さ、納期までの切迫感の欠如、成果物の見えにくさなどにより、問題を内在させたまま工程を進めてしまう傾向がある。「計画との乖離」、「放置してはいけない問題」をつかまえるため、

図表1-2●フライトにおける目的地設定と飛行計画



まだ顕在化していない問題をとらえ、プロジェクトの状況を見えるようにすることが大切である。

上流工程での見える化の目標は、『どこに問題があるのかを、早期に発見すること』である。また、不確定な要素への対策を立てることを含め、『不確定な要素がいつまで不確定のままでよいのかを評価し、判断すること』である。

そのため、発注側と受注側がプロジェクトの目的について合意し、その合意に基づいた計画の中で、成果物の作成基準や変更基準を事前に決めておく必要がある。こうしておけば、プロジェクトの環境変化に対応することができる。

このことは、ITプロジェクトを旅客機のフライトに例えてみると分かりやすい（**図表1-2**）。旅客機のフライトでは、初めに顧客がどこに行きたいのかという目的地を設定するが、「アメリカに行きたい」とだけ言われても曖昧で、パイロットは困ってしまう。「アメリカの東海岸に」でも困る。「ニューヨークのJFK空港までだ」という明確な目的地がなければ、フライト計画が具体化しない。

目的地が具体的でなければ、ハワイまでの燃料しか積まずに出発して、途中で燃料不足に気付いたり、大量の燃料を積み過ぎて燃費が悪くなって費用がかさんだりする。運よくJFK空港に着いても、滑走路が空いていなくて着陸できなかったりすることもあるはずだ。

飛行中も、現在の旅客機の位置や高度、燃料、天候情報などの様々な計器や情報を見ながら、予定通りに飛んでいるのかを確認しつつ、目的地に向かう必要がある。もし、何かのトラブルで別の旅客機が近くを飛んでいる場合は、ニアミス防止のために飛行高度を変えたり、飛行高度を変えることによる燃料消費量の変化に合わせて飛行経路を変えたりしなければならない。

このように、常にフライトの状態についての時間的な遷移や、発生する可能性のある問題事象を想定しながら、目の前で次々に起こるトラブルに冷静に対処する必要がある。もちろん、旅客機を破損させてはいけないし、顧客を時間通りに目的地に連れていかなければならない。

ITプロジェクトの上流工程でも、最初に与えられた予算や開発範囲、体制や期間などの条件を基に、プロジェクトを進めるのか、中止するのかを判断すべき場面がある。ITプロジェクトでは当初想定していた問題や想定していなかった問題が噴出する場合もあるが、プロジェクト・マネージャはそのつど的確に判断し、計画を柔軟に変更して対処しなければならない。詳細は本書の第3章以降で解説する。

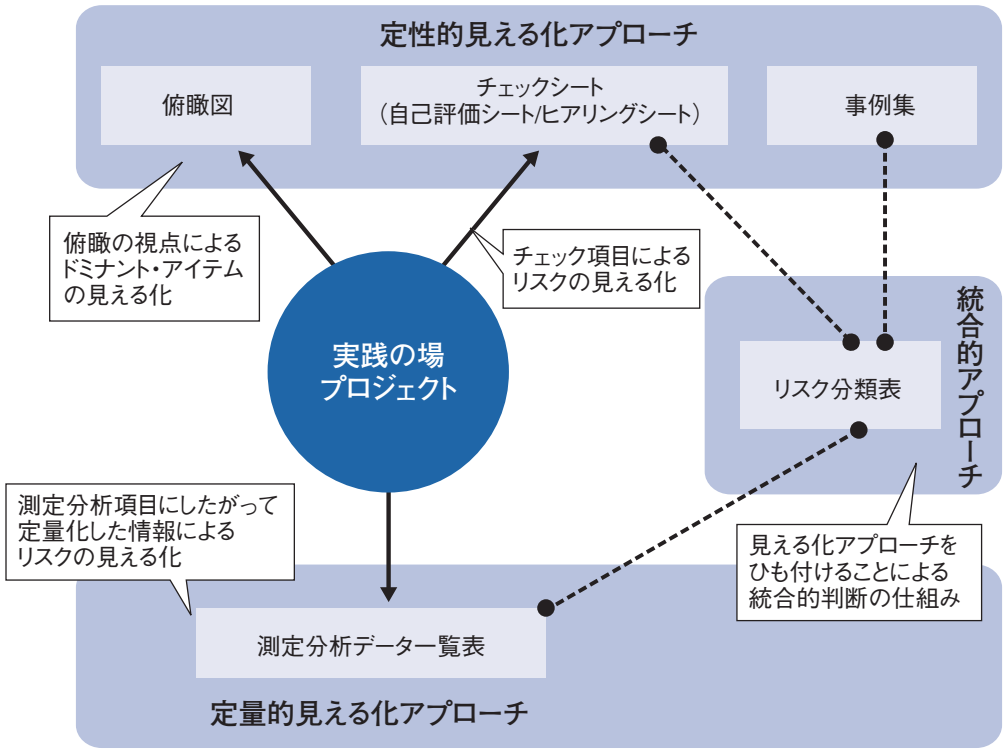
第2章

上流工程における「見える化」の全体像

2.1 3つのアプローチ

前述したように、ITプロジェクトの上流工程においては、目標と計画をきちんと立て、計画との差異を把握しながらプロジェクトを進めることが重要である。これらのリスク、計画との差異を把握するには、3つのアプローチが必要になる。

図表2-1●「見える化」方法の全体とツール



それは、①「定性的」見える化アプローチ、②「定量的」見える化アプローチ、そして③両者を統合的に見る「統合的」アプローチである。

「定性的」見える化アプローチ

定性的見える化アプローチとは、プロジェクトの当事者の目や外部の有識者の知見を通して、プロジェクトの状態を「定性的」に把握することを指す。

そのためのツールとして、プロジェクトを全体的にとらえるための「俯瞰図」、問題が潜んでいる可能性のある個所を特定するための「チェックシート」、過去のITプロジェクトで発生した上流工程の問題に関する「事例集」を用意し

図表2-2●「見える化」のツールとその概要

ツール・資料			説明
定性的見える化アプローチ	俯瞰図		ITプロジェクト・マネジメントにかかわる諸要素を俯瞰するための図。 ITプロジェクトの持つ本質的な問題（変化/変革による潜在的問題の多さ、ステークホルダーの多さ、加速度的な進化・変容を続けるIT技術・利用技術などの影響）を概観する。これを用いることにより、プロジェクト・マネジメント・プロセスで起こり得る問題を予測。「リスクの見える化」を図り、リスクを織り込んだマネジメントを展開できる
	チェックシート	プロジェクト・リスクの洗い出しに用いる。自己評価シートとヒアリングシートの2種類がある。プロジェクト・マネージャ自身の意識と、第三者がプロジェクト・マネージャにヒアリングした結果との差異に基づいて、プロジェクト・マネージャにリスクへの気付きを与える	
		自己評価シート	プロジェクトの状況について、プロジェクト・マネージャが自らチェックするためのスプレッドシート（Excel）。各チェック項目に3段階の評価を入力すると、レーダーチャートで結果を表示し、特に注意すべき項目を対策案とともに表示する
		ヒアリングシート	プロジェクトの状況について、外部の専門家がプロジェクト・マネージャにヒアリングをするときに利用するチェックシート（Excel）。各項目に5段階で評価を入力すると、レーダーチャートで結果を表示し、特に注意すべき項目を対策案とともに表示する。自己評価シートの結果と突き合わせれば、プロジェクト・マネージャの自己評価と乖離する点をレーダーチャートなどから見いだせる
	事例集		上流工程に起因するトラブルが発生したプロジェクトの事例をまとめたもの。こうした事例と同じミスを繰り返さないようにするための資料
定量的見える化アプローチ	測定分析データー覧表（および導出尺度一覧表）		プロジェクトのリスクを定量的に測定するための指標をまとめた一覧表。リスクの監視項目を知識エリアごとに整理した
統合的アプローチ	リスク分類表		ヒアリングシート、測定分析データー覧表、事例集をそれぞれ結びつけたもの。プロジェクトのリスクを確認する視点（分類）を軸にまとめた。これを用いることによって、ツールの横断的利用を可能にする

た（**図表2-1**）。上流工程を進めているプロジェクトは、これらのツールを使って問題点がどこにあるのかを把握できるようになる。

「定量的」見える化アプローチ

定量的見える化アプローチとは、文字通り定量的なデータにより、スコープの変化やプロジェクトの進捗状況、成果物の品質、コミュニケーションやモチベーションの状態などを見える化する手法である。

そのためのツールとして、「測定分析データ一覧表」を用意した。上流工程におけるプロジェクトの状態を数値で測定するための項目をまとめたものだ。「それらの数値を見れば、どのような判断ができるのか」ということが理解できる。プロジェクトの状態を定期的に測定する際に利用できる。

「統合的」アプローチ

統合的アプローチとは、定性的なアプローチと定量的なアプローチにより見える化した事象を基に、経験豊富なプロジェクト・マネージャが考えるのと同じようなやり方でプロジェクトの問題点を特定するための手法である。

そのためのツールとして、「リスク分類表」を用意した（**図表2-2**）。プロジェクトの問題点を洗い出すために分類した項目にしたがって、定性的・定量的な見える化ツール（チェックシート、測定分析データ一覧表、事例集）を関連付けたものだ。各ツールを統合的に利用することで、リスクを的確に洗い出せるようにしている。詳細は第5章を参照してもらいたい。

第3章

定性的見える化アプローチ

3.1 定性的アプローチの概要

上流工程における「見える化」は、まずプロジェクトの潜在的な問題を探ることから始める。どのようなツール（道具）で、問題を探ればよいのだろうか。

既刊の『ITプロジェクトの「見える化」下流工程編』では、既に問題が芽生えている（火を噴いている）という実態をプロジェクト・マネージャに気付かせ、実際どのような状態（症状）にあるか、症状からどのような対処策が見えてくるかという観点で「見える化」を考えてきた。

一方、上流工程では、大きな問題が顕在化することは多くない。たとえ顕在化しても、まだ上流工程であれば軌道修正する時間が残されている。

そこで本書では、まだ顕在化していない問題をプロジェクトの「リスク」ととらえ、プロジェクトの潜在的な問題を見える化することに重点を置いた。よくあるプロジェクトの不調や、放置しておくと悪くなる一方の現象を見える化できるようにする。そのための具体的なツールとして、プロジェクトを高い位置から見るための俯瞰図、チェックシート（自己評価シートとヒアリングシート）、事例集という3つの定性的見える化アプローチを採る。

俯瞰図

ITシステムを構築する際、プロジェクトに入り込んでしまうと、「木を見て森を見ず」という近視眼的な状態に陥り、結果的にシステム開発に不調を来たすことがある。プロジェクト・マネージャおよびメンバーが、客観的な視点で、「今自分たちは、どこにいて、どう動けばよいか」、「どこに問題が生じるか」を知ることが重要である。そのためには、プロジェクトの全体像を高い位置から眺めた俯瞰図の作成が役立つ。

自己評価シートとヒアリングシート

下流工程を見える化する際に開発した2種類のチェックシートは、上流工程

でも効果がある。プロジェクト・マネージャによる「自己評価シート」と、経験のある評価者による「ヒアリングシート」である。

要件定義を終え、基本設計に着手したあたりで、プロジェクトの実態を見ることが重要だ。ヒアリングに先立って、プロジェクト・マネージャに「プロジェクト概要」ドキュメントを記述させるのと同時に、自己評価シートを用いた自己チェックを実施してもらう。これらの資料を参考にしながら、評価者がヒアリングを行い、結果を分析する。

チェック項目は、実際にいくつかのプロジェクトで試験的に適用し、ブラシュアップを図ってきた。

事例集

上流工程の各ポイントにおける判断（見切り）の結果、様々なトラブルに至った事例を収集した。本書の付録資料として、58件のプロジェクト事例を紹介している。この事例集を読むことで問題プロジェクトの状況を追体験し、読者のプロジェクト運営に役立てることを期待するものである。

3.2 俯瞰図を使った「見える化」

ITプロジェクトにおける俯瞰図は、いわゆる「木を見て森を見ず」の弊害を排除し、ドミナント・アイテム（プロジェクトの目的達成を阻止する可能性のある本質的問題）を継続的かつシステム横断的にとらえながらマネジメントすることを可能とするツールである。

開発規模が大きい、ステークホルダーが多い、プロジェクトの構成が複雑であるなど、一目で全体像を把握することが難しい場合に、俯瞰図を作成することが望ましい。しかし、小規模であるとか、ステークホルダー数が少ないというプロジェクトでは、あえて俯瞰図を作成する必要はない。

俯瞰図の作成時期

俯瞰図は、プロジェクト発足時にプロジェクト計画書と同期して初版を作成する。計画書の一部として位置付けてもよいが、計画書を作成する前にプロジェクトの俯瞰図を作成して、プロジェクトの全体像をまず把握する方法が最も望ましい。

俯瞰図の修正

俯瞰図は、プロジェクト・マネージャがプロジェクトの状況を客観的に掌握して次に進むべき方向を指示する際のよりどころである。しかし、要件変更などが発生したときには、ただちに新たな俯瞰図を作成する必要がある。

基本的な俯瞰図

ITプロジェクトを俯瞰する際に用いる基本的な俯瞰図は6種類である（**図表3-1**）。プロジェクトの状況によって、必要であれば該当する俯瞰図だけを作成すればよい。ただし、俯瞰図の作成に慣れていない場合は、すべて作成することを勧める。

「ステークホルダー俯瞰図」は、ITプロジェクトに関するステークホルダーとその利害関係などを表現する。各ステークホルダー間の関係が不明確なときには、各ステークホルダーの立場に立って俯瞰図を作成していく。

「プロジェクト推進体制俯瞰図」は、プロジェクトの実施体制と構成する各組織のミッションを表現する。この俯瞰図によって各組織が担うべきミッションを確認し合い、プロジェクトをスムーズに運営できるようにする。

「周辺システム構成俯瞰図」は、開発システムがより大きなシステムの一部である場合に、開発対象システムと連動する他システムとの構造的な関係を表現する。

「システム構成俯瞰図」は、開発シス

図表3-1● 基本的な俯瞰図

ステークホルダー俯瞰図	●関係するステークホルダーの全体像
プロジェクト推進体制俯瞰図	●プロジェクト推進体制の全体像 ●ステークホルダー俯瞰図に含めることもある
周辺システム構成俯瞰図	●開発システムと連動する他システムとの関連 ●開発システムを含む、より大きなシステムの全体像
システム構成俯瞰図	●開発するシステムの全体像 ●外部ハードウェアとの接続関連
スケジュール俯瞰図	●プロジェクト全体のスケジュール ●クリティカル・パスのスケジュール
要員遷移俯瞰図	●キーパーソンのアサイン状況をフューズごとに示したもの

テムのハードウェア構成と入出力を中心とする構造を表現する。開発システムの性能や関連システムとの関係から、どのように復旧するかの要件を検討する際に役立つ。

「スケジュール俯瞰図」は、詳細なプロジェクト・スケジュールとは別に、クリティカルな作業のスケジュールだけを抽出して表現する。スケジュールの記述レベルは中程度の詳細さに留め、さらにその中で重要だと思われる上位7〜10程度に絞り込む。多すぎると管理できないし、重要なスケジュールがそれほど多いということはない。

「要員遷移俯瞰図」は、どこからどこまでの工程をどのキーパーソンが担当するのか、その遷移を表現する。時間の推移に伴う要員配置の妥当性をシミュレーションするのに役立つ。

どのような俯瞰図を作成すべきか

俯瞰図を作成するとき、どのような俯瞰図を、どのように作成すればよいか迷うことがある。だが、「なぜ俯瞰図を作成するのか」という原点に立ち返ることが必要である。

俯瞰図を作成する目的は、ドミナント・アイテムを見つけ出し、プロジェクトを成功に向けてマネジメントすることである。見栄えのよいものを作ることが目的ではない。

3.2.1　ステークホルダー俯瞰図

近年、複数の企業間での業務連携を前提とした情報システムの開発が増えている。こうした場合などは特に、多くのステークホルダーが関係し、複雑に利害関係が絡み合う。ここまで複雑でなくとも、ステークホルダーをいかにマネジメントするかは、プロジェクトの成否を左右する重要な要素である。

ステークホルダー俯瞰図は、基本的にプロジェクト・マネージャが作成する。まず、各ステークホルダーのキーパーソンとおぼしき人に当たりを付けて、話をすることから始める。結果として、その人がキーパーソンであることが分かるか、実は別の人がキーパーソンだったということが分かるだろう。ステークホルダー俯瞰図を作成する作業は、単に図を描くことだけではなく、ステークホルダーのキーパーソンを探すことであり、また、キーパーソンの考え方を知り、人間関係を構築する入り口でもある。

ステークホルダー俯瞰図の例を**図表3-2**に示す。ステークホルダー俯瞰図は、発注側の領域、受注側の領域に分け、協力会社が含まれる場合には、その会社も別領域にする。発注側の領域には、発注者の顧客も含める。システム運用開始時期を公表しているときには、その時期も併記しておく。

次に、ステークホルダーの各領域に、それぞれキーパーソンとなる人の名前を記入していく。単なる体制図を作るのが目的ではなく、それぞれの領域でのキーパーソンが誰なのかを理解することが重要である。

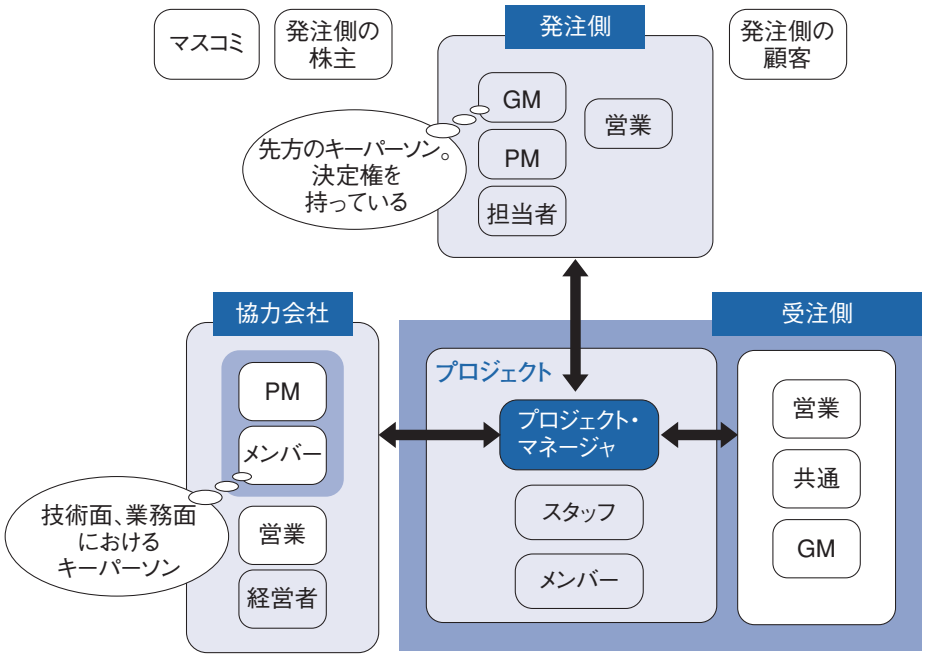
ステークホルダーの構成が単純な場合、ステークホルダー俯瞰図とプロジェクト推進体制俯瞰図を合体させることもある。だが、ステークホルダー俯瞰図は基本的に外部へ公表しない資料なので、使い勝手が悪くなる。

ステークホルダー俯瞰図をベースに、「コミュニケーション・マネジメント計画」や「会議体の設定」、「運営計画の作成」などに応用できる。

コミュニケーション・マネジメント計画を作成するときには、どのステークホルダーに、どのような情報をどのように伝え、情報共有するかを検討する必要がある。その際、ステークホルダー俯瞰図を使うと、コミュニケーション・マネジメント計画を適切に作成できる。

コミュニケーション・マネジメント計画で会議体を検討するときは、ステークホルダー俯瞰図から適切な参加メン

図表3-2●ステークホルダー俯瞰図の例



GM：ゼネラル・マネージャ　PM：プロジェクト・マネージャ

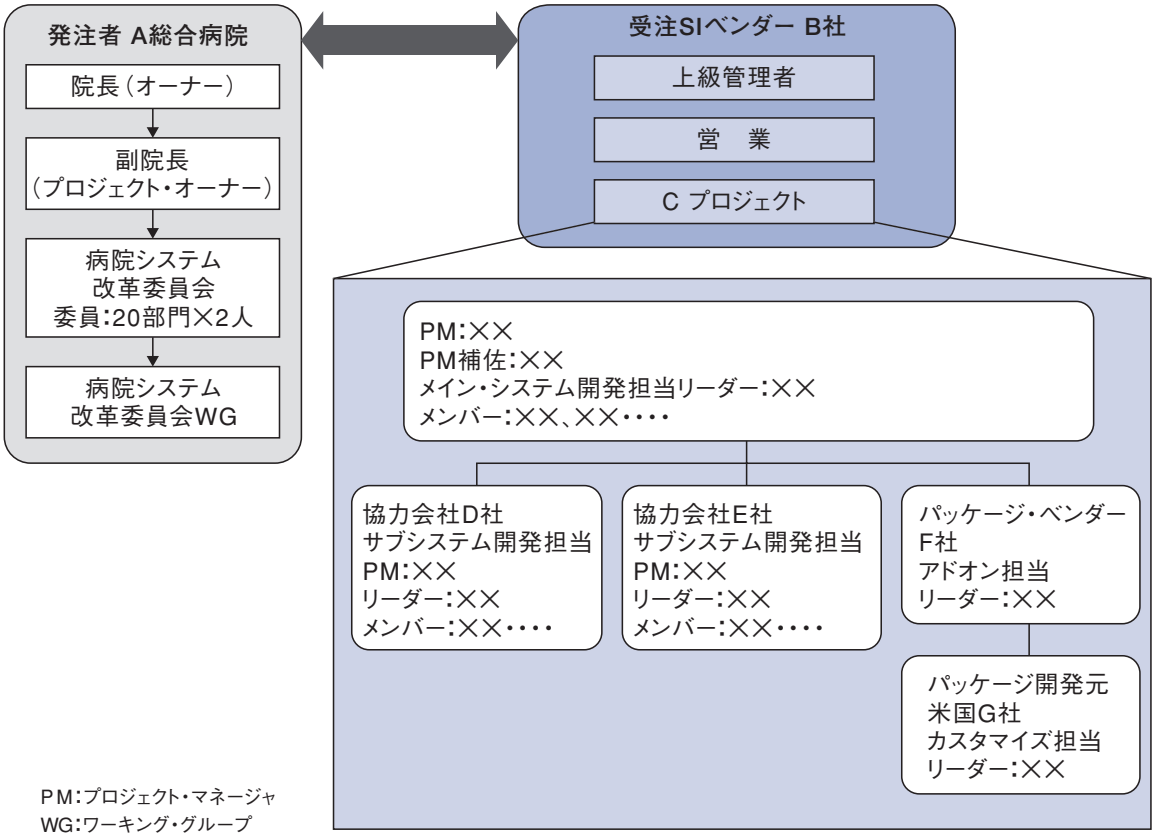
バーを選んだり、適切な運営計画を作成したりできる。ステークホルダーの体制やキーパーソンが変更された場合は、タイムリーにステークホルダー俯瞰図を更新する。

3.2.2　プロジェクト推進体制俯瞰図

一つの組織だけでプロジェクト推進体制が構成されることはまれで、通常は複数の組織によって構成されている。そのため、各組織体に対して果たすべきミッション、すなわち役割を明確に割り当てなければ、プロジェクト推進体制をうまく機能させることができない。

このことはプロジェクト・マネジメントにおける最も重要なポイントとなる。さらに、それぞれのミッションを有機的に連動させることこそが、プロジェクト・マネージャの最大のミッションでもある。

図表3-3●プロジェクト推進体制俯瞰図の例



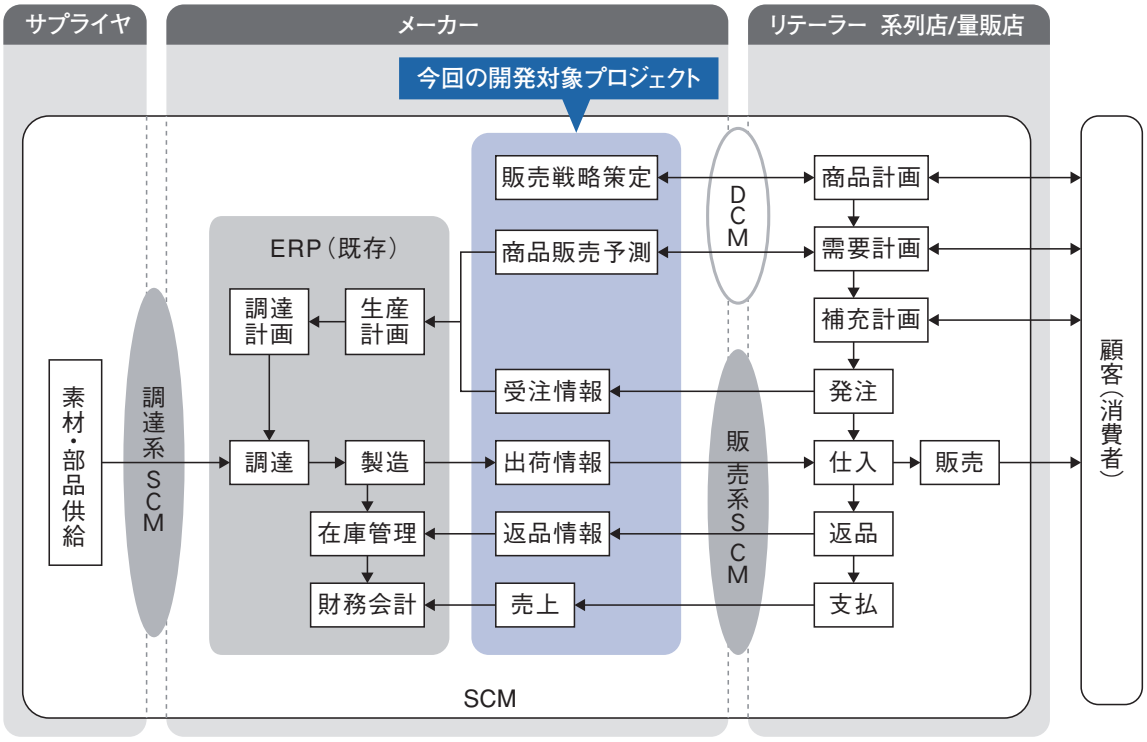
PM：プロジェクト・マネージャ
WG：ワーキング・グループ

図表3-3に示すプロジェクト推進体制俯瞰図は、プロジェクト・マネージャかプロジェクト・スタッフが、各ステークホルダーの役割を明確にしながら作成する。プロジェクト推進体制俯瞰図の目的は、各組織体のキーパーソンが、割り当てられたミッションを実施するのにふさわしい能力を持っているかを把握し、推進体制の強化策を立てることにある。

プロジェクト推進体制俯瞰図は、各組織体のスムーズな業務遂行のために、ミッションを明確にするものである。プロジェクトのキックオフ・ミーティングでは、プロジェクトにかかわるステークホルダーにミッションを認識してもらえるよう、プロジェクト推進体制俯瞰図を用いて説明する。

さらに、コミュニケーション・マネジメント計画の作成や、会議体設定、運営計画の作成時にステークホルダー俯瞰図を補完する資料として用いる。プロジェクト体制やキーパーソンが変更された場合は、速やかにプロジェクト推

図表3-4●周辺システム構成俯瞰図の例



DCM:デマンドチェーン・マネジメント　SCM:サプライチェーン・マネジメント

進体制俯瞰図を更新する。

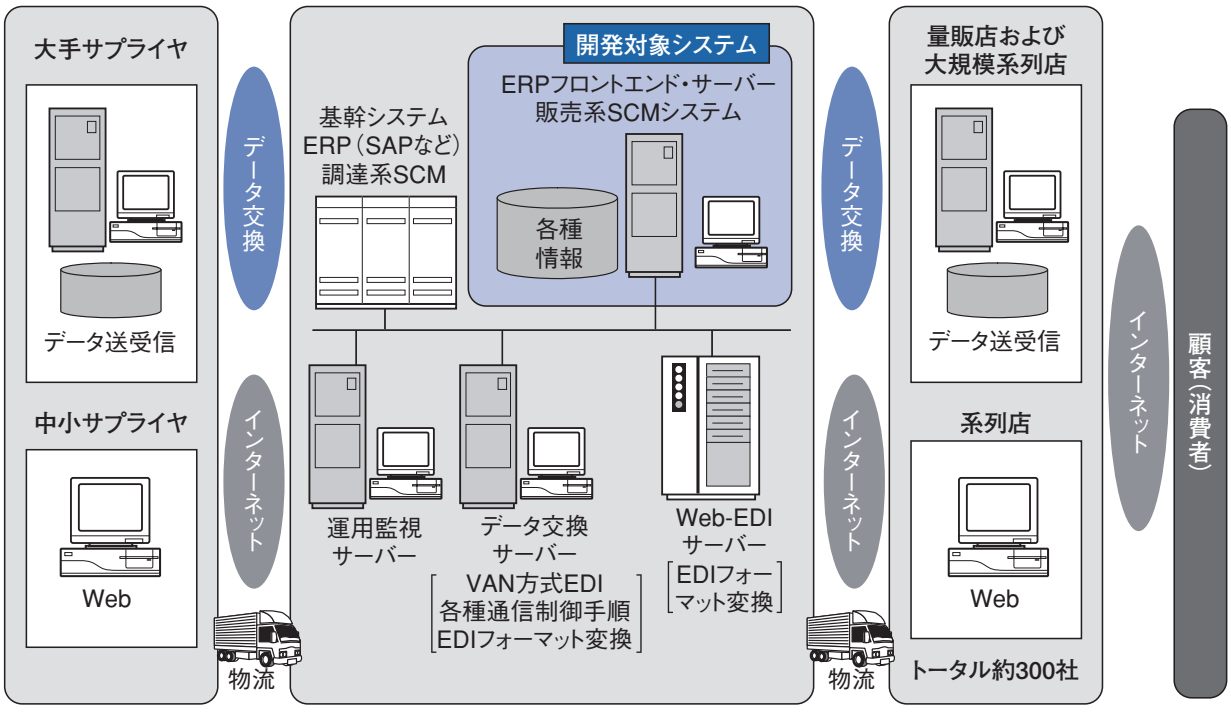
3.2.3　周辺システム構成俯瞰図

開発システムが単独で運用されるケースはまれになっており、他システムと連動して運用したり、システムを構成するサブシステムの一部として運用したりすることが多くなっている。

周辺システム構成俯瞰図（**図表3-4**）は、次の4つの目的のために作成する。①連動する他システムとの関係や位置付けについて検討する、②連動する他システムとの関係において、当該システムの性能がボトルネック(効果を発揮するうえでの障害）とならないようにする、③連動する他システムとのインターフェースを検討する、④総合テストおよび移行テスト計画を策定する、である。

周辺システム構成俯瞰図は、プロジェクト・マネージャかプロジェクト・スタッフが作成する。この俯瞰図はプロジェクト内部で使用することもできるが、主に発注側との打ち合わせで用いる。したがって、発注側が容易に理解

図表3-5●システム構成俯瞰図の例



ERP:統合業務パッケージ　SCM:サプライチェーン・マネジメント　EDI:電子データ交換　VAN:付加価値通信網

できる図にする配慮が必要である。

周辺システム構成俯瞰図を作成するとき、一次資料としてRFP（提案依頼書）を用いることができる。そこに、関連システムとの連動状況（データ総量、ピーク時のトランザクション量、運用時間など）、および開発システムの復旧要件などを発注者側に問い合わせて記入していく。

3.2.4　システム構成俯瞰図

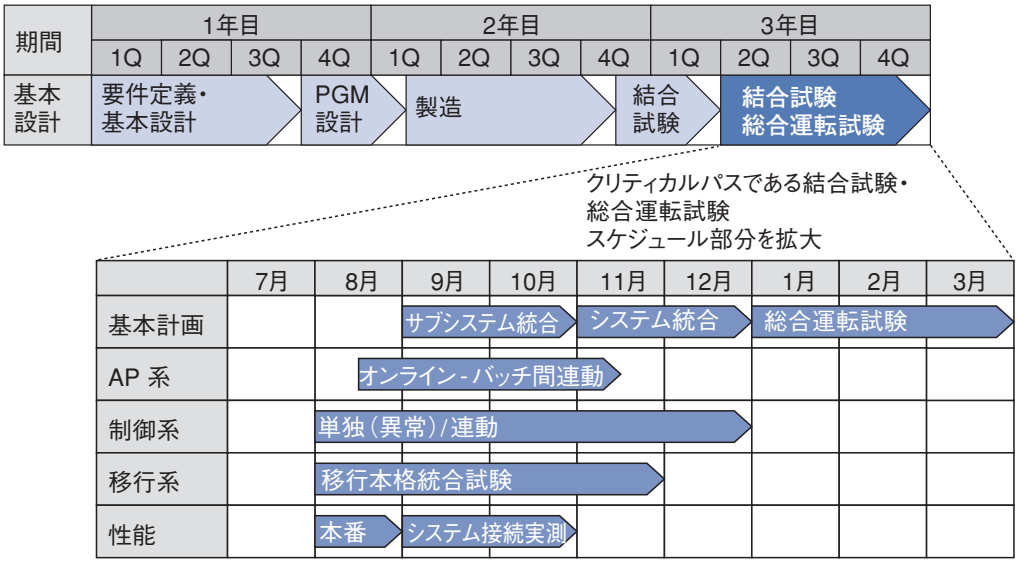
開発システムの性能要件および関連システムとの関係からどのように復旧するかの要件を検討する際、開発システムのハードウェア構成が必要となる。そのため、プロジェクトで作成するシステム構成図を基に、直接接続しているハードウェアを加えてシステム構成俯瞰図を作成する（**図表3-5**）。周辺システム構成俯瞰図はシステム間の連動について記述するのに対し、システム構成俯瞰図はハードウェア間の連動について記述する。

システム構成俯瞰図は、プロジェクト・マネージャかプロジェクト・スタッフが作成し、周辺システム構成俯瞰図と同時に用いる。

3.2.5　スケジュール俯瞰図

多くの詳細スケジュールがあるプロジェクトでは、プロジェクト・マネージャがそのすべてをマネジメントするのは

図表3-6●スケジュール俯瞰図の例



不可能であり、非効率でもある。1人が同時にマネジメントするスケジュールは、多くても10本以上にならないようにするのが現実的である。

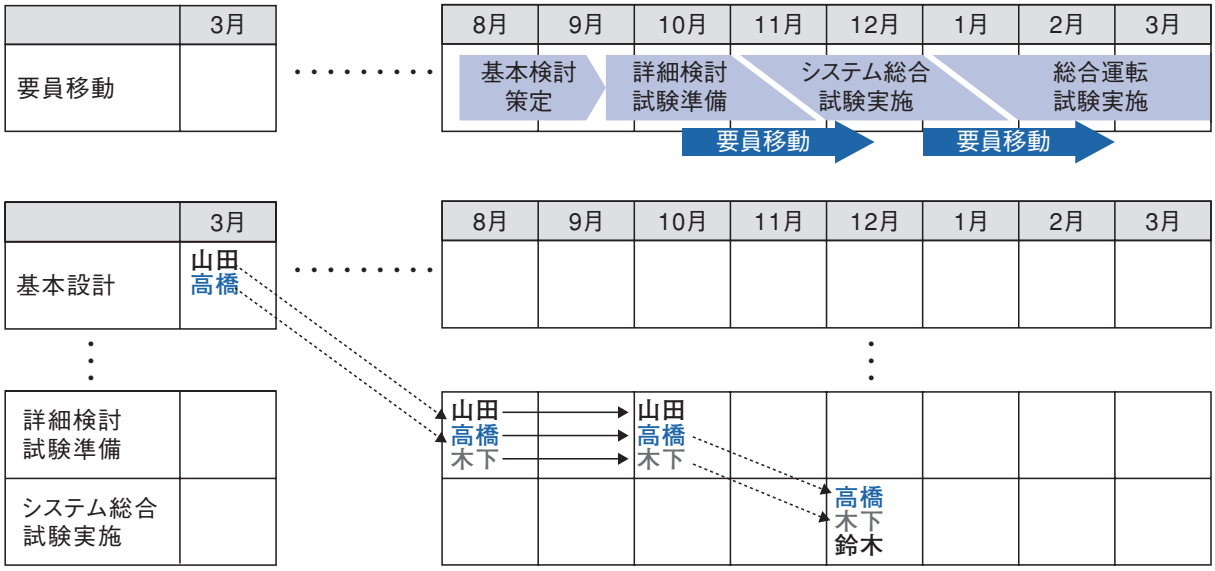
そのためプロジェクト・マネージャは、クリティカル・パス関連のスケジュールに絞ったスケジュール俯瞰図を作成し、重点的にマネジメントすべきだ(**図表3-6**)。例えば、構築期間が非常に長いプロジェクトにおいて、結合テストおよび総合テストこそがクリティカル・パスであると考えているとしよう。その場合、テスト工程のスケジュールは、さらに詳細まで落とし込んだ俯瞰図を書くことによって、スケジュール全体のドミナント・アイテムを俯瞰できるようになる。問題になりそうな部分をクローズアップすることが重要だ。

3.2.6　要員遷移俯瞰図

「要員（キーパーソン）がどの作業を、いつまでに終わらせて、次の作業に移るのか」。こうしたプロジェクト要員に関するマネジメントは、全体のスケジュール・マネジメントと同じぐらい重要である。

例えば、システム設計のキーパーソンが統合テストを担当すると、テストの品質が高くなるだけでなく、テスト効率も高まる。特に、納期に間に合わせるため、複数の作業を並行して進めるようスケジュールを変更するときには、

図表3-7●要員遷移俯瞰図の例



キーパーソンがネックにならないようなスケジューリングが絶対条件となる。

要員遷移俯瞰図は、専門家あるいは業務知識を持つなどのキーパーソンを考慮してプロジェクト体制を検討する場面や、プロジェクト固有の事情を加味したクリティカル・パスのスケジュールを検討する場面で利用する（**図表3-7**）。また、要員遷移俯瞰図を用いて、プロジェクト・メンバーにプロジェクトの特徴と全体の作業を意識付けることが重要である。

要員遷移俯瞰図は、次工程の準備段階ごとに、必要に応じて作成する。さらに、対象となる要員には、この図に基づいて次工程の作業を説明する。

3.3 チェックシートを使った見える化

要件定義・基本設計段階では、問題自体は表面化せず、リスクとして存在していることが多い。その将来起こり得る問題を見通し、適切に対策を打っていくことが重要である。見える化のツールであるチェックシートは、上流工程におけるプロジェクト・マネジメントの要点を示し、問題点を明らかにすることで、どの領域にリスクが多く存在しているかを見えるようにする。

チェックシートには、「自己評価シート」と「ヒアリングシート」の2種類がある。自己評価シートは、プロジェクト・マネージャやプロジェクト・リーダーが自らプロジェクトの問題点やリスクを明らかにするために用いる。ヒアリングシートは、専門家によるヒアリングを通して、プロジェクトの外側から客観的に問題を指摘するためのチェックシートである。

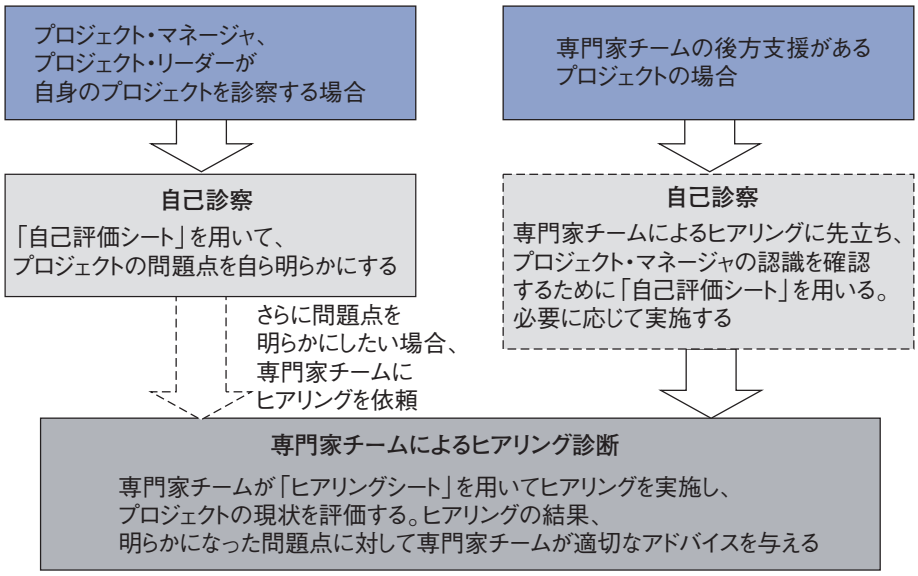
この2つのチェックシートを使用することによりプロジェクト・マネージャが認識できない問題を明らかにできる。自己評価シートとヒアリングシートは、付録資料にその全体を掲載する。

ただし、実際のプロジェクトで2種類のチェックシートを利用する際は、SECのWebサイトからダウンロードできるExcelファイル形式のチェックシートを利用してもらいたい。入力した評価の値から、各種の判定を自動計算できる。以下の説明は、このExcelファイル形式のチェックシートに基づいている。

3.3.1 チェックシートを用いた見える化の流れ

チェックシートは、次の3点を想定して作成されている。まず想定する利用場面は、要件定義の終了直前から基本設計の中盤頃まで。想定するプロジェクトは、業務アプリケーションを開発するプロジェクトとする。想定するプロ

図表3-8●チェックシートを用いた見える化の流れ



ジェクト・マネージャは、受注者側（SIベンダーなど）のプロジェクト・マネージャである。

したがって、適用工程が異なったり、想定するプロジェクトが異なったり、あるいは立場が異なるプロジェクト・マネージャの場合にはチェックシートの読み替えが必要になる。この読み替えについては【3.3.6項】で説明する。

チェックシートを用いるきっかけとしては、①プロジェクト・マネージャ、プロジェクト・リーダーが自分のプロジェクトの問題を明らかにしたいケース、②PMOやプロジェクトを監理する立場の者（専門家チーム）がプロジェクトの問題を見えるようにしたいケースの2つがある（**図表3-8**）。

3.3.2　自己評価シート

自己評価シートは、プロジェクト・マネージャがプロジェクトの問題点を明らかにするため、または専門家チームによるヒアリングに先立ち、プロジェクト・マネージャの現状認識を確認する場合に使用する。**図表3-9**に自己評価シートのイメージを示す。

自己評価シートには、実際にプロジェクトで行っているマネジメントについて35のチェック項目を用意している。チェック項目のそれぞれについてプロジェクト・マネージャが3段階で評価を入力すると、分野別に判定結果を表示し、

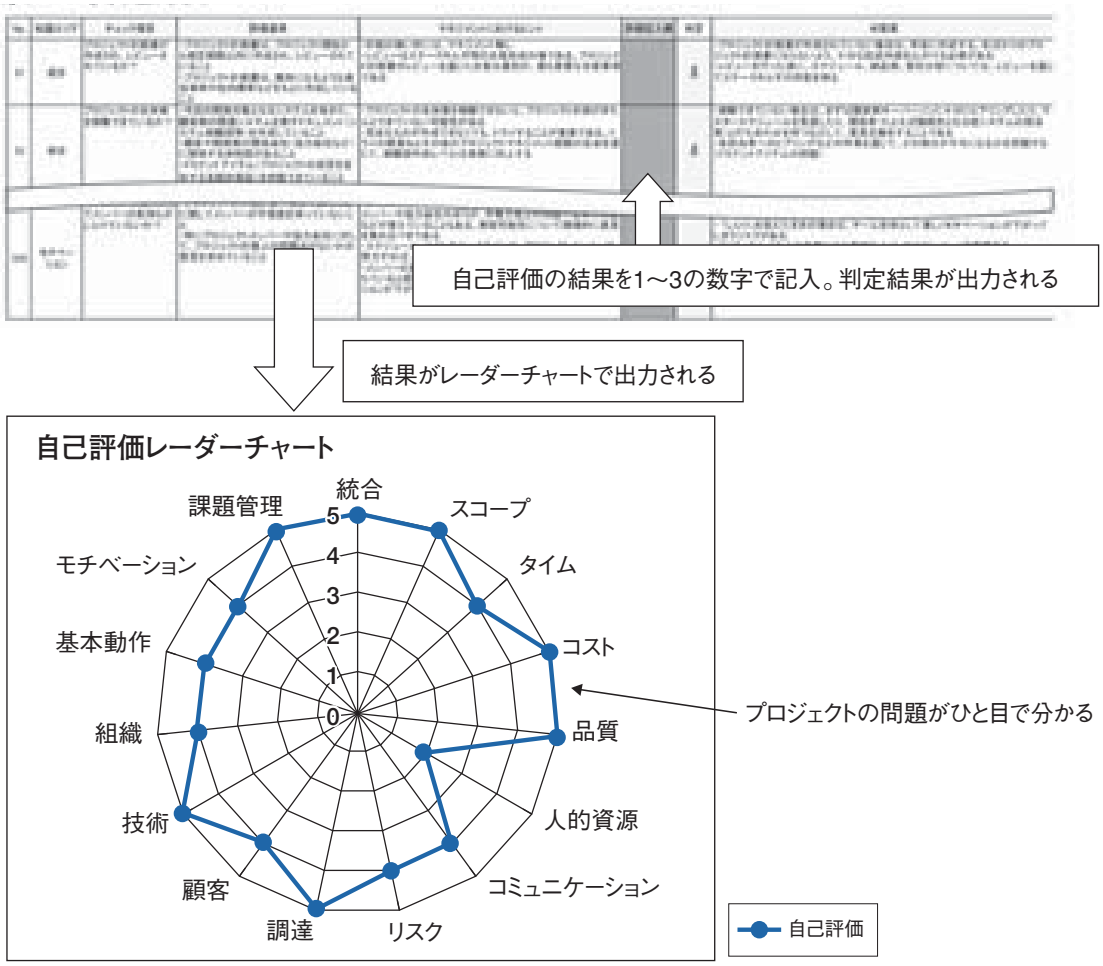
レーダーチャートとしてグラフ化する仕組みだ。レーダーチャートを見ると、プロジェクト・マネジメントの問題がひと目で分かるようになっている。

自己評価シートには、チェック項目や判定結果のほか、プロジェクト・マネージャに気付きを与えるヒント（マネジメントにおけるヒント欄）が含まれている。また、対策案欄では、問題があるチェック項目への対処方法を説明している（**図表3-10**）。

3.3.3 自己評価の実施方法

自己評価シートを用いて自己評価を実施する際には、まずチェック項目の質問とともに「評価基準」と「マネジメ

図表3-9●自己評価シート



ントにおけるヒント」をよく読んで、質問の趣旨を理解する必要がある。そのうえで、できるだけ客観的な目でプロジェクトの現状を評価することである。

自己評価シートに記述されているチェック項目は汎用的な表現になっているため、評価するプロジェクトによっては読み替えが必要な項目もある。読み替え方法については【3.3.6項】で説明する。もし読み替えもできない場合は、「対象外」の項目とする。

プロジェクト・マネージャは、自己評価シートの各チェック項目に対して、「評価記入欄」に「1」、「2」、「3」の3段階で数字を入力する。最低レベルの「1」は、「チェック項目のマネジメント

図表3-10●自己評価シートの項目説明

項目	項目説明
No	“S”で始まる連番が付けられている。チェック項目を識別するのに使用する
知識エリア	チェック項目が属する15種類の知識エリア名。PMBOKの9つの知識エリア（統合、スコープ、タイム、コスト、品質、人的資源、コミュニケーション、リスク、調達）に加え、PMBOKにヒューマンウェアの部分を追加している。追加しているのは、「顧客」「技術」「組織」「基本動作」「モチベーション」の5つとプロジェクトの監視コントロールに注目した「課題管理」である
チェック項目	質問の内容が記述されている。この質問に従って、あとの評価記入欄に結果を入力する。評価基準チェック項目の質問を補足する、具体的な評価基準の例が記述されている
マネジメントにおけるヒント	チェック項目を評価する際のポイントとともにマネジメントを行う際に気付きを与えるヒントが記述されている。特に自己評価のみ実施する場合にはこの項目をよく読んでチェック項目の趣旨と何を気付かせようとしているのか理解する必要がある
評価記入欄	この欄に評価結果を1～3の数字で記入する。このチェック項目が評価対象のプロジェクトに該当しない場合には－1を記入する。評価結果の入力方法は【3.3.3項】で説明する
判定	判定結果がA、Bまたは空欄にて表示される。判定の見方は【3.3.3項】で説明する
対策案	判定がAまたはBの場合の対策案が記述されている。対策案は、一般的なプロジェクトを基に考えられているので、評価対象プロジェクトでは読み替えが必要な場合もある

図表3-11●自己評価シートの評価基準

入力値	評価基準
1	チェック項目のマネジメント方法を知らない
2	チェック項目のマネジメントとして何をしなければならないかは分かっているが、様々な事情によりうまくできていない
3	チェック項目のマネジメントとして何をしなければならないかが分かっており、うまくできている
－1	対象外。評価対象のプロジェクトには、チェック項目に該当するものがない

方法を知らない」とする。具体的な評価基準は**図表3-11**を参照してもらいたい。対象外となるチェック項目の場合には「−1」を入力する。

評価記入欄の入力を行うと、判定欄に判定結果が表示される。判定結果は「A」、「B」か“空欄”であり、「A」は対策が必要な項目、「B」は注意が必要な項目である。特に何も表示されない（空欄である）場合は、「特に問題なし」という判定である（**図表3-12**）。

35のチェック項目に対して回答が完了すると、自己評価レーダーチャートが作成される。この自己評価レーダーチャートを見るとプロジェクトの弱点を洗い出せる。すなわち、レーダーチャートでは、問題のある知識エリアが低い評点で表示される（**図表3-13**）。

自己評価シートの評価欄やレーダーチャートを見て問題点、弱点が分かれば、マネジメントにおけるヒントや対策案を参照してプロジェクトに対する具体的な対策を考えることができる。

自己評価シートの評価結果については、評価記入者（プロジェクト・マネージャやプロジェクト・リーダーなど）による主観的判断が入る場合もある。このため、事実と異なる可能性があるという点に注意する必要がある。

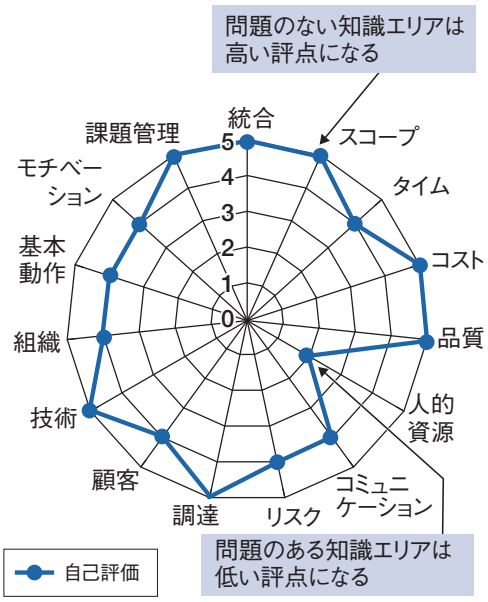
また、自己評価シートは、必ずしもすべてのプロジェクトで問題を網羅的に摘出できるわけではない。プロジェクトの上流工程で問題になりやすい状態についての汎用的なチェック項目で構成されているためだ。チェック項目の記述内容に固執し過ぎずに、ITプロジェクトを取り巻く環境に応じた判断をするように心がける必要がある。

自己評価シートによる評価で不十分な場合は、さらに専門家チームにヒア

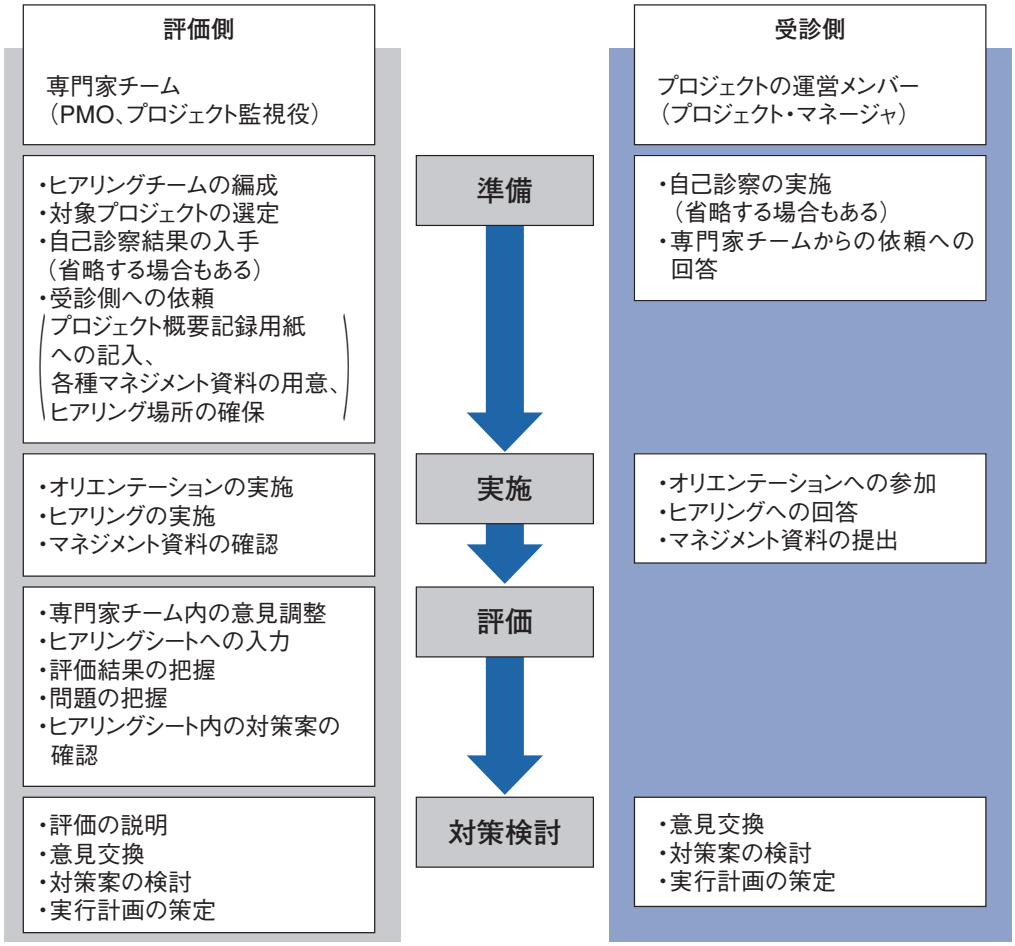
図表3-12●自己評価シートの判定

判定結果	内容
A	対策が必要な項目
B	注意が必要な項目
空欄	問題なし

図表3-13●自己評価シートのレーダーチャート



図表3-14●専門家チームによるヒアリングの流れ



リングを依頼する。なお、自己評価シートは、環境の変化に応じてアップデートしていくことが必要である。

3.3.4　ヒアリングシート

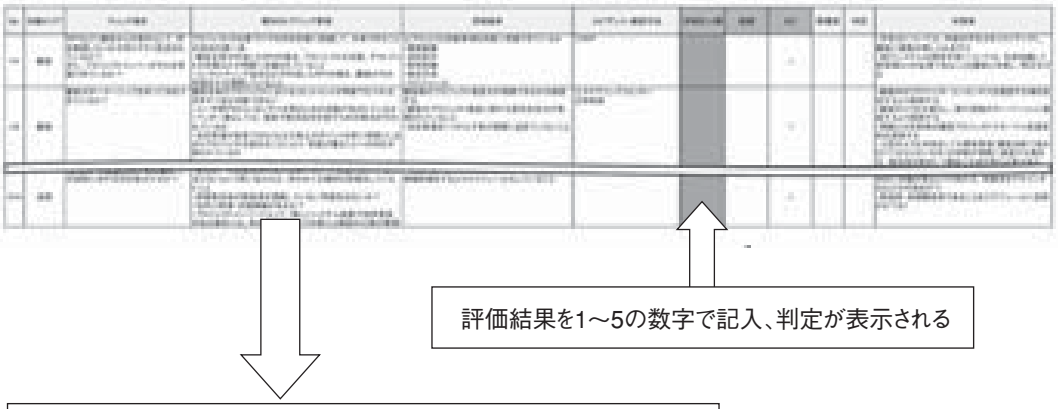
ヒアリングシートは、PMOやプロジェクトを監理する立場の者（専門家チーム）が、チェック対象のプロジェクトの問題点やリスクを明らかにするために使用する。ヒアリング作業は、ヒアリングを行う評価側とヒアリングを受ける受診側が協力して行う。

ヒアリングは自己評価シートでの評価と異なり、相応の準備と手順が必要となる。基本的なヒアリングの流れは**図表3-14**のようになる。

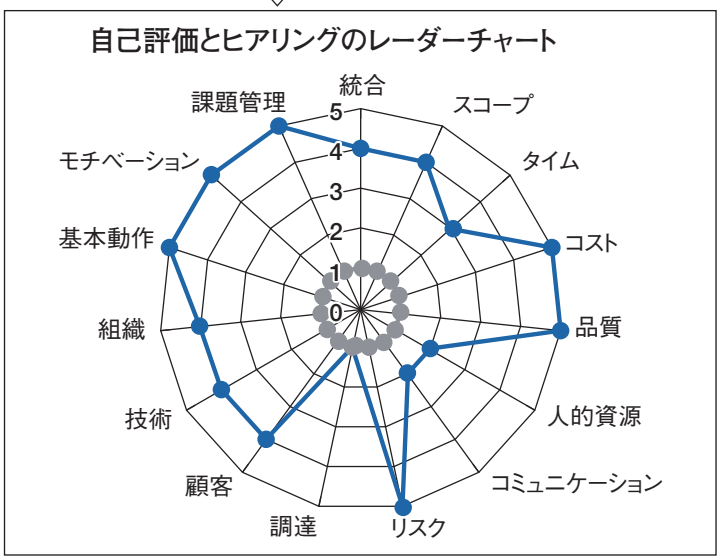
ヒアリングシートの評価結果は、専門家チームと受診側が対策を考える際の基本資料となる。ヒアリングシートは74項目のチェック項目からなり、評価記入欄に5段階で評価を入力する。これにより、チェック項目ごとに判定結果が表示されるとともに、レーダーチャートに分野別の判定結果が図示される（**図表3-15**）。

ヒアリングシートは、専門家チームが用いるように構成されているため、自己評価シートにあるプロジェクト・マネージャに気付きを与える項目はない。代わりに、チェック項目ごとのヒアリング要領や、エビデンスなどによるチェック項目の確認方法が記載されている

図表3-15●ヒアリングシート



評価結果を1～5の数字で記入、判定が表示される



診断結果が分かりやすく
レーダーチャートで表示される

(詳細は**図表3-16**参照)。なお、チェック項目は概要から詳細・個別へと、ヒアリングしやすいように並べてある。

3.3.5　ヒアリングの実施方法

ヒアリングは、図表3-14で示した通り、(1)準備→ (2)実施→ (3)評価→ (4)対策検討、という4段階の手順で実施する。

(1)準備作業

ヒアリングを実施するに当たり、

図表3-16●ヒアリングシート項目

項目	項目説明
No	“H”で始まる連番が付けられている。チェック項目を識別するのに使用する
知識エリア	チェック項目が属する15種類の知識エリア名。PMBOKの9つの知識エリア(統合、スコープ、タイム、コスト、品質、人的資源、コミュニケーション、リスク、調達)に加え、PMBOKにヒューマンウェアの部分を追加している。追加しているのは、「顧客」「技術」「組織」「基本動作」「モチベーション」の5つとプロジェクトの監視コントロールに注目した「課題管理」である
チェック項目	チェックする質問が書かれている。ヒアリング時にこのチェックに対する回答を評価記入欄に入力する
個別のヒアリング要領	ヒアリングを行う際のポイント、チェックする内容の詳細、そのチェック項目の確認方法が記載
評価基準	チェックする際の評価基準が説明されている
エビデンス・確認方法	チェック項目を判断する際に確認すべきエビデンスの種類がリストアップされている
評価記入欄	チェック項目に対する回答を1～5の数字で記入、具体的な入力方法は【3.3.5項】で説明する
影響度	チェック項目が判定に与える影響度合いを示す。この値が大きい(1.0)項目はよりリスクが高いと見なされる。チェックシートに記載されている値は、リスク分類表(5章で説明)へのマッピング度合いや、チェック項目の評価の具体性を考慮して重み付けされた値が設定されている
判定	チェック項目に対する判定をA、B、空欄で判定する、判定内容は【3.3.5項】で説明する
対策案	チェック項目に対して問題があった際の対策案が記述されている、プロジェクトによっては読み替えが必要な場合もある

図表3-17●専門家チームの編成

役割	内容	必要な経験/能力
ヒアリング担当者	専門家チームのリーダーとしてヒアリングを実施する	豊富なプロジェクト・マネジメント経験を持っていること。プロジェクトで「今後起こる問題」を見通す能力が求められる
観察者	ヒアリング対象者の表情・仕草などを観察する。ヒアリング担当者の兼務も可能	
記録者	ヒアリング結果を記録する	

PMOやプロジェクト監理担当者からなる専門家チームを編成する。専門家チームは、「ヒアリング担当者」を中心に、ヒアリング対象者の表情や仕草などを観察する「観察者」、ヒアリング結果を記録する「記録者」で構成する（**図表3-17**）。

専門家チームを編成したら、受診側に次のような作業を依頼する。まず、①自己評価シートを提出してもらう。急を要する場合であれば、省略することもある。次に、②プロジェクト概要書の作成、③各種マネジメント資料の用意をしてもらう。これらはヒアリングのための基礎資料となる。そして最後に、④ヒアリングの場所を確保してもらう。

専門家チームから依頼された受診側チームは、ヒアリング実施までに依頼されたことを実施しておかなければならない。

①自己評価シートの提出は、専門家チームがヒアリングを効果的に進めるために、プロジェクト担当者側が自分

図表3-18●プロジェクト概要の項目説明

項目	説明
開発システム	開発対象のシステムの5W1H（目的、ユーザー、利用場面・場所、入出力、システム形態/処理内容）を簡潔に記述する
特徴	開発対象システムにおいて特徴的な事柄を記述する。具体的には、新技術・未経験技術の適用有無、新規顧客か否か、新規業務/業務改革の有無、新しい開発体制（新しいプロジェクト・マネージャ、新しい協力会社）、品質・コスト・納期に対する制約の度合、大規模処理の有無、パッケージ利用の有無、特徴的な開発ツールの利用有無、遠隔地開発の有無、現行システム保証の度合、開発において制約となる事項（開発手法、モジュール共通化の要請、既存システムの再利用など）の有無など
プラットフォーム	OS、DBMS など開発対象システムのプラットフォーム
プログラム開発規模	開発対象システムのステップ数、ファンクション・ポイントなど
開発時期	開発の開始時期と終了予定時期
開発期間	開発期間
開発形態	最終顧客と開発にかかわる組織を記述する。請負開発が多重構造になっている場合は、請負構造をすべて記述
プロジェクト要員数（延べ）	プロジェクトにかかわる延べ要員数
プロジェクト要員数（現時点）	現時点でプロジェクトにかかわる要員数
現在のフェーズ	現在の進捗状況が、スケジュール上のどこに当たるのかを記述
あなたが感じている課題	現在の課題認識を記述

図表3-19●ヒアリング時に開示できるようにすべき資料

種別	資料名
俯瞰図	周辺システム構成俯瞰図、システム構成俯瞰図、プロジェクト推進体制俯瞰図、ステークホルダー俯瞰図、スケジュール俯瞰図、要員遷移俯瞰図
プロジェクト計画・ルール	プロジェクト計画書、リスク管理表、責任分担表、体制図、WBS、システム規模見積もり、テスト計画書、レビュー計画書、構成管理計画書、移行計画書、保守・運用計画書、要員育成目標、レビュー実施要領、変更管理ルール、チェックリスト類
スケジュール	マスター・スケジュール、チーム別スケジュール、要員山積表
進捗管理・報告書	課題管理表、リスク管理表、変更管理表、進捗報告書、メンバーの勤務管理表、各種議事録
設計書	RFP、要件定義書、基本設計書、用語集
契約	各種契約書、協力会社との契約書、提案書、見積書

自身をどう評価しているのかを明らかにすることが目的である。ヒアリング後の評価の際に、自己評価シートとヒアリングシート両方の評価結果が入力されていると、両方のレーダーチャートが表示され、相互の認識の相違点を明らかにできる。

②プロジェクト概要書は、専門家チームがヒアリング対象プロジェクトの性格を知るために、受診側チームに作成してもらう。主な記述内容は、開発システムの目的、ユーザー、利用場面・場所、入出力、システム形態などにはじまり、技術面やマネジメント面でのリスク、制約などを記述する。プロジェクト概要の詳細な項目説明は**図表3-18**を参照してもらいたい。

③各種マネジメント資料は、チェック項目それぞれの判定材料にする。各種俯瞰図やプロジェクト計画書、マスター・スケジュールなどである。ヒアリング時に、口頭説明と併せて見せることができればよい。

したがって、元々ないものを改めて作成する必要はなく、マネジメント資料のコピーを専門家チームに渡す必要もない。ヒアリング時に開示できるようにすべきマネジメント資料の一覧を**図表3-19**に示す。

④ヒアリング場所は、マネジメント資料を参照しやすい場所にするとよい。ヒアリングの中で、必要に応じてエビデンスの確認を行うので、実際にマネジメントを行っている場所の近くが好ましい。紙の資料ならドキュメント・ファイルなどをヒアリング場所に用意し、電子データになっているならすぐに見せられるように準備しておく。

(2)実施段階

専門家チーム側は、ヒアリング実施前に受診側から提供された自己評価シート、プロジェクト概要書について、あらかじめ理解しておく。

ヒアリングを実施する際には、まず①10分程度のオリエンテーションを実施し、その後に②2時間程度をかけてヒアリングシートを用いたヒアリングを実施する。

①ヒアリング前のオリエンテーションは、受診側にヒアリングの目的や実施手順を説明するための時間である。ヒアリング実施の目的を説明する際には、プロジェクトの実施状況を評価し、問題やその予兆を発見することによって適切な対処を行い、プロジェクトを成功させることが目的だと受診側に理解してもらうことが重要である。ヒアリングを受ける受診側はどうしても身構えてしまい、情報をガードする傾向がある。ヒアリングをスムーズに行うには、プロジェクト・メンバーと一緒にプロジェクトの問題点を洗い出す雰囲気作りが重要である。

目的を説明したあとに、専門家チームおよび受診側それぞれが自己紹介する。続いて専門家チームがヒアリングの進め方について説明する。受診側から提供されたプロジェクト概要書によりプロジェクトの性格を確認する。

②ヒアリングの実施は、ヒアリング担当者がヒアリングシートの内容に従い順次質問していく。ヒアリング担当者は、チェック項目を読み上げたあと、具体的な例を示すなどして、チェック項目の質問の意図を分かりやすく説明する。専門家チーム側は受診側の回答を基に評価し、ヒアリングシートに記入するとともに、必要に応じてエビデンスを確認していく。

チェック項目によっては、プロジェクトの状況に応じた読み替えや、当該項目を対象外とするかどうかの判断が必要なときがある。これらの判断はヒアリング担当者が適宜行う。疑問点や違和感があった場合には、問題を深掘りすることも必要である。

受診側の表情･仕草などは、観察者が記録していく。記録者は、受診側の回答やエビデンスの状況を記録していく。

(3)評価段階

ヒアリングの実施後、ヒアリング場所とは別の場所に移動してから、専門家チームのメンバーによる評価会を開く。専門家チーム内で円滑に評価・検討できるよう、評価会には受診側メンバーを参加させないようにする。

「実施段階」でヒアリングをしながら評価を行っているが、ヒアリングシートに記載されている評価基準をよく読み

図表3-20●ヒアリングでの評価レベル

評価レベル	評価基準
1	質問領域のプロジェクト・マネジメントがわかっていない。しかも、的外れなマネジメントを実施
2	質問領域のプロジェクト・マネジメントがわかっていないまま、勘と度胸のマネジメントを実施
3	質問領域のプロジェクト・マネジメントはわかっているが、ほとんどできていない
4	質問領域のプロジェクト・マネジメントはわかっているが、内部・外部の要因によって、そのすべてはできていない
5	質問領域のプロジェクト・マネジメントがほぼ完璧にできている
−1	対象外、当該プロジェクトには当てはまらない質問であり、読み替えることもできない

直し、主観的な評価ではなかったかを確認する。場合によっては、評価を見直す。また、「対象外」項目がどの項目だったか、専門家チームのメンバー間で意識合わせをする。

各チェック項目について5段階の評価を付ける。評価はヒアリングシートの「評価記入欄」に、1〜5の5段階で数字を入力する。最も低いレベルの「1」は、「質問領域のプロジェクト・マネジメントが分かっていない。しかも、的外れなマネジメントを実施している」という意味である。評価基準の詳細は**図表3-20**を参照してもらいたい。当該項目が対象外となる場合には「−1」を入力する。

ヒアリングシートの各チェック項目には、プロジェクトに対する「影響度(重み付け)」を設定できるようにした。この影響度は、「範囲」欄と「エビデンス」欄の値を基に算出している。

ヒアリングシートでは、以下に示す考え方に基づいて影響度を3段階に分類している。各チェック項目に対する5段階の評価は影響度と掛け合わせ、独自の計算方法に従って「判定」を導き出している。

「範囲」欄は、各チェック項目が明らかにしようとするリスクの範囲にばらつきがあることを調整するためのものである。例えばプロジェクト計画書の不備・不足を問うチェック項目の場合、プロジェクト・マネジメントの面で様々な影響を及ぼす可能性がある。一方、トラフィック量の想定の検討など、方式設計にかかわるチェック項目は、比較的リスクの範囲は限定されると考えられる。

各チェック項目がどの程度の影響度を持っているのかは、リスク分類表

（【5.2節】を参照）へのマッピングの結果から導出した。つまり、特定のチェック項目がリスク分類表の中でたくさんマッピングされているほど、広い範囲に影響を及ぼすと考えられる。広範囲に影響を及ぼせば、当然リスクも大きくなる。そこで、マッピングされた個所が1カ所の場合は影響度を「1」とし、2カ所の場合は「2」、3カ所以上の場合は「3」で表すこととした。

Excelのヒアリングシートの「範囲」欄には、デフォルトで上記の値が入力されている。専門家チームは、ヒアリングを通してデフォルト値を変更すべきと判断すれば、新しい値を範囲欄に入力する。

「エビデンス」欄は、チェック項目に対して具体的、客観的な回答を求める度合いを示している。チェック項目の質問が、エビデンスを含む具体的な回答を求めている場合は重く評価し、回答者の恣意的な判断が多分に含まれるマインドの問題や意識レベル（姿勢）を問うようなチェック項目については軽く評価するという考え方を採る。いくつかの実プロジェクトでの実地検証を通して、回答者や専門家チームの恣意性が混入しやすいチェック項目が見つかったからだ。そこで、具体性、客観性が高いチェック項目の影響度を「2」、恣意性が混入しやすいチェック項目の影響度を「1」とした。

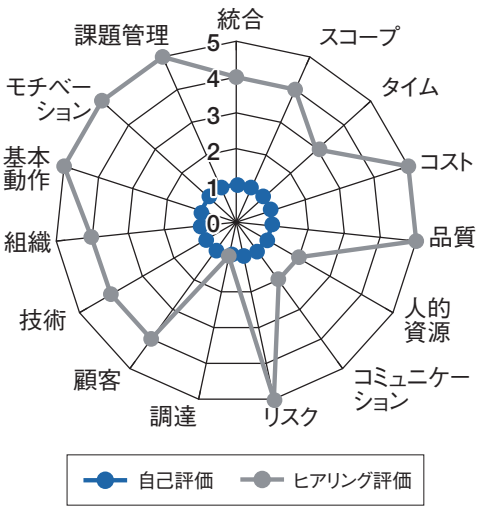
2つの影響度要因（範囲、エビデンス）の値はチェック項目ごとに合計し、統合した影響度を算出している。4点以上なら「1.0」、3点なら「0.7」、2点なら「0.5」という値とした。

各チェック項目に5段階の評価を入力していくに従い、問題の有無を示す「判定」が表示される。判定欄は自己診察と同じく「A」、「B」、“空欄”で示される。判定の意味は自己評価シートと

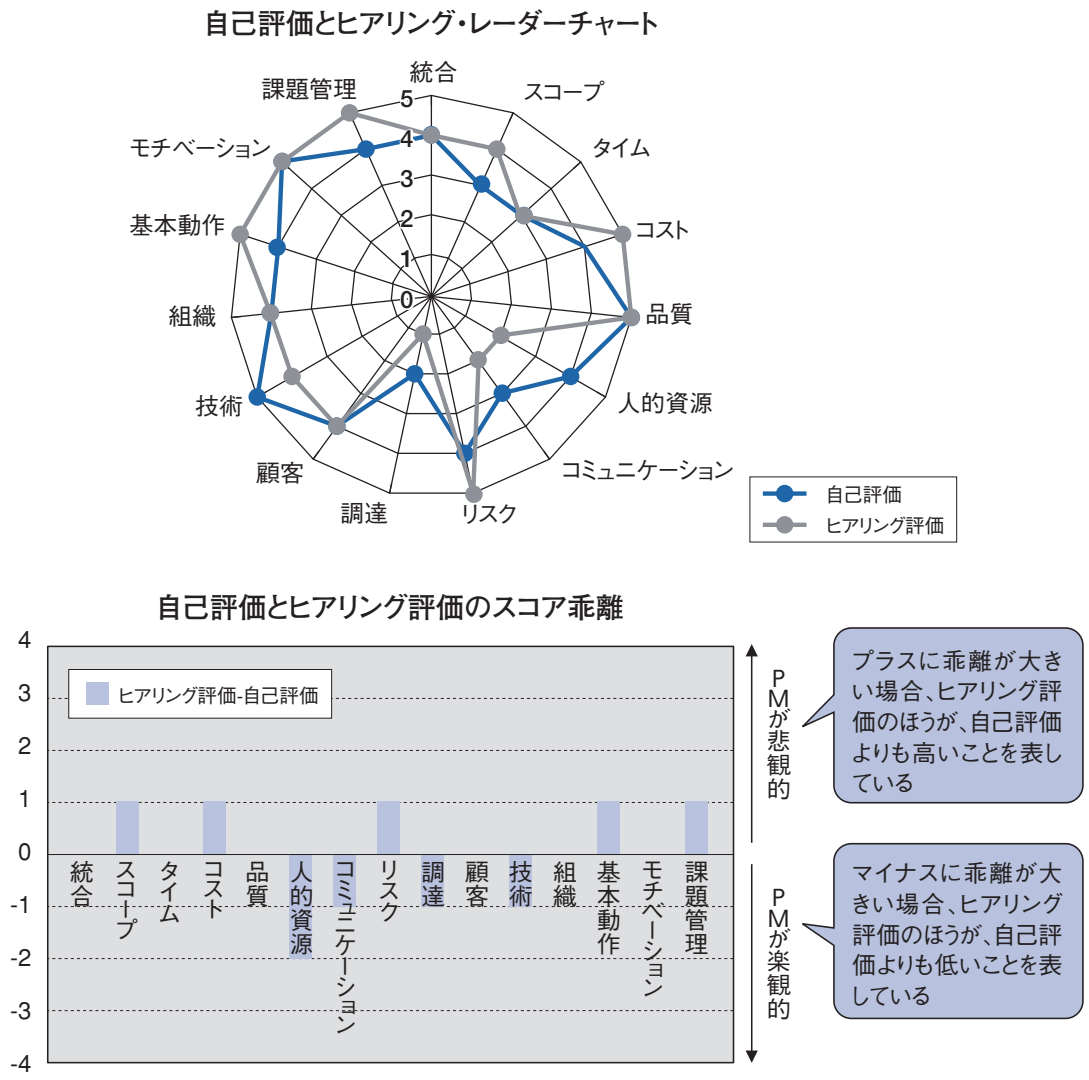
図表3-21●ヒアリングシートの判定

判定結果	内容
A	対策が必要な項目
B	注意が必要な項目
空欄	問題なし

図表3-22●ヒアリングシートのレーダーチャート



図表3-23●自己評価シートとヒアリングシートの入力後の評価



同じである（**図表3-21**）。

入力が終わると、レーダーチャートに知識エリアごとの判定結果が表示される（**図表3-22**）。問題ない知識エリアは評点が高く、問題の知識エリアは評点が低く表示される。

図表3-23に自己評価シートとヒアリングシートの両方を入力し終わったときのレーダーチャートと、それぞれの判定の乖離グラフを示す。

専門家チームは、この3種類の判定結果から問題点を洗い出す。その際、

ヒアリングシート内の対策案欄の内容を参考にして、問題点をより具体的にするとよい。

問題点を洗い出す際には、チェック項目の判定欄が「A」、「B」のものをよく調べ、マネジメント上の個別の問題点を拾い出す。また、ヒアリングシートのレーダーチャートを分析し、マネジメントの傾向を把握することで問題点を洗い出すこともできる。自己評価の判定とヒアリングの判定の乖離が大きい部分を調べ、問題点を掘り出す気持ちで検討するとよい。

問題点が明らかになったら、評価結果と問題点をまとめ評価案を作成する。

(4)対策の検討

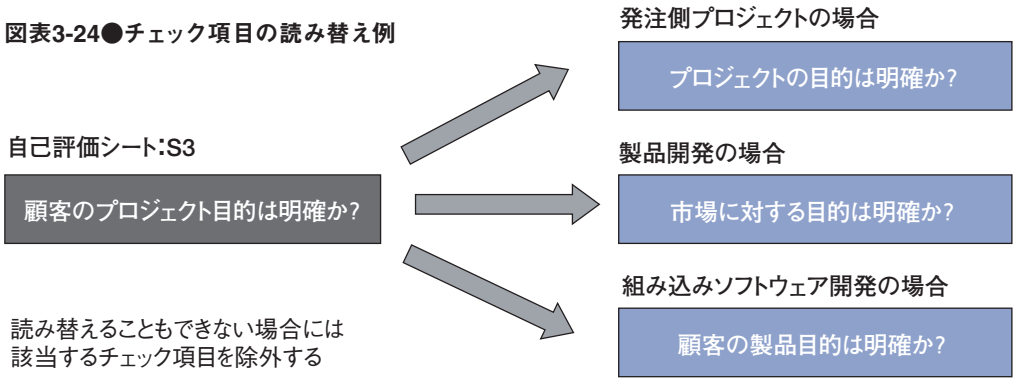
専門家チームは、評価案ができたら、受診側のプロジェクト・マネージャ、プロジェクト・リーダーに評価結果説明会を行う。説明会には、ヒアリングに参加した専門家チームとヒアリングを受けた受診側メンバー全員が出席することが望ましい。評価結果の説明後には、専門家チームと受診側チームが一緒になって問題点への対策を議論する。

3.3.6 評価項目の読み替え

自己評価、ヒアリング評価の中で説明したように、実際のプロジェクトでは、チェック項目に読み替えが必要になる場合がある。自己評価シート、ヒアリングシートとも、要件定義の終わりから基本設計の中盤に利用することを想定している。評価は、業務システムの開発を受注した側のプロジェクト・マネージャに対して行われる。この想定から外れた場合に、評価項目の読み替えが必要になる。

読み替えが必要な例としては、発注側のプロジェクト・マネージャが評価を行う場合や、パッケージ製品開発な

図表3-24●チェック項目の読み替え例



ど自社で仕様を決められる場合、また組み込みソフトウェア開発の場合などがある。**図表3-24**にチェック項目の読み替え例を例示する。読み替えできない場合には、該当するチェック項目を対象外とする。

3.3.7　チェックシートの改良

自己評価シート、ヒアリングシートとも、IT業界の動向や経営環境、自社の位置付け、自部署の状態が変化していくに従い、本当の問題点を示せなくなっていく。このため、自己評価やヒアリングの結果をプロジェクトの実態と比較することで、チェックシートをブラッシュアップし、常に“使える状態”に維持しておく必要がある。ブラッシュアップを怠ると、チェックシートは陳腐化し、使いものにならなくなる。

ブラッシュアップのためには、「チェック項目自体を変更する」、「マネジメントにおけるヒントや対策案などを変更する」、「ヒアリングシートの影響度を修正して、適切な判定が得られるように変更する」という3つの方法が考えられる。プロジェクトへの適用を積み重ねる中で、継続的にブラッシュアップしていくことが重要である。

3.3.8　チェックシートの実地検証

SECのプロジェクト見える化部会では、上流工程向けの自己評価シートおよびヒアリングシートの有効性を実証するために、実際に進行中のプロジェクトへ適用を試みた。チェックシートに関する課題の抽出と見直しを繰り返しながら、合計8プロジェクトに適用し、最終的に付録資料に示すチェックシートの内容にまとめた。以下に実プロジェクトに対するチェックシートの実証の概要を示す。

本来、付録資料に示すチェックシートは、要件定義の終了ごろから、基本設計中盤までの時期に利用することを想定しているが、チェックシートの実地検証に協力してもらったプロジェクトとのスケジュールが合わない場合もあり、実際には現状分析からテスト工程まで様々な時点で適用することになった。また、ヒアリング対象としては受注側プロジェクトを前提としているが、発注側のプロジェクトも含まれていた。

ヒアリングを行う専門家チームはプロジェクト見える化部会のメンバー5人から7人で構成し、あらかじめヒアリング担当者を決め、参加メンバー全員で観察と記録を行った。ヒアリングを受けるプロジェクト側は、プロジェクト・マネージャを中心に、プロジェクト・リーダーや品質管理担当者が参加する場合もあり、1人から4人であった。

プロジェクト概要書の記入用紙と自

己評価シートをヒアリング対象プロジェクトに事前に渡し、ヒアリング前に回収して専門家チームのメンバーに配布しておいた。ヒアリング場所は、開発作業を実際に行っている場所に近い会議室で行い、管理帳票などの準備を依頼した。ヒアリング時間は2時間を目標に実施した。

ヒアリングは、ヒアリングシートのチェック項目をプロジェクタで映しながら、一項目ずつ読み上げ、回答を求めた。専門家チームのメンバーには、ヒアリング開始時にプロジェクト概要記述用紙とヒアリングシートを配布し、チェック項目ごとの回答や説明に対して、その場で評価を記入し、気になる発言などを書きとめた。また、プロジェクトによってはチェック項目の一部を読み替える必要があったが、その場で協議してどのように読み替えるかを決めた。事象が発生していない項目、例えば課題が発生していない時点での課題管理に関する質問などについては、ほかのチェック項目への回答内容を勘案して評価することにした。

付録のチェックシートでは、初めにプロジェクトを大きくとらえた質問が続くため、初めの項目ほど内容確認に時間がかかるが、後半は比較的簡単な確認項目になる。時間配分を考える場合には、全体の時間をチェック項目数で均等に割らずに、前半に時間をかけてプロジェクトの全体像を確実に把握したほうがよい。

ヒアリングの最後には、ヒアリングを受けたことに対する感想などを聞き、チェックシートを使ったヒアリングについて意見交換した。次に、判定結果を説明する打ち合わせ日程（1週間後が目安）を決めて終了した。

ヒアリング終了後、専門家チームは別施設の会議室に移動して評価会を開き、ヒアリング結果のまとめと対策が必要な課題について検討した。ヒアリングした内容を忘れないよう、同じ日に評価会を開催するようにした。

この評価会では、対象外とすべきチェック項目を最初に確認した。その後、対象プロジェクトの課題として、対策を検討すべき項目についてのブレーンストーミングを実施。抽出された課題に対する対策案について検討した。

また、チェック項目の内容や順番などについても検討し、チェックシートの見直しも行った。検討にかけた時間は、プロジェクトの課題の多さにもよるが1時間から3時間程度だった。なお、各チェック項目に対する評価は、専門家チーム・メンバーの評価点の平均値を採用した。

実地検証で抽出された問題は、例えば以下のようなものがあった。

「責任分担が曖昧になっている」
「ドミナント・アイテムが把握できていない」
「見積もり根拠が曖昧になっている」
「リスク管理と課題管理の区別ができていない」
「課題の重要度に対する判定基準がない」

ヒアリング結果の説明会では、まず自己評価結果を確認するとともに、ヒアリング結果でA判定となった項目についての確認、ヒアリング結果でB判定となった項目の紹介（詳細な説明は省略）をした。続いて、自己評価とヒアリング結果の差異（乖離）について確認し、課題とその対策案について説明した。すべてを説明するのに1時間程度を使った。専門家チームの認識が間違っている項目がないかを質問しつつ、提示した対策案について意見交換もした。

ヒアリングを受けた側の感想としては、以下のようなコメントがあった。

「客観的にアドバイスしてもらえるのはうれしい」
「客観的にヒアリングされることで気付くことがある」
「的を射た聞き方で回答を導き出してもらえたのがよかった」
「日頃なんとなく問題を感じているが、文書化されると、見えなかったものが見えてきた」
「課題のあるところを、さらに深掘りしてヒアリングすると効果的である」

8つのプロジェクトへのヒアリングを通して、チェックシートのチェック項目内容を見直し、概要から詳細になるように項目の順番を入れ替え、協力会社に発注する場合や業務改革を含む場合など、プロジェクトの内容や条件によって選択する項目をまとめた。また、対象外の項目については評価を「−1」として処理できるようにした。

チェックシートが本来想定している工程や立場と異なるプロジェクトでもヒアリングを行ったが、チェック項目を読み替えることで対応できることが分かった。すなわち、チェックシートを使った「見える化」の枠組みはそのままで、チェック項目の内容を読み替えることによって、様々な進行状況のプロジェクトに適用できることが判明した。

ヒアリング時間については、最初は2時間で終わらなかったが、項目の見直しや順番を変更したことに加え、次第に慣れてきたこともあり、最終的には2時間に収まるようになった。チェック項目を詳細に確認するには、2時間では不十分かもしれないが、プロジェクトの状況を大まかにとらえることはできる。

ヒアリング対象となるプロジェクトの

事情があらかじめ詳細に分かっていれば、ヒアリングする必要のない項目もあると思われる。プロジェクトの性格に合わせて事前にチェック項目を見直しておくと、効率的にヒアリングを実施できる。

3.4 上流工程事例

3.4.1 事例集の公開

上流工程におけるプロジェクト推進の難しさは、最終的なゴールがはっき

図表3-25●プロジェクト事例のサンプル

事例番号 1 **顧客側担当者が異動になり約束事が反故に**

顧客側の担当課長レベルと費用および作業内容について内々に話をつけ仕事をしたが、顧客側の職制が変更になった（前任者は退職）。後任者は前任者からこのことについて全く引き継ぎがなかったため今までの話がご破算になった

事例における見切り	内容本来の見切りの考え
決められた社内ルールを無視して顧客との契約をせずに開発に着手	●費用と作業内容については最低限、文書で取り交わしておく。作業を実施するに至った経緯についても記録しておき、たとえ後付けでも作業の妥当性を顧客側の新担当者に説得できるようにしておく必要がある ●その非公式な約束事が反故になるケースを想定し、そのリカバリ方法が確保できていて、かつリカバリコストへの対策が決まっていれば見切ってもよい
捉えるべき徴候	**対処例**
●顧客側の人事異動情報	●契約する前であれば、機能を削減し、予算判明以降に発生する赤字を減額する ●契約内容を確認し、プロジェクト単体が最小限の損害で契約を満たす方法を検討する ●システム化対象範囲を分割して複数年度の契約にする等、戦略的に採算が取れるような交渉を進める

図表3-26●事例の項目

事例番号、タイトル、概要	事例に付けられた通番とタイトル、トラブルの概要を説明
事例における見切り内容	プロジェクト・マネージャがその事例でどのような見切りを行ってプロジェクトを推進したか、またそのときの状況
本来の見切りの考え方	本来はどのような観点で見切り判断を実施すればよかったのかの説明、および補足コメント
捉えるべき兆候	そのような状況に陥る兆候としてどのようなものがあるかを例示
対処例	見切りを行うときの対処例や、問題が起こってしまったあとの対応方法の例示

図表3-27●事例一覧

事例番号	事例タイトル
1	顧客側担当者が異動になり約束事が反故に
2	本当のユーザーを見誤ってしまう
3	新サービスと非合理的な納期
4	仕切る気がないとりまとめ役
5	発注側から要件が小出しにされ、マネジメントできなくなる
6	体制の事前確保が仇に
7	営業がプロジェクト・マネージャの承認のないまま概算見積もりで受注契約を結んでしまった
8	基本設計の承認なしで詳細設計に着手
9	評価が終わっていない基本設計で協力会社に着手依頼
10	性急な立ち上げでの体制確保のため、要員調達でコスト高に
11	業務経験・業務知識の乏しい要員で要件定義工程を実施
12	コーディングの標準化を実施する時間を惜しんだためにかえって非効率に
13	法務部同士で合意に至らず契約が破談
14	初物プログラム言語での開発で十分な性能が出ない
15	現行システムと新システムの並行開発
16	運用テスト入ってから現場固有の業務要件が多数発覚
17	標準仕様に先駆けてシステム構築したが、標準化への対応で品質問題が発生
18	仕様検討に時間を要し、十分なテストができないまま受け入れテスト(1)
19	仕様検討に時間を要し、十分なテストができないまま受け入れテスト(2)
20	顧客側メンバーの体制が弱く、仕様が詰められない
21	不条理な予算圧縮要請で開発に着手
22	さほど大きな影響はないだろうと思い、既存システムのプログラムを修正
23	時間のかかる顧客側承認プロセスを見切って、承認前に作業着手
24	契約金が未決定ながら進めたプロジェクトだったが、契約時に予算額が大幅に減額
25	保守フェーズにおける作業慣習で、仕様変更の見積もりとともに作業を着手してしまった
26	最新技術てんこ盛りで、納期も短いシステム構築で手戻り多発
27	DBMSが詳細設計後に決まって手戻り
28	開発効率向上のためのフレームワーク作りがボトルネックになってしまった
29	プロジェクト・マネージャに大規模プロジェクトの経験がなく、結合テストの頃から調整不足が露呈

事例番号	事例タイトル
30	未知のパッケージを協力会社から提案してもらったが、そのパッケージに品質問題が発生
31	経験の少ないオープン系システムで、プロジェクト・マネージャがコントロール不能に
32	複数会社でのプロジェクトで旗振り役が不在
33	顧客指定のフレームワークで受注し、大幅コスト増
34	本番データに現場部門のパッチが当たっており、論理的不整合が多く移行コスト増大
35	現行システムを再利用した新システムを導入しようとして例外処理に苦しむ
36	顧客側からの要件出しが細切れで、なかなか要件を確定してくれない
37	既存資産が流用できると思っていたが、流用できないことが分かりコストオーバー
38	顧客との共同開発という名目で、イニシアチブが取れずに要件がなかなか確定できない
39	何でも言うことを聞くという応札条件の案件
40	パッケージの品質と機能は十分に確認しよう
41	仕様変更に関する扱いは、顧客と意識合わせをしておくこと
42	大幅な再利用を前提とした開発は要注意
43	作りながら仕様を確認
44	知らないもので決まっていないものを作る
45	無理な見積もりを通すと作り手か使い手のどちらかが破綻する
46	個別に検討したものを繋げてみたら、繋がらなかった
47	顧客側の体制、顧客との役割分担、作業分担を明確にすること
48	オフショア開発では、ドキュメントの記述は詳細に、プロジェクト管理はしっかりと
49	顧客の体制が弱いときは、仕様追加、変更が多発する可能性が大
50	規模が大きくなくても、協力会社に丸投げは危険
51	信頼関係だけでは適切な課題管理はできない
52	インパクトが大きい要件への対処は先送りしない
53	フィット&ギャップ分析をせずにパッケージ導入の決断はしない
54	過去データ移行の難易度はプロジェクト・コストにインパクトあり
55	中核と思われる機能以外に相応の開発規模が隠れていることも
56	ドキュメントの成果レベルは事前調整が不可欠
57	RFPとの差異が明確になっているか
58	パッケージ開発では業務要件とのギャップが重要な見極めのポイント

りと見えていない段階で作業を進めなければならない点にある。たとえ設計書が完成しつつある状況においても、突然、思いもよらないところからの横やりでひっくり返ることがある。プロジェクト・マネージャ数人に話を聞くと、「昔こんなことがあった」と過去のプロジェクトの失敗談について教えてくれるが、同じような経験をしている人が多い。

プロジェクトで問題を起こさないように進めるためには、リスクに対して「どれだけ鼻が利くか」が重要だ。ただし、これは経験に負うところが大きい。過去に似たような経験をした人は、その経験を基に何を疑うべきかをよく理解している。経験と勘だけに頼るマネジメントは良いマネジメントとは言えないが、少なくとも経験は非常に重要である。こういった経験を蓄積し、統計を取ったり、「べからず集」を作ったりして、次に失敗しないための施策を実行している企業も多い。

失敗に学ぶことは、失敗を繰り返さないためにも大切なことである。他の失敗プロジェクトの事例を知ることで、自分のプロジェクトに対する引き締め効果もあるだろう。

本書では、上流工程における58件の問題事例を公開する。これらは実際のプロジェクトで起こった事象を収集したものである。プロジェクトの詳細な経緯や個別の情報は掲載してはいないが、特徴的な問題事象を取り上げて説明した。

特に、問題の発生が予測されたとき、あるいは問題に出くわしたとき、プロジェクト・マネージャがその問題に対して、「リスクを内在させたまま“見切り”を行ったのか」という点に注目した内容にしている。上流工程においては、ゴールが曖昧なままでも、納期は決まっているのでなんとか作業を進めなければならないこともある。そのときに、どういう判断をしたら、どうなってしまったのか。そのときに本来どうするべきだったのか、を各事例で紹介する。

事例集のイメージは**図表3-25**の通りである。各項目についての概要説明は**図表3-26**を参照してもらいたい。また、**図表3-27**に、収録した58事例のタイトル一覧を示す。

3.4.2 問題発生の傾向と対策

上流工程に関して収集した問題事例や、プロジェクト見える化部会の委員が携わったことのあるプロジェクトで発生した問題には、いくつかの典型的なケースが見られた。その問題発生の傾向と対策を分類したのが**図表3-28**である。主に、①マネジメント、②要件定義と開発範囲、③設計・構築技術、④ステークホルダー、⑤モチベーションに

かかわる問題が典型例として浮かび上がった。

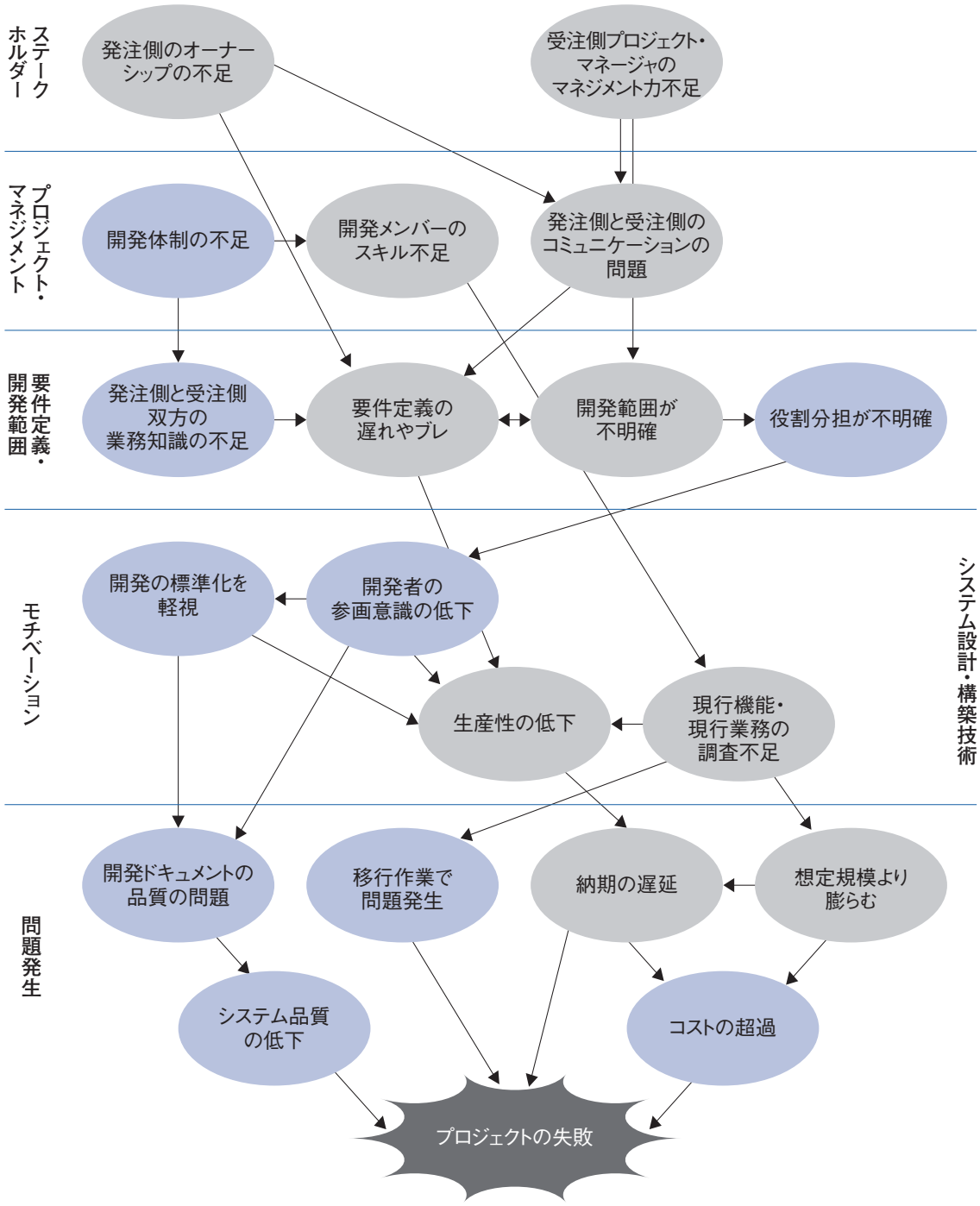
ここで記述した問題や現象は、それぞれに因果関係がある。これを整理して図示したのが**図表3-29**のリスク原因結果分析図である。

この図では、問題カテゴリを階層的に配置することにより、プロジェクトが

図表3-28●問題のカテゴリーとパターン

分類	問題のパターン	対策の傾向
プロジェクト・マネジメントの問題	プロジェクト・マネージャのスキル不足や経験不足に起因する問題。たとえば開発者の進捗管理ができていなかったり、規模の見積もりが甘かったりすることによるマネジメント不足など	作成する成果物の定義や、実施工程の終了基準、そこまでに必要な工数に見通しをたてて、関係者間で共通認識にしていくことが重要である。 プロジェクト・マネージャの交代や分業を上流工程で行うと、プロジェクト全体としてほぼ振り出しに戻る覚悟が必要。進行を是とするなら、手続き面ではスタッフワークを担うメンバーを補うことでリスクを軽減できる。終了基準や成果物定義については、有識者のレビューを適宜実施して、遅れの検知や見積もりの精度アップを図るなどの予防策をとる
要件定義など開発範囲にかかわる問題	要件定義がなかなか確定しないことによって開発工程が遅れるケースや、要件定義が曖昧だったり矛盾があったりすることで、システム仕様に問題が埋め込まれて後工程で破綻をきたすこと	上流工程にて、そもそも到達すべきところまで到達できていないということなので、要件定義の記述方式の設定や、別視点からの要件定義内容の検証方式を導入するなどの方法論の導入が重要となる。 標準開発工程との照合による検討プロセスのチェック、作成した成果物間の整合性チェックなどをタスクに追加して不完全な要件定義のリスクを回避する。タスクを増やすことでさらに工程完了が遅れることになるが、後工程におけるプロジェクトの破綻回避のためであることを関係者と調整して、上流工程の期間延長を受容する
システム設計・構築技術にかかわる問題	アーキテクチャ設計やパッケージの利用、移行方式の検討不足などに起因する問題。現行機能や現行業務の調査不足が引き金になることも多い。開発側のスキル不足も品質低下などにつながる	システム構築の専門家としてサービス提供のために十分な情報、調査、ノウハウが不足した状態でプロジェクトを進めなければならないので、不足分をいかに迅速に認識して、かつ補っていくかが重要になる。 不足分が明確な場合は、体制面の補強を行って不足を補う場合が多い。現行機能や業務の調査については、必要十分な判断を組織的に行うことや、工程に合わせた調査を継続する
ステークホルダーにかかわる問題	顧客側のプロジェクトに対するオーナーシップの不足、重点ステークホルダー（たとえば役員や現場の利用者）への対応不足による問題	顧客（発注）側の体制やプロジェクトの位置付けによって、意思決定への推進力が乏しくなる場合があるので、システム部門やベンダーからの提案型の進め方が重要になる。 発注側キーパーソンとの直接交渉、発注側で検討成果をどのように取り扱うかを含めた提案支援を行う必要がある。開発体制への内向きのマネジメントに対して、ステークホルダー全体を把握した上での外向きなマネジメントに重点をおいてリスク軽減を行う
モチベーションにかかわる問題	プロジェクト・メンバーのモチベーションの低下に起因して作業品質や生産性が低下したり、セキュリティ問題などの他の問題に発展したりすることがある	プロジェクト参画者のマインドの問題ではあるが、善管注意義務がある程度の前提になっているシステム構築の上流工程では、モチベーションの維持が極めて重要になる。 管理レベルを上げて標準的なパフォーマンスを維持する方法と、コミットメント・レベルを上げて自発的な活動を活性化する方法がよく見られる。いずれの場合もコミュニケーション・レベルの向上に注力する。参画者のマインドの悪化が特定の人物に端を発している場合には、当該者の入れ替えが必要である

図表3-29●リスクの原因結果分析図の例



不調に陥る原因を、因果関係をさかのぼりながら概観できる。逆に、上段の階層から、すなわち顧客のオーナーシップを根幹としてプロジェクトを発足した直後から、プロジェクト・マネジメントを駆使して運営する必要があるということも分かる。技術やマインドの問題も、プロジェクトを進行するうえで優先的にクリアすべきハードルである。これらの問題カテゴリをすべてクリアしていかないと、結果としてプロジェクトの不調・破綻に陥ってしまうということである。

リスク原因結果分析図の中には、主な流れとして、「発注側のオーナーシップの不足→発注側と受注側とのコミュニケーションの問題（情報共有できない）→要件定義が振れる→想定規模よりも膨らむ」という状況の遷移が示されている。また、それらの事象と他の事象がどのように関連しているかも示している。読者が実際に経験しているプロジェクトの現象をこの図と比較すれば、プロジェクトの不調・破綻につながる最悪の道筋を知ることができ、事前に対策を検討できるようになる。

3.5 パッケージ利用時の見える化

現在、ITプロジェクトにおいて、個別のリスク・マネジメントが必要となる可能性が大きいテーマは、①流通パッケージを使う場合、②協力会社を使う場合、③契約事項の3つが考えられる。このうち本節では、現場のプロジェクト・マネージャにとって管理プロセスが一部異なる「流通パッケージを使う場合」に関するリスクの見える化について解説する。

現在、パッケージを利用しないシステム開発を探すのが難しいほど、ほとんどのシステム開発でパッケージ・ソフトを使っている。パッケージを使って開発量を削減することで、コストと開発期間を削減できるからだ。

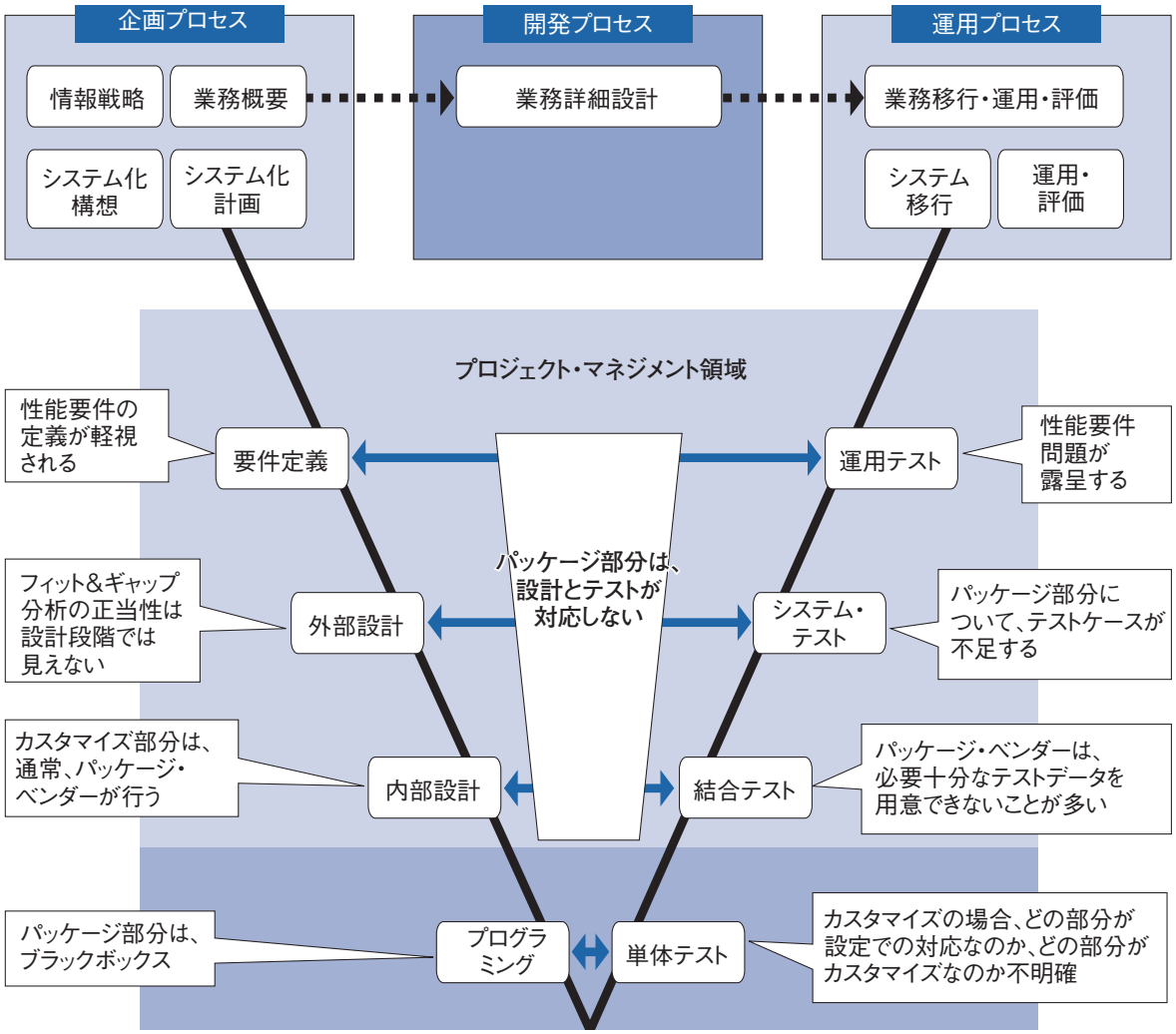
ところが、プロジェクト・マネージャ自身が「パッケージを使うからプロジェクト・マネジメントも楽になる」と誤解していることが原因で、プロジェクトが破綻する事例も少なくない。パッケージ・ソフトの中には、詳細マニュアルが提供されない製品があったり、これまでに開発したシステムを流用した「パッケージもどき」が混在していたり、性能要件がほとんど説明されていなかったりするなど、システムがブラックボックスになりやすい。このため、パッケージが提供する機能について、顧客が正しく理解しないまま、広告やパンフレットの情報をそのまま鵜呑みする可能性も高い。

このような状況であるため、パッケージを使ったシステム開発を成功させるためには、パッケージの内部がブラックボックスであることを前提としたマネジメントが必要とされる（**図表3-30**）。

また、パッケージの中身や設計思想を理解したうえでフィット＆ギャップ分析を行うが、そこではカスタマイズを極力しないために、業務をパッケージに合わせて変えてもらうように導く必要がある。

さらに、受注側（SIベンダー）のプ

図表3-30●パッケージを用いたシステム開発で必要なマネジメント



図表3-31●パッケージ候補を選定するときの「見える化」チェックシート

パッケージ・ベンダー	海外の場合	海外の導入実績	いつから、どれくらい、どこで、どのような業務に、またどのような規模で導入されているか	
			出荷履歴	いつ頃から出荷しているか
			出荷累計	これまでに何本出荷しているか
			導入企業	どこが導入しているか
			適用業務	どのような業務に適用しているか
			導入規模	どれくらいの規模か
		国内の導入実績	日本で使用する場合にローカライゼーションが必要か。日本版があれば、いつから、どれくらい、どこで、どのような業務に、またどのような規模で導入されているか	
			ローカライゼーション	●ダブルバイト化(FEP 導入)が必要か ●法令や慣習に適合させるため、内部ロジックの修正が必要か ●パッケージ・ベンダーはローカライゼーションに積極的か ●プロジェクト開始までにローカライゼーションが完了するか
			出荷履歴・出荷累計本数・導入企業・適用業務・導入規模については、海外での導入実績の各確認項目と同じ	
	国内の場合	導入実績	いつから、どれくらい、どこで、どのような業務に、またどのような規模で導入されているか	
			出荷履歴・出荷累計本数・導入企業・適用業務・導入規模については、海外での導入実績の各確認項目と同じ	
	経営	経営が安定しているか。パッケージのカスタマイズやアドオン開発に積極的か(積極的でも小規模のため、所定の期間内に必要な工数を確保できない場合がある)		
	取り組み	●パッケージのカスタマイズやアドオンに対するスタンスは積極的か ●保守支援能力があるか ●海外ベンダーの場合、コミュニケーションが難しいか		
利用者の評価	このパッケージを顧客、プロジェクトの関係者がどのように評価しているか			
パッケージ販売代理店[*1]	パッケージ・ベンダーとの役割分担が明確か			
パフォーマンス	このパッケージを採用することで、納期、品質、コストにどのような影響があるか			
性能	このパッケージを採用することで、性能にどのような影響があるか			
バージョンアップの影響	プロジェクト期間の前後にバージョンアップの予定があるか。カスタマイズやアドオンにバージョンアップが影響するか			

[*1] パッケージ販売代理店を通さず、ベンダーから直接購入することもある。パッケージ販売代理店経由の場合には、販売代理店の対応力などを見極める必要がある。

ロジェクト体制に、パッケージ・ベンダーやパッケージ販売代理店（パッケージ・ベンダーとの明確な役割分担を考慮する）を必ず加える必要がある。

こうしたプロジェクト・マネジメントの「見える化」のポイントを、①パッケージ候補を選定する時点、②パッケージのソリューションを評価する時点、③プロジェクト体制構築および契約時点において、それぞれ解説する。最後に、リスク対応策の策定と、それに基づくマネジメントを実施する方法を説明する。

3.5.1 パッケージ選定時の見える化

受注側（SIベンダー）がパッケージ候補を選定する際に確認すべきチェック項目には、次のような点がある（**図表3-31**）。

まず、パッケージを提供するパッケージ・ベンダーの実績情報を確認すること。もし、そのベンダーが海外のベンダーである場合は、海外での導入実績だけではなく、国内の導入実績についても確認する。国内導入に関しては、日本固有の問題への適応内容（ローカライゼーション）も確認しておく必要がある。

次に、そのパッケージの販売代理店について、パッケージの内部処理にかかわる不具合が発見された場合の対応方法などを確認する。パッケージ・ベンダーと販売代理店との役割分担も明確にしておく。

パッケージを利用すること、あるいは利用しないことによる、納期、品質、コストへの影響度合いや、性能が出なかった場合の影響、製品のバージョンアップがカスタマイズやアドオンにどのように影響するのかを確認しておく必要がある。

3.5.2 ソリューション評価時の見える化

パッケージ候補を選定した後に、パッケージを利用したソリューションを評価する際のチェック項目には、以下のようなポイントがある。

一つは、パッケージ導入を支援する受注側（SIベンダー）に、そのパッケージでの導入経験があるかを確認する必要がある（**図表3-32**）。導入計画を策定する場合や、パッケージ・ベンダー側の対応内容の妥当性を評価する場合など、当該パッケージの導入経験が重要になる局面があるためだ。

また、パッケージ・ベンダーの体制や、サービス拠点についても確認する。何か問題が発生し、調査や対処が必要なとき、サービス拠点が海外にしかない場合は対処が遅れる可能性がある。

業務内容への適合性、業務が想定している規模とパッケージが想定している規模、ハードウェアの適合性などを

図表3-32●パッケージのソリューションを評価するときの「見える化」チェックシート

発注側（SIベンダー）の該当パッケージの使用経験	パッケージの導入経験があるか。パッケージ導入経験者を確保できるか		
	社内のパッケージ導入経験者をメンバーとして確保できるか	自社内で確保する場合、パッケージ導入経験者が業務適合性分析や機器適合性分析を行えるか	
		協力会社で確保する場合、社外のパッケージ導入経験者を確保できるか。また、そのメンバーが業務適合性分析や機器適合性分析を行えるか	
	発注側が使用しているパッケージからのデータ移行が必要になるか。また、データ移行技術を持つ要員を確保できるか		
パッケージ・ベンダー	パッケージ・ベンダーの要員の定着率は良好か。良好なコミュニケーションができるか。 特に、海外ベンダーの場合 ・適用業務知識を持った要員を確保できるか ・パッケージに関する技術を持った要員を確保できるか ・発注側が使用しているパッケージからデータ移行できる要員を確保できるか		
	海外ベンダーの場合	サービス拠点が国内にあるか	
		サービス拠点が国内になく、販売代理店しかない場合、そこがベンダー機能を代行できるか	
業務適合性（フィット&ギャップ分析）	パッケージの機能がどの程度適合するか		
	パッケージ導入経験者が業務適合性分析を行うか		
	発注側が業務をパッケージに合わせられるか	（Yes）発注側に、現場の抵抗を収めてくれるキーパーソンがいるか	
		（No）カスタマイズが必要か	（Yes）カスタマイズボリュームはどれくらいか。 カスタマイズ部分に、パッケージのバージョンアップが影響するか
		（No）アドオンが必要か	（Yes）アドオンボリュームはどれくらいか。 アドオンのインターフェース要件がしっかりしているか
	既存システムとの連携が必要か	データ連携か	（Yes）データ連携方式はしっかりしているか
		インターフェースによる連携か	連携のインターフェース要件がしっかりしているか。 インターフェース連携に、パッケージのバージョンアップが影響するか
規模適合性	発注側の事業規模やユーザー数に対してパッケージのカバレッジが適正か	パッケージ導入経験者が規模適合性分析を行ったか	
		短期間でパフォーマンスが悪化することはないか （将来の事業規模の拡大・従業員数の増加・処理可能端末の増加などを考慮）	
		短期間でライセンス料の負担が増加することはないか （ユーザー数の増加・トランザクション数の増加・処理可能端末の増加などを考慮）	
機器適合性	・予定されたハードウェア構成で稼働できるか ・パッケージ導入経験者が機器適合性分析を行ったか ・周辺システムとの連携を含むトータルなパフォーマンスを発揮できるか		

図表3-33●プロジェクト体制構築と契約に関する「見える化」チェックシート

受注側の体制確立の要件	該当パッケージの使用経験の程度	パッケージ導入経験者をメンバーとして確保できるか	(Yes)パッケージ導入経験者が業務適合性分析や機器適合性分析を行えるか
			(No)プロジェクト期間中にメンバーがパッケージのスキルを習熟する時間を確保できるか
		これまでのパッケージからのデータ移行が必要になる場合、移行技術を持つ要員を確保できるか	
	カスタマイズに関する折衝能力	カスタマイズの優先順位付けなどによって予定のコストに収めるよう発注側との折衝ができるか	
	代替ソリューションへの対応力	途中でそのパッケージの採用を断念した場合に、代替ソリューションでも推進できる体制を整えているか	
契約にかかわる要件	発注側のプロジェクト方針の明確度	パッケージ導入と同時に業務改革(BPR)を実施するか	トップの業務改革断行の意思がしっかりしているか
			強い権限を持ったプロジェクト専任要員がいるか
		利用部門の協力体制がとられているか	
		発注側との契約において、対象業務範囲が明確に示されているか	
		発注側との契約において、カスタマイズやアドオン量の上限が明示されているか	
		発注側との契約において、業務資料・システム資料(仕様書)が開示することを条件にしているか	
		発注側がパッケージ・ベンダーのテーラリング方法やテスト方法に同意しているか	
		発注側とパッケージ・ベンダーとの担当作業が明確にされ、双方の契約に反映されているか	
	パッケージ・ベンダーとの契約	契約上、パッケージ導入作業の委託範囲が明確になっているか ・パッケージ自体の導入・カスタマイズ・アドオン開発・操作研修の実施など	
		契約上、カスタマイズ・アドオン時のフォロー体制が明確になっているか ・フォロー期間・フォロー要員・作業場所・フォロー時間・休日対応の有無・交通費や宿泊費の負担など	
		契約上、カスタマイズ・アドオン時の作業条件が明確になっているか ・開発言語・著作権の帰属・納品物(プログラム、ドキュメント)など	
		エスクロー契約を締結できるか	
		パッケージのバージョンアップを考慮した契約になっているか	
パッケージ・ベンダーと発注側間の契約にかかわる要件	ライセンス契約と保守契約が発注側の要望に合っているか		
	保守契約のサービス内容、サービス開始時期及び費用が適切か		
契約間の整合性の確認	受注側と発注側との契約、受注側とパッケージ・ベンダーとの契約、発注側とパッケージ・ベンダーとの契約の相互間に不整合がないか		

考慮する必要もある。

3.5.3　体制構築、契約時の見える化

プロジェクト体制を構築する際に考慮すべきチェック項目は、まず、受注側のパッケージ導入経験や、発注側（顧客）との折衝能力など、受注側に求められる要件に関する確認事項がある。

次に、ソリューションを発注側に提供する際の契約要件がある。例えば、その顧客へのパッケージ導入に関して、業務改革（BPR）を含むかどうかを確認する。もしソリューションに業務改革を含む場合、顧客側に求められるのは、トップの業務改革断行の意思と、必要な権限委譲である。当然、導入するパッケージがその目的を達成するために妥当なものかどうか、すなわちパッケージの採用動機が明確になっているかを確認する必要がある。その他、いくつかのチェック項目があるので、**図表3-33**を見てもらいたい。

3.5.4　リスク対応策の策定とマネジメントの実施方法

パッケージを使ったシステム開発において、典型的な問題・リスクと対策案を3つ紹介する。

1つめは「パッケージの選定後に、パッケージ・ベンダーの経営に問題が起こる」という例である。この場合、パッケージ・ベンダーと適切な契約を結んでおけば金銭的なリスクは小さいが、プロジェクトの納期やサポート対応に関してリスクが高くなると考えられる。基本的に、開発サポート条件や違約時のペナルティなどを契約書の条件に盛り込んでおく必要がある。

2つめは「カスタマイズやアドオンの品質が低い」という例。開発スキルが低いことに起因するものではあるが、管理体制を強化すればリスクを解消できる。パッケージ・ベンダーを巻き込んで品質計画を立て直す方法や、定例会議での品質報告、デバッグ体制を明確化する方法などがある。

3つめは、「パッケージに対応可能な技術者が離脱する」という例である。パッケージ・ベンダー側の技術者の場合もあるが、それを利用するSIベンダー側の技術者が離脱する場合も見過ごせない。大きな問題に発展する可能性もあるため、こうしたリスクも想定しておく必要がある。その際、運営体制の改善を図る必要がある。

第4章

定量的見える化アプローチ

4.1 定量的アプローチの概要

自己評価シート、ヒアリングシートなどのツールを用いた定性的見える化アプローチは非常に有効である。ただ、プロジェクトを一層見える化するためには、プロジェクトの実際の動きを客観的なデータで把握することが不可欠である。

なぜなら、ヒアリングだけでは実態の一部を垣間見るにすぎないし、定性的な状況を表しているだけである。定性的なデータを裏づけのあるものにするため、第4章では、要件定義から基本設計にかけて、何のために、どのようなデータを測定すればよいかについて解説する。

例えば要件定義工程の終了時点で、すべての要件が確定していないまま次工程に進んだ場合に、基本設計での手戻りとそれによる工程遅延がどの程度

図表4-1●測定分析データ一覧表（タイム）の例

発生するのか分析する場面があるだろう。このとき、要件の確定状況が「良い」、「ほぼ良い」という定性的な把握では不十分である。「要件確定度が90%で、計画に対して10%遅れているが、ここ1カ月間では20%上昇している」というように、定量的に把握することが必要である。

このような定量的把握により、「最大10%の手戻り発生の可能性はある。要件確定要員を継続確保すれば、あと2週間で要件をすべて確定できる。この程度であれば、人の投入により工程遅延は防げるであろう」といった見通しを立てられる。

定量的測定をする場合、あらかじめ要件確定とはどういう状態のことをいうのか、要件確定度とはどのような式を用いて定義することができるのか、その式に使うデータとして何を測定するのかを決めておく必要がある。同時に、測定の時期・頻度、使い方などを決めておかなければならない。

4.2 測定項目リスト

プロジェクトの状況を定量的に把握するための測定項目を「測定項目リスト」として整理した。

測定項目リストは、2種類の一覧表で構成する。1つは、測定項目とその測定方法、分析方法を整理した「測定分析データ一覧表」。もう1つは、測定分析データ一覧表の各項目を測定する際のベースとなる多様な定量情報を整理した「ベース尺度一覧表」である。

4.2.1 測定分析データ一覧表

測定分析データ一覧表は、プロジェクトの状況をどのような狙いで測定するかという「測定の目的」と、その目的を達成するための測定方法について、知識エリアごとに分類してまとめたものだ（**図表4-1**）。

測定方法は、「導出尺度」と「ベース尺度」で表現している。「導出尺度」とはプロジェクトの状況を見るための項目、「ベース尺度」はその導出尺度を導くための測定要素を表したものとなっている。

例えば、「測定の目的」が「要件定義レビュー作業の進捗を把握し、計画との差異を確認する」である場合、「導出尺度」の1つは「要件定義書のレビュー進捗率」である。この導出尺度を導くためには、「要件定義書のレビュー実施回数や累計時間、レビュー計画書における計画回数や計画所要時間」という定量情報が必要となるが、これらを「ベース尺度」とした。

測定分析データ一覧表は、測定の目

的から導出尺度を選び、それに対応したベース尺度を求める際に利用する。また、「導出尺度の見方、分かること」に記述されている内容から、導出尺度の値、傾向を見て、次工程以降に起こり得る問題を見つけられる。

ただし、測定分析データ一覧表に列挙した項目のすべてを測定することは、プロジェクトを運営していくうえで管理負荷が高く、現実的には難しい。そこで基本的に押さえておくべき測定項目については、測定分析データ一覧の「重点項目」欄に★印を付記したので参照してもらいたい。

しかしながら、1つの導出尺度だけを見ていると、次工程以降に起こり得る問題やリスクを見誤る可能性がある。リスクを的確に把握するためには、複数の尺度や定性的見える化アプローチの判定結果を統合的に評価する必要がある。

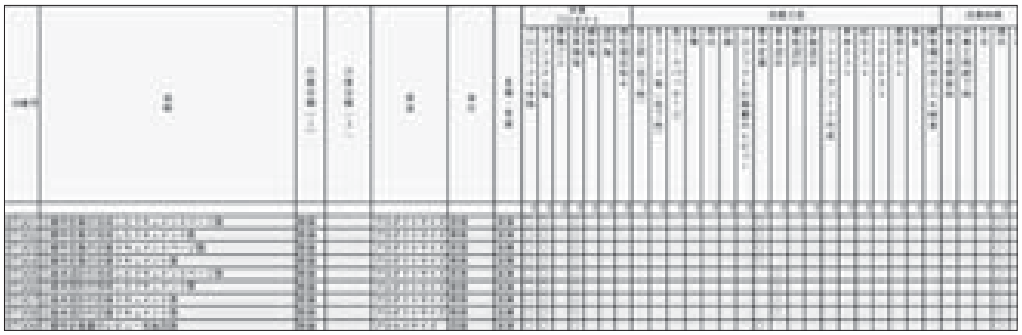
4.2.2 ベース尺度一覧表

ベース尺度とは、実際にデータとして測定する対象項目である。ベース尺度一覧表は、ベース尺度の単位、収集工程・時期・タイミング、収集者を知識エリアごとに分類してまとめたものである（**図表4-2**）。

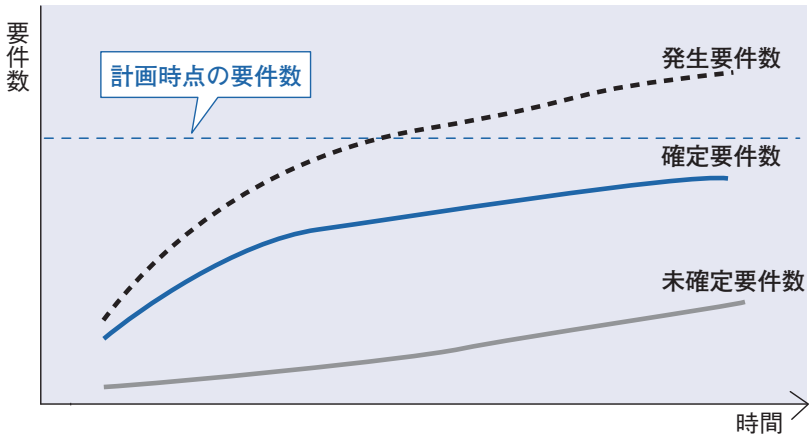
例えば、要件定義作業の進捗状況を見るための導出尺度である「要件定義ドキュメント作成の進捗率」について、測定分析データ一覧表を見てみよう。その定義式は、「完成したドキュメント・ページ数／計画ドキュメント・ページ数」である。定義（式）に用いられる「完成したドキュメント・ページ数」、「計画ドキュメント・ページ数」が実際に測る尺度であり、ベース尺度である（付録にも関連記述あり）。

ベース尺度一覧表を用いれば、①測定プロダクト(粒度：サブシステムごと

図表4-2●ベース尺度一覧表（タイム）の例



図表4-3●発生要件数、確定要件数の推移



や機能ごと)、②測定する工程(要件定義工程や基本設計工程など)、③測定時期（月次、週次、日次など)、④測定タイミング（マイルストーンで定めたタイミングや進捗会議時など）と⑤収集者(プロジェクト･マネージャ、PMO、品質管理者など）を求めることができる。

定量的測定を実施するためには、あらかじめ見るべき導出尺度を決めるとともに、ベース尺度に関する上記の諸項目を調べて、測定データを蓄積しておく必要がある。また、測定タイミング、収集者はプロジェクトの事情により異なるので注意が必要だ。

4.3 導出尺度の見方と分かることの例

定量的に測定したデータだけで、次の工程で起こり得る問題やリスクを特定することは難しいが、大まかな問題点を見出すことはできる。

測定分析データ一覧表にまとめた「導出尺度の見方と分かること」の例を次に説明する。

4.3.1 スコープに関する例

要件定義工程で、要件数および確定した要件数の推移を測定することは、プロジェクトの全体工期やコストへの影響を考えるうえで有意義である（**図表4-3**)。例えば、計画時点より要件の数が増加している場合、基本設計以降の工程の作業工数や期間が増加し、全体工期、コストを守れなくなる可能性がある。

また、要件の数が途中で急激に増加あるいは減少している場合、顧客の考

える目的、前提条件が途中で変わったことが考えられる。この際には、目的、前提条件の確認が必要である。

未確定の要件が多く残っている場合はどうか。その対応（要件確定）に工数と期間がとられ、基本設計に移れず、工期が守れなくなる可能性がある。

要件定義の後半で確定要件が急激に増加している場合、本当は完全に確定していない要件が多く残っていることも考えられる。つまり、要員を追加投入して、短期間で無理やり要件定義工程を終わらせている疑いがあるからだ。

さらに詳しく見る必要があれば、要件を重要度別に、優先度別にランク分けして、それらの推移を見ることができる。

例えば、要件数が増加する要因のほとんどが、重要度の高い要件数の増加である場合を考えてみよう。スコープを計画の規模に抑えるために、重要度の低いものはスコープから外すことができる。だが、重要度の高い要件数ばかり増えている状況ではスコープから外すことができない。結果としてスコープの規模が増加し、基本設計以降の工程の作業工数、期間が増加し、全体工期、コストを守れないリスクが生じる。

また、要件数が増加する要因のほとんどが、優先度の高い要件数の増加である場合でも同様だ。工期を守るため必要な機能から順次リリースをしようとしても、最初にリリースすべき機能数が増加しており、後回しにできなくなる。つまり、工期を守れないリスクが生じる。

基本設計工程では、機能数、確定した機能数の推移、ファンクション・ポイントなどの推移を見ることにより、大まかなリスクを見つけ出せる。

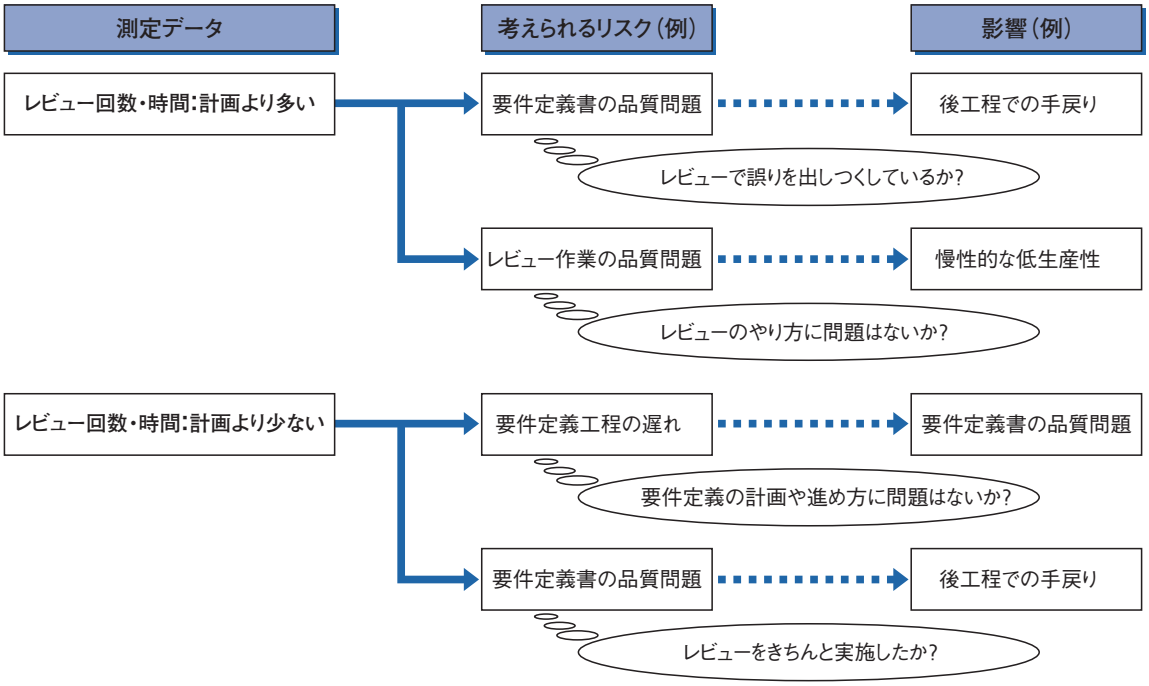
測定する要件の数、機能の数としては、次のものを採用することができる。要件定義書、基本設計書の企業内標準フォーマットがある場合には、その標準フォームの特定欄の項目数を要件の数としてカウントする方法がある。標準フォームにおける項目単位では、粒度がある程度そろっていると考えられるからだ。

さらに、重要度、優先度を利用する場合、その定義を明確にし、顧客の合意を得ておく必要がある。例えば重要度については、「開発するシステムの目的に照らして、それがないと目的を達成できない程度」と定義できる。優先度については「その要求・機能が実現するまでの間、取り得る代替手段の少なさの程度」と定義することができる。

4.3.2 タイムに関する例

要件定義工程で、妥当性のある要件にまとめ上げることは、以降の工程に

図表4-4●要件定義書レビューの進捗



おける中間成果物や製品の品質を高めるだけではない。手戻りを少なくして、コストや工期の目標達成に大きく貢献する。そうするためには、要件定義書のレビューを確実に実施することが肝要である。その進捗を定量データから見る例を6つ紹介しよう。

（1）要件定義のレビューの回数や累計時間が計画より多い場合は、要件定義書の品質が元々悪い可能性が考えられる（**図表4-4**）。レビューで十分な品質レベルを確保できていなければ、基本設計以降の工程で手戻りが発生し、コスト増加やスケジュール遅れにつながる。レビューにおける誤りの検出や修正の状況、再レビューの状況（手法、かけた時間、参加した人、検出された誤りと修正、終了時の状況など）を調べ、誤りが出つくしていると考えられるならば、品質問題が残るリスクは小さいと考えられる。

一方、レビューのやり方が非効率なことが原因となっているとも考えられる。この場合は、プロジェクトのスケジュールやコストに慢性的な悪影響を与えるので、レビューの手順書の改善や

研修などの対策が必要になる。

一般に、ドキュメントの品質が悪くてレビューに時間がかかったり、レビューが繰り返されていたりする場合には、多少のスケジュール遅延が発生しても、しっかりしたレビュー完了基準に至るまでレビューを繰り返すほうがよい。そのほうがドキュメントの品質が向上して下流工程がスムーズに進み、プロジェクト全体としてスケジュールの遅れを最小化できる場合が多い。

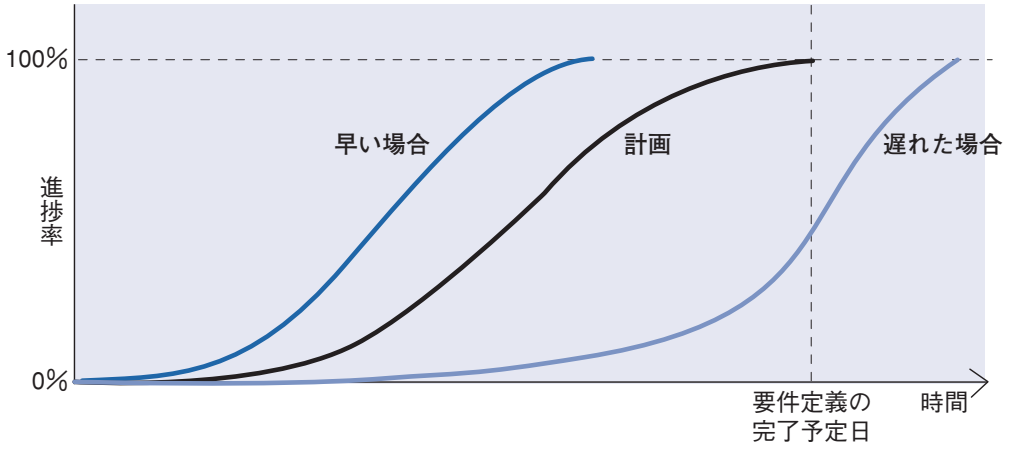
前述の例とは逆に、(2) 要件定義のレビューの回数や時間が計画より少ない場合を考えてみよう。レビューを行った範囲（ドキュメントの数やそのページ数など）が計画より少なければ、要件定義そのものが遅れていると考えられる。

要件定義や基本設計で作成したドキュメントのページ数や章の数の推移を見ることでも、次のようなことが分る。

(3) 要件定義の進捗（作成したドキュメントのページ数など）が計画より遅れている場合は、顧客業務の理解不足や顧客とのコミュニケーション不足などにより、要件定義が進んでいない可能性がある（**図表4-5**）。スケジュールを挽回するため、要件の検討が不十分なまま最後に一気に決めようとすると、以降の工程で要件の見直しが発生するリスクが生じる。

このとき、要件定義のレビューが、適切な人員と時間をかけ、チェックリストなどの標準を用いてきちんと行われていれば、このリスクは小さくなると考えられる。また、要件定義の参加者

図表4-5●要件定義の進捗



の基準や、決めるべき項目の標準を作っておくと、このリスクを小さくすることができる。

⑷ 反対に、要件定義の進捗が計画より早い場合を考えてみよう。新規システムの場合や担当者の経験が少ない場合などにおいては、要件を見落としている可能性がある。要件の再検討や基本設計時の設計漏れが発生することにより、品質の低下やスケジュールの遅れにつながるリスクがある。このとき、計画より早く終わった正当な理由を説明できれば、このリスクは少ないと考えられる。

⑸ 基本設計の進捗が計画より遅い場合は、次のようなリスクが考えられる。要件定義が不十分で基本設計が進んでいない、計画での見積もりが甘かった、基本設計のためのリソースやスキルが不足している、などだ。適切な人員の追加投入が可能ならスケジュールを挽回できるかもしれないが、これがうまくいかないままスケジュールの挽回を拙速に進めると、品質の悪化、さらなるスケジュール遅れ、モチベーションの低下を引き起こすリスクがある。顧客とのネゴシエーションが必要な場合もある。

⑹ 基本設計の進捗が計画より早い場合は、基本設計における検討度合いが不十分で、品質の低下につながるリスクがある。特に、新規システムを開発するときや担当者の経験が少ないときに起こりやすい。進捗が計画より早いとき、その正当な理由を説明できていればリスクは少ない。

4.3.3 品質に関する例

要件定義工程でのレビュー状況、レビュー結果への対処状況を見ることにより、要件定義書の品質がある程度分かる。例えば、要件定義書レビュー時の指摘事項数あるいはページ当たりの指摘事項数が少ない場合、レビューが有効に行われておらず、レビューが不十分な可能性がある。また、要件定義書レビュー時の指摘に対する修正数が少ない場合、あるいは指摘事項対応率が低い場合に、レビュー結果への対処が確実に行われておらず、要件定義書の品質が低いことが分かる。

指摘事項対応率を測る尺度としては、次のものを使うことができる。

指摘事項対応率＝
指摘事項修正数累計／指摘事項数累計

基本設計工程での品質の良しあしは、品質特性に対応する設計がどの程度行われているかで決まる。それを測定するための尺度としては次のものを利用できる。

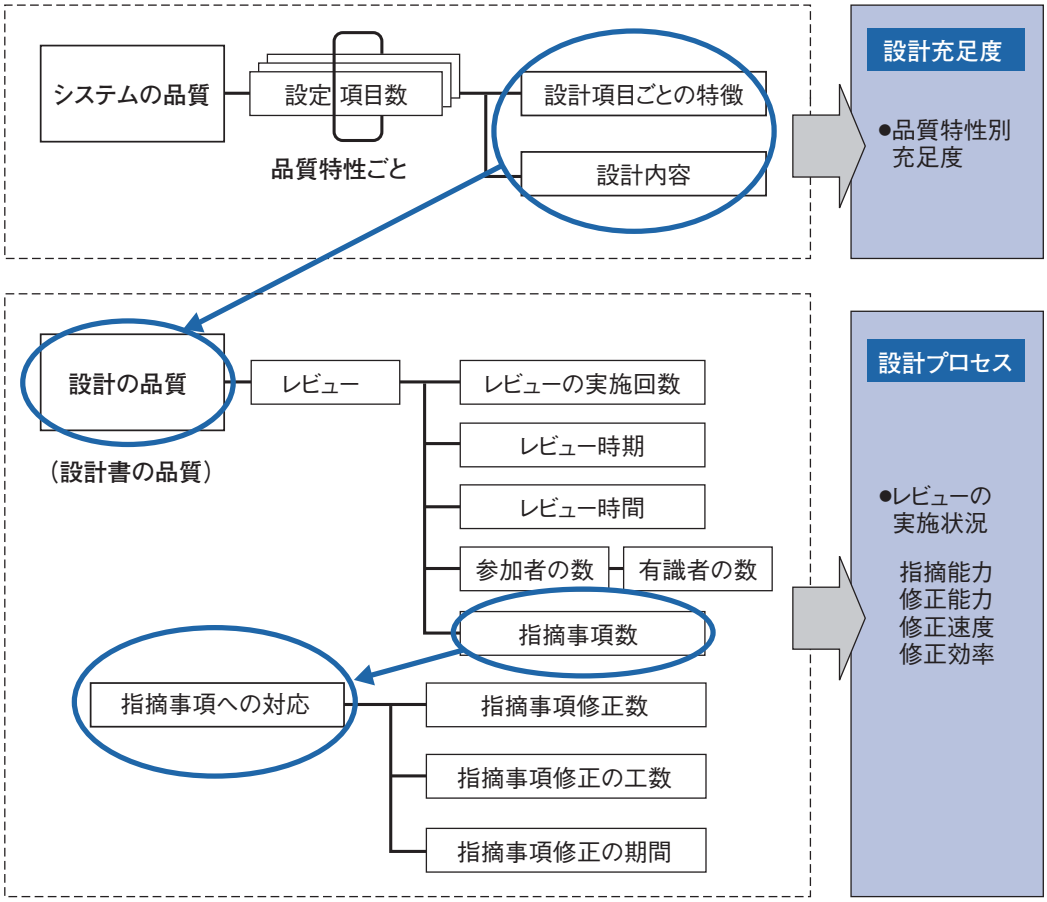
品質＝設計済みの品質特性／設計すべき品質特性（品質特性の充足度）

一般的に品質は、機能性、信頼性、使用性、効率性、保守性、移植性の各品質特性に分類される。このうち機能性を除くと非機能要件がシステムを実現するうえで品質の要素となる。

品質特性を単一の指標で測定することはできず、それぞれの品質特性に応じて設計内容が異なっている。しかしながら、レビューの実施回数や指摘数などは共通の尺度ととらえられる。これらの関係を**図表4-6**に示す。

ここでは、システムの稼働時に問題となりやすい時間効率性、すなわち性能を例として、トランザクション処理の性能設計を行う際の測定例を5つ示す。

図表4-6●品質測定の品質特性と共通的な尺度の関係



まず、(1) 性能設計進捗度（性能設計完了率）の測定について考えてみたい。性能設計を行う場合は、要件定義の段階でトランザクション種別とトラフィック量、トランザクション別の要求応答時間、スループットを洗い出し、エンドユーザーと合意ができていることが前提である。

これらを目標値として設計を行うため、要求が明確になっていない場合は品質目標値がないという状態である。この状態では総合テストを実施したときに、性能が満足できているかどうかを客観的に判断できない。総合テストに至ってから性能要件のすり合わせをすると、性能に関する発注側の想定と受注側の想定との乖離が大きい場合に、性能要件を満たすことが難しくなってしまう。

性能設計を行う場合は、前提条件として、ハードウェアやミドルウェアの能力などを基本情報として押さえておく必要がある。これらの基本情報が押さえられていない場合は、プロトタイプを作成して事前に性能を測定する必要がある。

例えば、データベースに関して測定しておくポイントは、①SQLの処理時間（参照、更新、削除、連結、結合など）、②1件検索時のSQLの処理時間、③複数件検索時のSQLの処理時間、④所要メモリー量、⑤トランザクションの実行時間などである。

次に (2) タスク進捗度を測定する例を考えてみる。性能設計の進捗を測定する場合は、このような作業をWBS（Work Breakdown Structure）に展開し、プロトタイプの設計、開発、ハードウェアの調達、測定項目の完了状況などをWBS上のタスクの消化と連携させて把握する。この場合の進捗度の尺度として、次のものを用いる。

タスク進捗度＝
WBS上の完了したタスク数／タスク総数

ここでは進捗度としているが、必要なWBSをすべて消化しているかどうかは、性能設計の品質も表している。

(3) トランザクション設計進捗度の測定は、まず性能要件、基本情報を確定した後、トランザクション別のデータベース・アクセス回数などをおおまかに見積もり、重点的に性能を確保すべきトランザクション処理を決定する。続いて、トランザクション処理の種類別の設計がどの程度完了したかを測定する。このトランザクション設計進捗度を見ることで、性能設計に漏れがないことを確認し、それがレビューされて見直しが実施され、品質が確保されていることを確認できる。

トランザクション設計進捗度＝設計が完了したトランザクション処理の種類数／トランザクション処理の総種類数

図表4-7はトランザクション別の処理時間の設計内容を示している。例えば、在庫照会処理は平均処理時間が680msと見積もられていることが分かる。

トランザクションごとの平均処理時間を求め、縦軸に平均処理時間をとり、横軸に平均処理時間がかかる順に処理名を並べたのが**図表4-8**である。図中の折れ線グラフは、各トランザクションの平均処理時間が全トランザクションの平均処理時間の合計に占める累積比率を示している。

この累積比率を見ると、処理に時間がかかる上位3つのトランザクション処理（在庫照会、販売伝票作成、販売伝票訂正）が、全体の処理時間の8割を占めていることが分かる。これらの処理の処理方式を中心にプログラム構造やデータベース設計のチューニングを計画する。

このような分析を行うことで、重点的に性能を確保すべきトランザクションを決定し、プログラムの処理方式設計にフィードバックする。

対象とすべきトランザクション処理に漏れがあると、設計時であれば追加設計を行うだけで済む。だが、総合テストで性能設計の漏れが判明すると、

図表4-7●トランザクション別の処理時間例

設計の網羅性を判断

項番	処理名	ピーク時間帯のトランザクション量[Tr/s]	1トランザクション当りの平均処理時間[Tr/ms]			平均処理時間[ms]			平均処理時間合計[ms]
			アプリケーションサーバー	データベースサーバー	データベース	アプリケーションサーバー	データベースサーバー	データベース	
1	在庫照会	2.0	200	100	40	400	200	80	680
2	販売伝票作成	0.8	300	60	60	240	48	48	336
3	会員照会	1.0	100	50	10	100	50	10	160
4	販売伝票訂正	0.4	400	200	90	160	80	36	276
5	配送指示	0.06	100	50	10	6	3	0.6	9.6
6	会員登録	0.04	100	50	10	4	2	0.4	6.4
7	会員訂正	0.0020	150	100	15	0.3	0.2	0.03	0.53
8	入庫予定照会	0.1	200	100	15	20	10	1.5	31.5
9	販売員登録	0.0003	100	50	10	0.03	0.015	0.003	0.048
	合計	4.4023	—	—	—	930.3	393.2	176.5	1,500

システム全体の方式の見直しが必要となる場合がある。この場合、多大な影響を及ぼしてしまう。

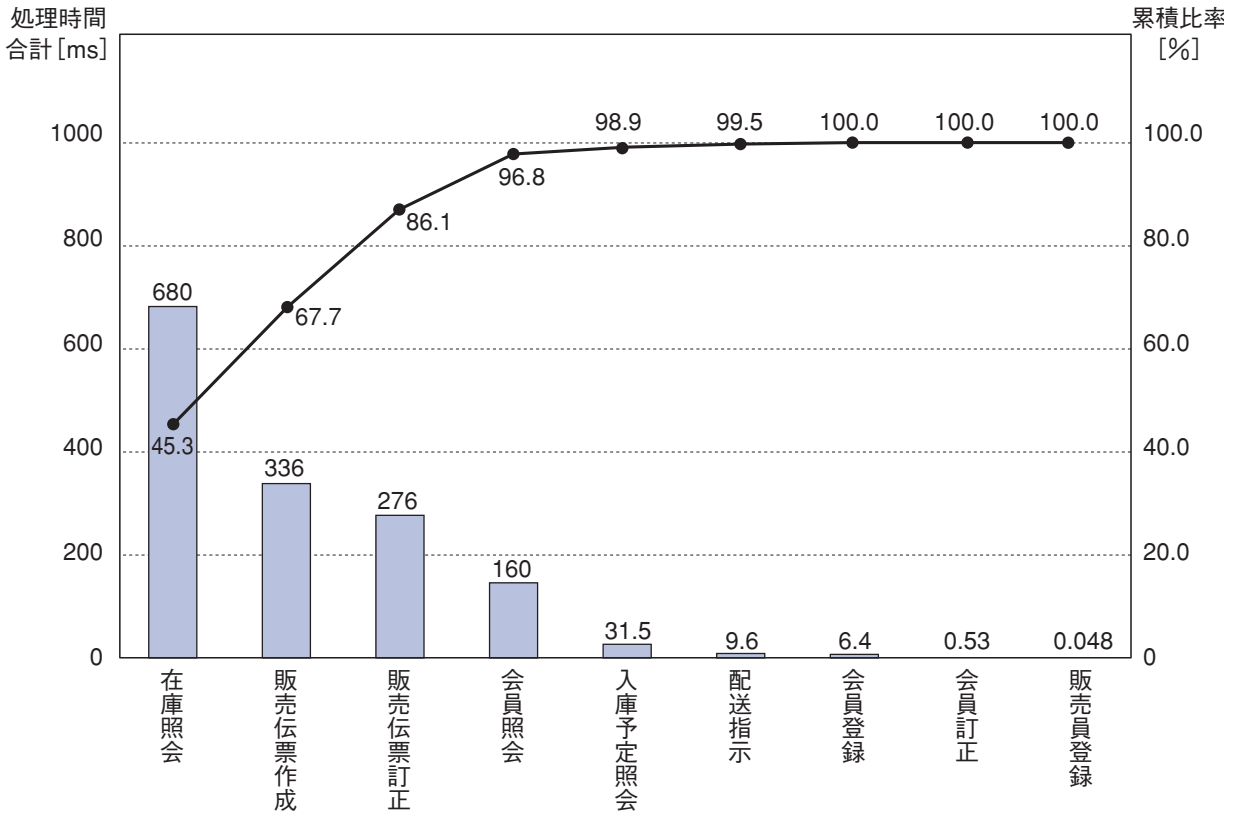
性能設計自体の品質を確保するうえで、対象トランザクションを漏れなく洗い出すことは重要である。総トランザクション数は、要件定義からの機能分割数、画面数、帳票数などから漏れがないことを検証すべきだ。

こうした分析結果に対して、(4) 対策実施率を測定するには、以下の指標を利用する。

分析結果の対策実施率＝分析結果の対策を実施したトランザクション数／分析結果の対策実施対象のトランザクション数

最後に (5) トランザクション別設計品質の測定方法を見てみよう。設計品質を上げるためには、繰り返しレビューを実施することによって、設計上のミスや改善点の洗い出しを行う必要が

図表4-8●トランザクション処理時間のパレート分析



ある。レビューでの指摘事項に対して、どれだけの指摘がされているのか（指摘率）と、指摘内容に対してどれだけ対応しているのか（指摘事項対応率）で、設計品質を評価できる。

指摘事項対応率＝指摘事項への対応数／指摘事項数

指摘率 ＝ レビュー時の指摘事項数／ドキュメント・ページ数

4.3.4 人的資源に関する例

要件定義および基本設計工程におけるプロジェクト体制の強弱(参画したメンバーのスキル、作業負荷)を見ることにより、成果物である要件定義書、基本設計書の品質を推定できる。

プロジェクトの体制の強さが、過去のデータを基に定めた基準値以下であれば、要件定義、基本設計が十分にされていないリスクがある。また、プロジェクト・マネジメントの体制が弱い場合、管理が十分にされていない可能性がある。具体的な測定項目として次のようなものがある。

(1) 要件定義工程の体制の強弱を測る指標としては、次の3つが考えられる。

ITコーディネータ相当のスキル保有者数／要件定義メンバー数

対象業務分野経験者数／要件定義メンバー数

Σひも付け率／要件定義メンバー数(＝要件定義メンバーのプロジェクト間兼務率)

要件定義メンバーの兼務状況を調べておくことは重要である。組織構成上は経験者が確保されていても、他のプロジェクトとの兼務なら、十分な戦力にならない。

(2) 基本設計工程の体制の強弱を測る指標には次のようなものがある。

基本設計経験者数／基本設計メンバー数

対象業務分野の経験者数／基本設計メンバー数

対象プラットフォーム経験者数／基本設計メンバー数

Σひも付け率／基本設計メンバー数(＝基本設計メンバーのプロジェクト間兼務率)

近年、システム構築に用いるプラットフォームの差異により、必要技術、難易度の差は大きい。基本設計におい

ては、対象業務の経験と合わせて、対象プラットフォームに関する経験の重要性が高まっている。

(3) プロジェクト管理体制の強弱を測る指標には次のものがある。

プロジェクト・マネージャのITスキル標準のレベル

プロジェクト管理チーム・メンバーの人数、ITスキル標準のレベル

4.4 定量的管理指標の分析事例

定量的な尺度の利用と、その基となるデータの収集・分析活動はソフトウェア開発の見える化における基本的な実践項目である。ここでは、本書で解説する定量的尺度の枠組みの基となった、奈良先端科学技術大学院大学と日立製作所 情報・通信グループ 生産技術本部との共同研究事例について紹介する。

4.4.1 背景

定量的管理の概念は、プロセス改善のための組織成熟度モデルであるCMM（Capability Maturity Model）やCMMI（Capability Maturity Model Integration）でも大きく扱われている。成熟度レベル3で組織標準のプロセス定義を確立したうえで、レベル4で定量的な開発管理を行うことが想定されている。データの計測と分析を行うことは、定量的管理を実践するうえで必須の活動であり、CMMIではプロセス分野の1つとしてまとめられている。

ここでは、特定の観点から定量的管理を実践するための「指標」を頂点に、それを得るための「測定量」（あるいは尺度とも呼ぶ）による階層構造に基づいた概念を整理した。

定量的管理の実践には2つの大きな課題が存在する。1つはデータ収集の省力化である。例えば、ソースコード行数のように、プロダクトを対象とした定量的計測は、ツールを介してある程度自動的に計測可能だ。しかし、障害件数や工程遅延時間数など、管理的側面の強いデータについては、紙ベースの報告書を基に手動計測されることが多く、収集コストやデータの信頼性がしばしば問題とされる。

2点目の課題は、データに対する正しい理解の提供である。データの収集・分析の目的を理解せずに、形式的な実践が行われることは、最も好ましくない事態である。だが、これを避けるには、利用される各管理指標が目的に対して適切に定義されていることや、管理指標の定義が運用者によって正し

く理解されていることが必要になる。さらには、管理指標のために収集されるデータの定義や目的が、収集者（多くの場合は現場の開発者やマネージャ）によって正しく理解されていなければならない。管理目的そのものが、対象となるプロジェクトにとって必要・適切なものかどうか、あらかじめ検討しておくことも、もちろん必要である。

4.4.2 研究の内容

ここでは、日立製作所 情報・通信グループ 生産技術本部（以下、生産技術本部）がプロジェクト・マネジメントのための指標として試行中である「測定・分析データ一覧」の内容について、奈良先端科学技術大学院大学 ソフトウェア設計学講座（以下、NAIST）と共同で行った分析内容を紹介する。さらに、NAISTで試作された、定量的管理指標運用のための支援ツールEPDG2についても紹介する。

4.4.3 分析対象

生産技術本部では、プロジェクトの見える化やプロセス改善などを目的として、定量的管理指標群を策定し、運用を進めている。指標群の中には、従来、社内で実践されてきた手法を整理したものや、CMMIベースによる組織改善を進めるうえで、新たに提案・試行されているものなどいくつかの種類が

図表4-10●指標番号と利用分類の対応

指標番号	利用分類
#1～#21	進捗管理
#22～#26	レビュー管理
#27～#31	テスト管理
#32～#35	プロセス品質管理
#36～#38	リスク管理
#39～#42	要件管理
#43～#45	支援プロセス管理

図表4-9●組織標準管理指標の一例

#	目的	利用者	目的達成度合を見る指標	指標を求めるための測定データ	データの収集元・方法	データ収集者
2	プロジェクト計画立案状況と実施状況をレビューし、プロジェクト計画の進捗と質を向上させる	プロジェクト管理部門	①プロジェクト計画書のレビュー遅延日数 ②…	①プロジェクト計画書のレビュー予定日、プロジェクト計画書のレビュー実施日 ②…	①プロジェクト受注時にレビュー予定日を収集し、レビュー実施日にその差を求める ②…	プロジェクト管理部門

含まれている。**図表4-9**はその一例である。

これらの指標群から、単一プロジェクト内での定量的管理を目的に定義した45の管理指標を分析対象としている。これら対象指標群の利用目的は、進捗、レビュー、テスト、プロセス品質、リスク、要件、そして支援プロセスの管理である（**図表4-10**）。

本研究では以下のような分析・検討を行っている。

（1）指標定義構造を分析する。各管理指標に対し、ISO15939の測定情報モデルやCMMIのプロセス・エリア「測定と分析」に準じたデータ分類を行い、測定データと分析データ間それぞれを構造的視点から分類する。さらに、測定データと分析データ間の依存関係に着目した整理を行う。

（2）各管理指標の利用実態について、プロジェクト・マネージャを対象に調査を行う。

（3）定量的管理の支援環境を試作する。データ間の依存関係を基に、計測作業の段階的導入や、代替データによる分析、標準データへのスムーズな移行に対する指針を考案する。さらに、測定者、分析者がデータを理解することを助ける仕組みを開発する。

4.4.4　指標定義構造の分析

まず、管理指標リストの内容からベース尺度（基本測定量）と導出尺度を抜き出し、それらのデータ型や、スティーブンズの尺度定義（**図表4-11**）に基づく分類などの情報を補足した。

図表4-11●スティーブンズの尺度定義（参考）

分類	名称	零点	順序	等間隔	連続量	説明
計数値	名義尺度	×	×	×	×	対象を単に区別する目的で付けられる尺度。数の大小には意味がない。例えば、対象業種：{公共、産業、金融、その他}を{0、1、2、3}に対応させたもの
	順序尺度	×	○	×	×	大小関係にのみ意味がある尺度。例えば、要員の経験値：{達人、ベテラン、中堅、初心者}を、それぞれ{4、3、2、2}に対応させたもの。平均値は定義できないが中央値は定義できる
計量値	間隔尺度	×	○	○	○	絶対値よりも数値の差に意味がある尺度。順序尺度の性質も備えている。例えば日付など
	比例尺度	○	○	○	○	数値の差とともに数値の比にも意味がある尺度。順序尺度・間隔尺度の性質も備えている。絶対零点を持つ。例えばコード行数など

データタイプやその他の性質をパラメータ化して分類し、明らかに同一尺度であるとみなされる重複を除去した結果、178のベース尺度を得た。ただし、この中には単位や収集方法が若干異なるだけで、同一種類のものであると思われるものも若干含まれている。今後さらに整理を進める予定である。

これらのベース尺度を尺度分類、用途分類、および、単位の観点から整理したものが**図表4-12**である。一方、導出尺度は、本指標群ではすべて指標に相当するもののみが特定でき、数値（5種類）、サイズ、項目数、割合、日数、工数（人月）、リスト（3種類）、表（7種類）、グラフ（27種類）に大別された。これらの分析を基に、ベース尺度と導出尺度で定めるべき属性を**図表4-13**および**図表4-14**に整理した。これらの属性は本書のベース尺度一覧表

図表4-12●ベース尺度の抽象的分類

尺度分類	用途分類	主な単位
名義尺度	ラベル付け	なし
順序尺度	イベント位置の計測	日付、時刻
比例尺度	イベント回数の計測	回数、件数
	総合的な分析値	金額、無単位数（リスク値、%など）
	プロセス・サイズ計測	項目数、個数、時間、人月、人数
	プロダクト・サイズ計測	行数、個数、ステップ、バイト、FP、頁、無単位数
	列挙型	件数

図表4-13●ベース尺度の属性一覧

名称	ベース尺度の名称
データタイプ	データ形式
尺度分類（1）	数値、尺度、属性、モデルの区別
尺度分類（2）	名義尺度、順序尺度、間隔尺度、比例尺度の区別
見積もり・実測の区別	その値が見積もり値か実測値かの区別
計測対象プロダクト	対象となるプロダクトが存在する場合の詳細
計測対象工程	対象となる工程が存在する場合の詳細
収集者	データの収集を担当する役割
収集方法	収集に当たって利用するツールや手段についての説明
収集のタイミング	データの収集を行うタイミングや工程
バリエーション	類似のデータタイプが想定される場合に、それを列挙したもの

の原型にもなっている。

4.4.5　管理指標の利用実態調査の概要

調査はアンケートを用いて実施した。回答者はそのソフトウェア開発組織において、定量的管理を適用したプロジェクト・マネージャとした。各プロジェクト・マネージャに対して、45の組織標準管理指標に関する実際の利用状況について質問した。

アンケートの前半部分は、開発規模、対象業種、そして回答者の経験としてこれまでの経験月数や経験プロジェクト数に関する質問を並べた。このアンケートの主要部分である後半部分には、対象とした45の管理指標を並べ、回答者にはそれぞれの管理指標ごとに利用の程度を記入してもらった。

ここでの利用の程度の記入とは、「利用した」または「利用しなかった」とい

図表4-14●導出尺度の属性一覧

名称	導出尺度の名称
目的	利用目的
対象プロセス・エリア	この尺度を利用することで効果が得られるプロセス・エリア
利用者今尺度を利用する役割	テスト管理
尺度分類（1）	数値、尺度、属性、モデルの区別
尺度分類（2）	名義尺度、順序尺度、間隔尺度、比例尺度の区別
データタイプ	データ形式
依存するベース尺度	この尺度を求める上で必要となるベース尺度（ID 番号で列挙）

図表4-15●指標利用調査アンケートの選択肢一覧

利用状況	選択肢	意味
利用した	必須指標	同種のプロジェクトでは必ず利用する
	選択指標	プロジェクトの状況によっては利用する
	類似指標	プロジェクトに合うように改変した。あるいは類似の指標で代用した
	その他	その他の理由により利用した
利用しなかった	不要	当該プロジェクトには不要と判断した
	理解不足	指標の理解が不足していたため利用しなかった
	高コスト	利用のためのオーバーヘッドが大きいため利用しなかった
	環境未整備	指標利用のための環境が整っておらず利用できなかった
	その他	その他の理由で利用しなかった

うそれぞれの場合について、**図表4-15**に記載される選択肢の中から選ぶ形式とした。さらに、管理指標ごとに詳細記入欄を設け、利用した／しなかった理由、類似指標を利用した場合はその詳細について、自由に記述できるようにした。

実施したアンケート調査では、最終的に17件のプロジェクトから回答を得た。

開発規模については、これらの業種内で大きな偏りがないように抽出した。17件のプロジェクトの中には、現在進行中のもの含まれていた。回答者の属性は、経験年数が0カ月から25年で、平均は約6.7年。プロジェクト管理経験回数が0回から100回で、平均は約9.1回であった。

4.4.6 利用実態調査の結果

今回の調査対象プロジェクトのうち、ソースコード行数による規模が判明している14件から開発規模の中央値（100万行）を得た。これより大きい7件を大規模（開発規模100万行以上）、中央値より小さい7件を小規模（開発規模100万行未満）に分類した。

さらに、それぞれのプロジェクトを、回答者の経験年数4年未満の「ノービス・クラス」と、経験年数4年以上の「シニア・クラス」に分類して、管理指標の利用状況をグループ間で詳細に比較した。クラス分けによるサンプルの分割数は、大規模プロジェクトにおいてはシニア・クラス4件、ノービス・クラス3件であった、小規模プロジェクトにおいては、経験年数が不明であった1件を除外し、シニア・クラス、ノービス・クラスともに3件であった。

特に顕著な傾向として以下のような特徴が得られた。

（1）進捗管理、テスト管理、要件管理に関する管理指標は、プロジェクトの開発規模や対象業種によらず、ほぼすべてのプロジェクトで利用されている。ただし、進捗管理指標は、月次の進捗会議や顧客との会議の遅延状況を監視するもので、小規模プロジェクトではほとんど利用されなかった。会議実施には遅延が発生しないことが明白であるためだ。逆に、大規模プロジェクトでは、常に利用されていた。

（2）レビューに関する管理指標のうち、「レビューとそれ以降の工程で発生した欠陥数を追跡することによって、レビュー作業の効果を監視する」といった比較的複雑な概念に基づく指標は、現場での理解が進んでいなかった。そのため、ほとんどのプロジェクトにおいて利用されていなかった。

（3）進捗管理に関する管理指標に関して、小規模プロジェクトにおいてはシ

ニア・クラスで類似指標を代用するケースが多く見られた。それに比べ、ノービス・クラスでは利用するかしないかのどちらかに分かれた。これは、経験のあるマネージャはプロジェクトの実情に応じた計測値を用いて、指標を柔軟に設定していることを示唆している。

(4) プロセス品質管理、リスク管理、支援プロセスに関する管理指標の利用状況は、開発規模や対象業種、計画者の経験の違いによる目立った特徴は見られなかった。しかし、プロジェクトの個別事情による差異が顕著に表れていた。例えば、支援プロセスが外部委託されていて、管理されていなかったケースや、プロセス品質管理を行わなかったケースがそれに該当する。

また、プロセス品質管理指標は主に大規模プロジェクトで利用されることが多いが、シニア・クラスのマネージャはこれらを小規模プロジェクトにおいても利用する傾向がみられた。過去に経験した大規模プロジェクトにおいて、プロセス品質保証の意義とその効果を理解しているためと考えられ、興味深い。また、シニア・クラスはノービス・クラスと比べ、リスク管理をより重視する傾向があることも確認された。

4.4.7 調査結果に対する考察

これらの事例は、あくまでも特定の組織で利用されている固有の管理指標群に対する評価である。だが同時に、以下のような一般的と思われる傾向も確認できた。

第1に、プロジェクトによっては、不要な指標や、規模に合わない不適切な指標が存在すること。第2に、経験を積んだマネージャは、プロジェクトの実情に合わせて指標の取捨選択やカスタマイズができることだ。

第3に、指標の運用を進めるには、適切な指標定義と運用者による定義の理解が重要であるということである。

4.4.8 電子ガイドブック/テーラリング支援システムの試作

組織の文化・特性により、測定指標の定義そのままの形では収集・利用が困難である場合や、プロジェクト・マネージャにデータ活用の経験が少ない場合などには、計測活動に関する教育・トレーニングが必要である。ここではNAISTが開発した定量的管理指標のための電子ガイドブック兼テーラリング・システム「プロセスデータガイドブックEPDG」シリーズについて紹介する。

このツールは、日立製作所の管理指標群の定義ファイルを基に作成したXML形式の管理指標定義ファイルと、組織標準WBSの定義ファイルを読み込

み、これらの情報を様々な角度から閲覧し、テーラリング（カスタマイズ）する機能を提供している。開発メンバーは、提供されたプロセス・データ電子ガイドブックを参照することにより、各尺度の持つ性質や収集の手段、実例などを容易に参照できる。

最新バージョンのEPDG2には、管理に用いる指標の取捨選択や、管理スケール（時間的スケールやプロセス／プロダクト・スケール）の調整といったテーラリング機能が追加されている。経験の浅いマネージャに対して、各管理指標の仕組みの理解と実際のプロジェクトへの適用を支援できるようにしている。

定量的管理を開発計画に導入する際には、管理指標の取捨選択、測定量の精度や頻度に関する調整を行った後、開発計画に管理指標利用に必要な測定量の測定・分析活動を組み込む。これらテーラリング作業を行う際には、管理指標およびその指標の利用に必要な測定量の理解が必要である。EPDG2は、管理指標の選択、管理指標利用に必要な測定量の分析・測定活動の開発計画への組み込み、管理指標および測定量の理解などを支援する。

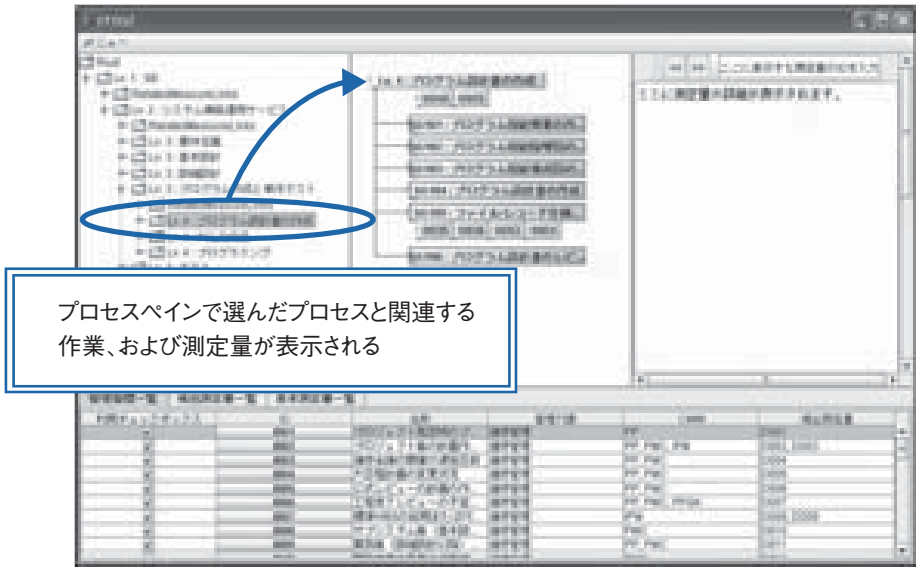
EPDG2はJavaで記述されており、スタンドアローン・アプリケーションとして動作する。また、グラフィック・コンポーネントとして、オープンソースのJGraphパッケージを採用している。

4.4.9 EPDG2の基本操作

図表4-16にEPDG2の画面構成を示

図表4-16●EPDG2 の起動時の画面

図表4-17●対象工程の絞り込み中の画面



す。上段にある①プロセスペイン、②確認ペイン、③詳細表示ペインと、下段にある④指標・測定量一覧ペインの4領域から構成される。

以下に、EPDG2の利用法を簡単に説明する。

(1)起動と定義ファイルの読み込み

EPDG2起動時には、組織標準の開発プロセス定義および管理指標・測定量の定義ファイルを読み込む。そしてプロセスペインには、開発プロセス(WBS)をツリー構造で一覧表示する。指標・測定量一覧ペインには、読み込んだ管理指標・測定量に関する概要を表形式で一覧表示する。

(2) テーラリングの実施

現在、EPDG2によるテーラリングでは、管理指標の取捨選択のみが可能である(今後、指標のカスタマイズに対応する予定)。指標を取捨選択するには、指標・測定量一覧ペインから管理指標一覧のタブを選択し、管理指標一覧を表示させ、利用チェックボックスにより選択する。

(3) WBSと測定作業の参照

プロセスペインに表示されたWBSツリーのノードを選択すると、選択したプロセスとそれに関連する開発作業項目が確認ペインに表示される。**図表4-17**では、プログラム作成時の「プログ

ラム設計書の作成」というプロセスとその関連作業にリンクした測定量を表示している。例えば、「SG105：ファイル／レコード仕様…」というセルがプロセス・作業項目であり、それに関連付けられた測定量が「B035」、「B036」などであることを示している。

(4)管理指標、測定量の定義参照

指標・測定量一覧ペインにおいて、管理指標一覧中にあるIDボタンをクリックすると、**図表4-18**のような管理指標と基本測定量、導出測定量のデータ依存関係が図示される。確認ペインに表示されている測定量を表すマークをダブルクリックすると、詳細表示ペインに当該測定量に関する詳細内容が表示される。詳細表示ペインには、測定量ごとにその測定方法や分析方法、測定担当者などの情報が表示されている(**図表4-19**)。

また、測定のサンプルや関連する測定量などがリンクされており、それらの関連情報も参照することができる。さらに、確認ペインでダブルクリックする以外に、詳細表示ペインの上部にあるテキストボックスに管理指標・測定量のIDを直接入力することで、その指標・測定量の詳細情報を表示することも可能である。

(5)テーラリング・ファイルの保存

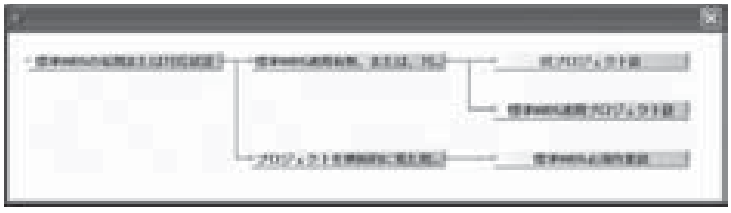
最後に、テーラリングを行ったプロセスをファイルに保存する。保存したプロセスはEPDG2で読み込むことで参照され、再利用される。

4.4.10 期待される効果

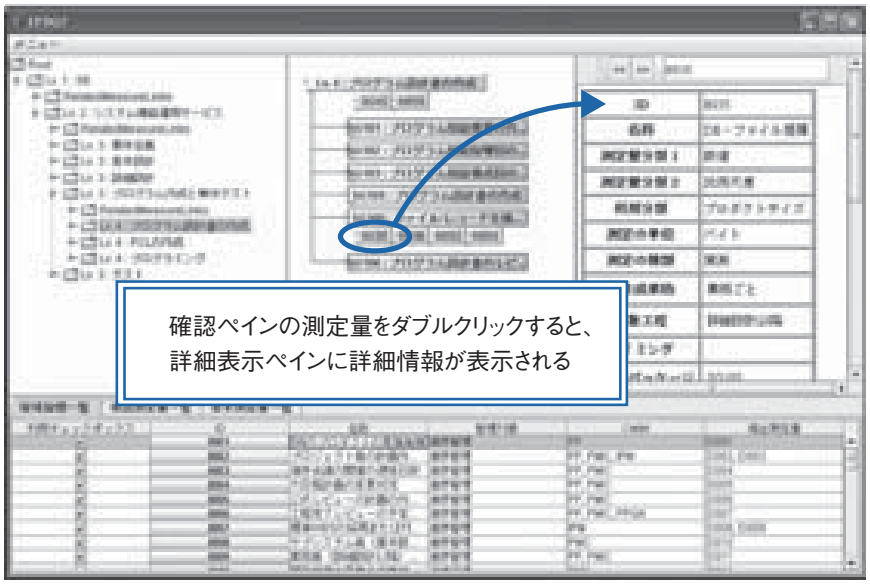
テーラリング作業時にEPDG2がプロジェクトの計画者に対して提供する支援機能は、以下の目的にも役立つと考えている。

まず、管理指標の取捨選択に役立つこと。プロジェクトで適用する管理指標をEPDG2で一覧することができ、その場で容易に取捨選択ができる。このため、一貫した視点に基づいて管理指標を導入できる。

図表4-18●データ依存関係画面



図表4-19●測定量の詳細表示



次に、プロセスに統合された測定作業を適時確認できること。各工程やその各作業で必要となる測定作業の存在を視覚的に確認できるため、計測と管理に対する具体的なイメージを持ってプロジェクトに臨める。

指標の理解も進むだろう。管理指標や測定量の概要および詳細を容易に参照できるため、指標や計測データに対する理解を深めることができ、管理や測定が正しく実践される。

事例の再利用も容易になる。テーラリングに関する情報を保存・収集する機能を利用して、系統的なテーラリングガイドの作成や、テーラリング・パターン・ライブラリを構築し、経験の浅いマネージャに提供することができる。

第5章

統合的アプローチ

5.1 統合的アプローチの概要

プロジェクトを成功させるためには、開発工程のなるべく早い時期（すなわち本書がターゲットとする上流工程）にリスクを発見し、そのリスクが顕在化しないように注意しながらプロジェクトを進めることが肝要である。

プロジェクトには、その特性に応じて様々なリスクが潜んでいる。なかば明示的に「そこにはリスクがあるだろう」と思えるものから、事前に気付くことができず「まさかそれが問題になるなんて」と、後になってからリスクだったことに気付くような場合もある。

すべてのリスクは、それが顕在化した場合に、最終的には品質（Q）、費用（C）、納期（D）のいずれかに影響を及ぼすと考えられる。また、QCDに影響が及ぶほどの問題に至るまでには、なんらかの予兆があるはずである。この予兆に気付くか否かがリスクを発見するうえでの重要なポイントになるが、経験や知識に負うところが多い。

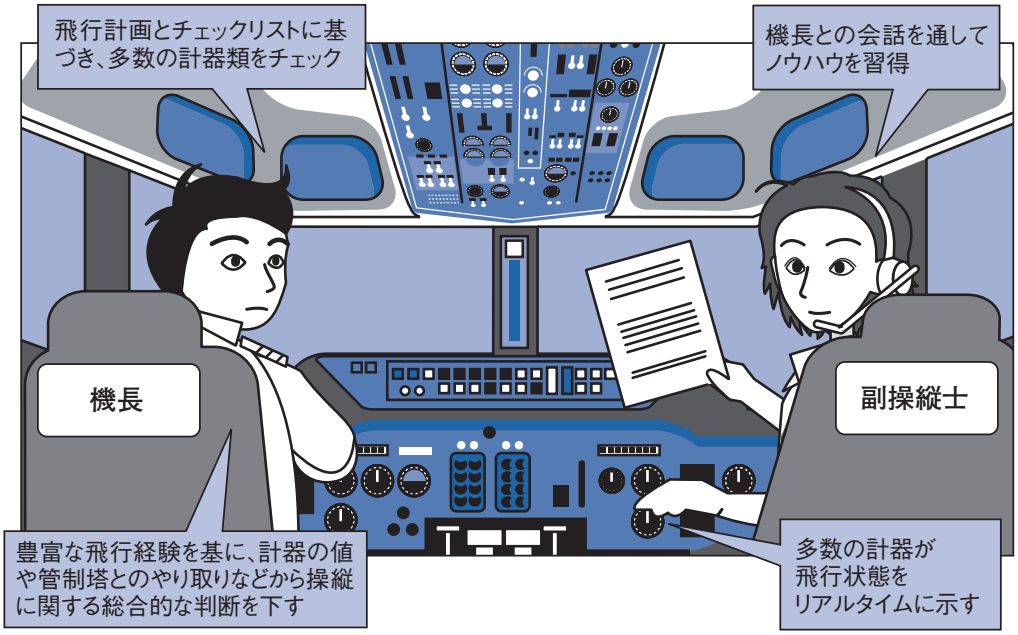
第3章、第4章では、プロジェクトに潜むリスクを特定するために、チェックシートや定量的な測定方法を利用したリスクの「見える化」について説明した。これらは、いわば個別のツールであり、プロジェクトのリスクをそれぞれの側面でとらえる方法の説明である。

この章では、リスクを統合的な視点で特定するための実践的な方法「統合的アプローチ」について説明する。その中心となるツールは、プロジェクトにおいてリスクが潜みやすい事項をまとめた「リスク分類表」だ。リスク分類表は、これまで説明してきた見える化ツールの各項目をひも付けている。

5.2 リスク分類表によるアプローチ

上流工程でプロジェクトのリスクを的確に特定し、対策を打つには、経験豊富なプロジェクト・マネージャによる、非定型で発見的なレビューが欠か

図表5-1●コックピット・ドリルの利用イメージ



せない。上流工程では、プロジェクトが始まったばかりであり、プロジェクトに関する情報収集の仕組みも出来上がっておらず、プロジェクトの今後を十分に見通せないことが多い。そのため、的確にプロジェクトの状況を把握し、リスクを洗い出すには、曖昧で不確実な情報も含め、できるだけ多くの情報を集め、起こり得る事象への対策を検討する必要がある。

その意味で、上流工程を「見える化」する（＝リスクを検出し、プロジェクトの問題を特定する）ためには、事象や情報をできる限り俯瞰し、総合的に判断する「統合的アプローチ」による手法が有用と考えられる。

リスクの洗い出しは、何の根拠もなく、感性や偶然だけで未来を占うといった非科学的作業ではない。網羅性、一覧性を確保し、組織的知見を蓄積するために各種ツール類を準備したうえで、経験の少ないプロジェクト・マネージャでも、上流工程の問題に統合的にアプローチできることを目指す。

上流工程のリスクの検出と対策は、航空機の離陸時の「コックピット・ドリル」にたとえて考えると分かりやすい（**図表5-1**）。パイロットは、離陸前に、飛行計画をよりどころとして、膨大なチェックリストをチェックし、計器類を

図表5-2●リスク分類表

項番	知識エリア	リスク分類	ヒアリングシート	測定分析データ一覧	事例集
1	顧客	顧客予算	H6,H42,H47		6,13,24
2		顧客特性	H2,H3		4,20,24,27,32,38,39,46,47
3		案件特性	H32,H70		23,26,27,35,36,38,39,46,47,56
4		顧客役割分担	H2	Ko1,Ko2,Ko3	2,4,5,20,32,34,46
5	スコープ	開発範囲の取り決め	H32,H34,H52		5,6,23,25,27,41,42,43,44,45,46
6		要求と要件の整合性		S1,S2,S3	3,43,45,52,53,57
7		要件とシステム化範囲の整合性	H1,H4,H5,H6,H33,H34	S5,S6,S7,S8,Q10,Q11,Q14,Q15,Q16	16,17,32,36,43,45,49,52,53,56,57
8		システム化範囲と体制の整合性	H3,H4,H16		2,32
9		システム化の実現可能性	H5,H6,H7,H25,H35,H72	Q14,Q15,Q16	26,31,33,53
10		変更対応	H32,H33,H42,H47	S4,S9	14,15,22,27,43,45,51,55
11	タイム	クリティカルパスの特定	H5,H22	T15	3,12,31,32,44,49
12		実行可能スケジュールの作成	H4,H16,H18,H19,H21,H23,H31	T16	3,5,10,17,18,26,32,44
13		進捗確認	H27,H68	Q10,Q11,Q14,Q15,Q16,T3,T4,T7,T8,T9,T10	49,50,51
14		作業ボリューム推移	H7,H18,H21	T1,T2,T5,T6,Q16	31,49
15		タスクの分解	H16,H19,H31	Q10,Q14	32,46,49
16	コスト	上流工程の見積り妥当性	H46	C1	7,26,43,44,45,46,56
17		基本設計以降の見積もり妥当性	H7,H46,H47,H48	Q10,Q11,Q14,Q15,Q16	26,43,44,45,49
18	人的資源	業務有識者	H28	H2,H6	11,21,43,45,48,54
19		技術系専門家	H29	H1,H5,H7	11,26,28,34,35,58
20		PJ内部体制	H13,H14,H15,H17,H30,H66,H69,H70,H7	H3,H4,H8,H9,H10,Ko1,Ko2,Ko3	10,25,26,28,29,31,48
21	調達	外部調達の妥当性	H66,H67	H3,H4,H8,H9,H10,Ko1,Ko2,Ko3	10,30,46,48

図表5-2●リスク分類表（続き）

項番	知識エリア	リスク分類	ヒアリングシート	測定分析データー覧	事例集
22	調達	パッケージ適用	H32,H73		40,43,45,53,55,58
23	組織	PJ支援体制	H14		26,29,50
24		組織の効率	H14	O6	26,29,50
25	コミュニケーション	PJ内情報共有	H1,H8,H9,H11,H20,H60	Cm2,Cm3,Cm4	25,51
26		PJ外情報共有	H2,H10,H12,H20,H24,H38,H43,H44,H53	Cm1	17,38,43,44,45,46
27	品質	品質計画	H26,H49,H50,H51,H68,H73	Q10,Q11,Q14,Q15,Q16,Q17,Q18,Q19	17,38,43,44,45,46
28		品質管理	H9,H40,H41,H49,H51,H56,H68,H72,H73	T11,T12,T13,T14,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11,Q14,Q15,Q16,Q17,Q18,Q19,O1,O2,O5	30,50
29		品質保証	H9,H40,H41,H49,H51,H56,H68,H72,H73	Q1,Q12,Q13,Te1	30,50
30	技術	未経験技術	H74	Te2	14,26,28,31,33,48
31		標準化検討	H26,H40,H41,H52	Q13	12
32		システム実現方式検討	H25,H35,H54,H56,H57	Te1	26,31,33,35,46
33		移行・パッケージ	H55,H73,H74	Te1	30,33,34,37,39,47,54,55,58
34	基本動作	作業標準	H40,H58,H59	O1,O2,O5,O7,O8	1,8,9,24,25,43,45
35		セキュリティ順守	H39	O1,O2,O5,O7,O8	1,8,9,24,25,43,45
36	モチベーション	人材活用	H61,H62	M1	
37		当事者意識	H15,H60,H62,H63,H64,H65	M2	
38	リスク	リスク識別	H36	R1,R2	31,43,45
39		リスク分析・評価	H37	R1,R2,R3	26,27,45
40		リスク対応策	H38	R1,R2,R3	27
41	課題	課題認識	H43,H44	K1,K2,K3	27
42		課題解決	H45	K1,K2	27

確認し、安全を確認してから離陸操作を開始する。このときパイロットは、個別のチェックを行うだけでなく、先輩パイロットのノウハウを蓄積した講義・シミュレータ・実機による訓練や長年の経験から、総合的にフライトの実現可能性と安全性を確認している。

プロジェクト・マネジメントも同様である。プロジェクト計画をよりどころとし、まずチェックシートや測定データでプロジェクトの状態を定性的・定量的に把握する。続いて、事例集でベテラン・プロジェクト・マネージャの貴重な経験に触れ、統合的にプロジェクト状況を確認する。こうした作業は、プロジェクトが上流工程を乗り切り、成功へのスタートを切るために欠かせないものである。

5.2.1 リスク分類表

上流工程のリスクを統合的アプローチで発見するためのツールとして、「リスク分類表」を用意した（**図表5-2**）。リスク分類表は、見える化のための3つのツール（ヒアリングシート、測定分析データ一覧表、事例集）を連携させ、統合的な視点でリスクを洗い出すためのツールである。チェックシートを用いた定性的な視点、測定分析データ一覧を用いた定量的な視点、事例集を用いた類似事例からの視点で、プロジェクトで生じやすいリスクを探ることができる。

リスク分類表の縦軸は、上流工程で生じやすいリスク項目をリストアップし、知識エリアで分類したものを一覧にしたものである。この知識エリアによる分類は、多くのプロジェクトを推進した10人以上のプロジェクト・マネージャが議論して、経験的に導いたものだ。各企業の現場で使われているリスク・チェック項目も加味して作成している。

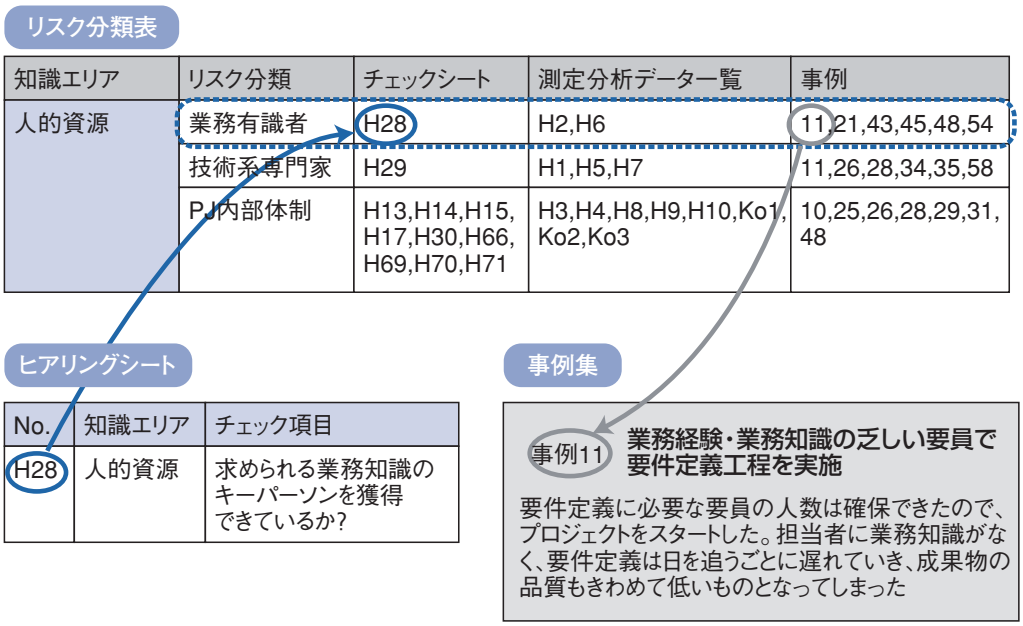
5.2.2 統合的アプローチの活用法

リスク分類表を用いて、統合的に上流工程でのリスクを洗い出し、的確にプロジェクトを導くための活用例を3つ紹介する。

(1)プロジェクト・マネージャへのヒアリング結果を基に、類似の過去事例を参照してリスクを検討する

PMOのような専門家チームがプロジェクトの状況を把握する場合、最初にヒアリングシートを使ってプロジェクト・マネージャにヒアリングすることが多い。この場合、ヒアリングした結果は、あくまでもプロジェクト・マネージャの意見や、表面的な現象から分かる状況を確認したものなので、さらに踏み込んだ調査を行うのが一般的である。

図表5-3●リスク分類表活用例（1）：ヒアリングシートから事例を検索する



見識の深い専門家であれば、ヒアリングの内容から判断して、どの部分をさらに踏み込んで調査すべきかを考えることができるだろう。

しかし、見識の深い専門家が、常にヒアリングに参加できるわけではない。経験の浅いチームによってヒアリングが行われた場合、リスクが顕在化するとプロジェクトがどのような状況に陥るかを具体的にイメージできないことがある。具体的なイメージがなければ、その後に十分な調査が行なわれず、結果として重要なリスクや問題を見過ごすことにもなりかねない。

そこで、ヒアリングした結果を基に類似の事例を参照してイメージをつかめるように、リスク分類表を利用する。

例えば、ヒアリングを実施した結果、ヒアリングシートの項目番号H28「求められる業務知識のキーパーソンを獲得できているか？」というチェック項目に問題があったとする。この「H28」をリスク分類表の中で探すと、同じ行に、関連事例がいくつかマッピングされていることが分かる（**図表5-3**）。例えば事例番号「11」は、「業務経験・業務知識の乏しい要員で要件定義工程を実施」という事例だ。業務知識のキーパーソンを獲得できていないと、どういうリスク、問題が起こるか過去の事

例から知ることができる。

(2)ヒアリングシートからリスク分類表を参照し、測定項目を調べ、定量的に評価する。定量評価が行われていない場合は測定を開始する

活用例(1)と同様に、初めにプロジェクトの状況を把握するためにヒアリングを実施する。次に、ヒアリングで見えてきたリスクを定量的に確認するために、測定分析データ一覧を参照して、具体的なプロジェクトの定量指標を確認する。ヒアリングシートのみによる確認では、主観的な判断が混じりやすいが、併せて定量的な測定結果を参照すれば、そのプロジェクトに潜むリスクを的確につかめる。

例えば、活用例(1)でも説明したヒアリングシートのチェック項目「H28」に対し、リスク分類表を見ると、いく

図表5-4●リスク分類表活用例(2):ヒアリングシートから定量測定項目を検索する

リスク分類表

知識エリア	リスク分類	チェックシート	測定分析データー覧	事例
人的資源	業務有識者	H28	H2,H6	11,21,43,45,48,54
	技術系専門家	H29	H1,H5,H7	11,26,28,34,35,58
	PJ内部体制	H13,H14,H15,H17,H30,H66,H69,H70,H71	H3,H4,H8,H9,H10,Ko1,Ko2,Ko3	10,25,26,28,29,31,48

ヒアリングシート

No.	知識エリア	チェック項目
H28	人的資源	求められる業務知識のキーパーソンを獲得できているか?

測定分析データー覧

項目番号	知識エリア(主)	知識エリア(関連)	測定の目的	利用者	導出尺度
H2	人的資源		要件定義工程における体制の強弱を測定する	PM、PMO	要件定義工程における、対象業務分野(小売業とか保険業とか)の経験者率(対象業務分野経験者数/要件定義メンバー)

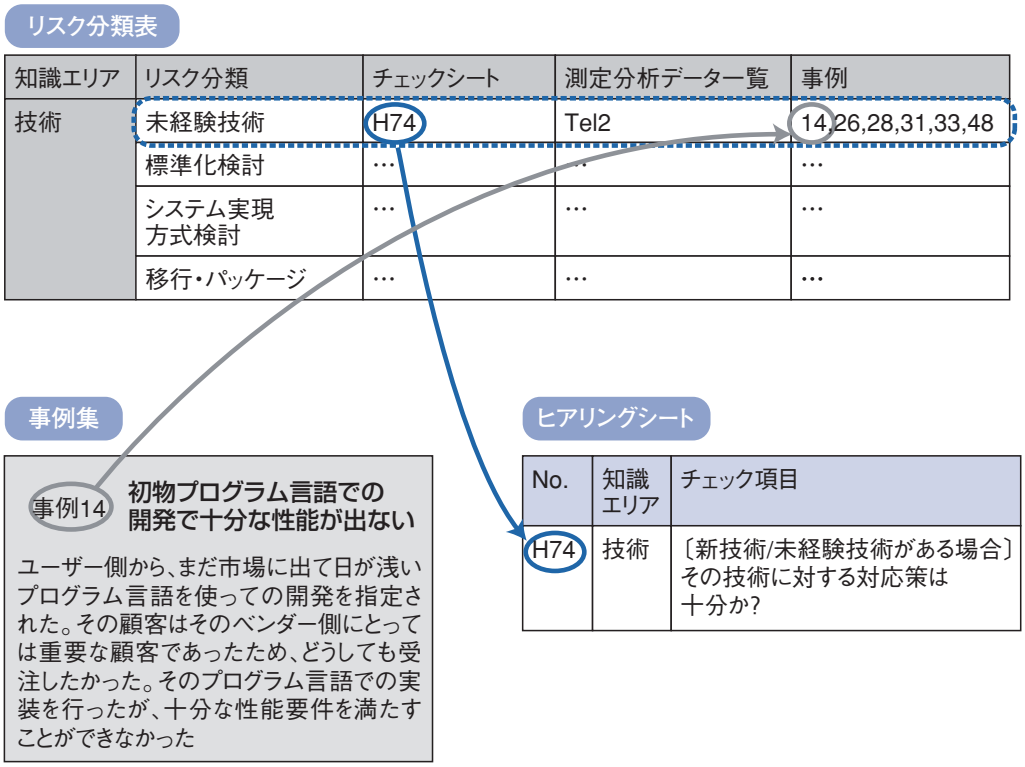
つかの測定項目がマッピングされていることが分かる（**図表5-4**）。そのうちの1つである「H2」は、「要件定義工程における体制の強弱を測定する」という測定項目だ。「求められる業務知識のキーパーソンを獲得できているか？」というヒアリングシートの質問に対し、定量的なデータで裏付けをする方法を示している。具体的には、要件定義工程における対象業務分野での「経験者率を求める数式」と、その測定のための測定項目（経験者数と要件定義メンバー）を指し示している。

その時点で定量的なデータを収集しておらず、定量的判断ができない場合もある。しかし、それからでも遅くないので、定量的なデータの収集・分析方法を計画する機会と考え、定期的・継続的に測定していくことが望ましい。

(3)事例集から現在の状況に類似したプロジェクトを見つけ、チェックシートでプロジェクトの状況を確認する

プロジェクト・マネージャが本書の

図表5-5●リスク分類表活用例(3)：事例集からヒアリングシート項目を検索する



事例集を読むと、自身が参画しているプロジェクトに似た状況の事例を見つけ出すかもしれない。または、プロジェクトの進行中に、ふと、目の前で起きている現象に似たような事例が、事例集の中にあったということを思い出すかもしれない。

そのようなとき、チェックシートや測定分析データ一覧にある関連項目をたどれば、より広い、統合的な視点で自身のプロジェクトの状況をつかめるようになる。

例えば、自らが手がける新規プロジェクトで、あまり経験のない新しいプログラム言語を使う計画があるとしよう。事例集を探すと、事例「14」に「初物プログラム言語での開発で十分な性能が出ない」がある。事例14に対応するチェックシート項目は、リスク分類表から「H74」の「その技術に対する対応策は十分か？」であることが分かる(**図表5-5**)。事例やチェックシートに記載されている詳細な情報を見れば、どのような対応を打つべきかを検討できるようになる。

5.2.3 リスク分類表の高度な利用法

リスク分類表は、一度作って利用するだけのものではない。プロジェクトを取り巻く環境は、常に変化し続けている。現実のプロジェクトを運営することによって得られる情報とノウハウを整理・蓄積し、より精度が高く、より早い時期にリスクが特定できるようにリスク分類表をブラッシュアップすることが肝要だ。そうすれば、プロジェクトの成功率は一層高まるだろう。

リスク分類表は、リスクが現れやす

図表5-6●リスク分類表の拡張

知識エリア	リスク分類	チェックシート	測定分析データ一覧	事例	独自ツール
技術	未経験技術	H74	Te2	14,26,28,31,33,48	A01,A02
	標準化検討	…	…	…	…
	システム実現方式検討	…	…	…	…
	移行・パッケージ	…	…	…	…

新たな列を追加することで
リスク分類表を拡張利用できる

い項目と各種ツールの要素を緩やかにマッピングしたものであり、各組織で個別にカスタマイズしてもよい。例えば企業の個別事情に依存する新たなチェック項目を追加したとして、そのチェック項目をリスク分類表にマッピングしたり、組織の事情に合わせて既存のマッピングを変えたりしてもよい。組織あるいは業界ごとに、このマッピングを見直すことで、より高い分析精度が期待できる。

また、各社独自の見える化ツールを作成した場合、このリスク分類表に新たな列を追加して、その独自ツールの項目番号をマッピングしてみると、チェックシートや測定分析データ一覧、事例集と、独自ツールを連携させることができる（**図表5-6**）。

第6章

曖昧性と不確実性のマネジメント

「プロジェクト」は、システム開発をはじめ、起業、経営、ビジネス、サービス、製品開発・製造、研究開発などあらゆる領域の「変化／変革」を実現するための優れた手法である。しかし、同時にプロジェクトの本質が、「変化／変革」を実現することであるがゆえに生じる数多くの問題を抱え込んでいる。

この章では、ITプロジェクトが他のプロジェクトと比べて、曖昧性と不確実性が圧倒的な量と質で存在するということを理解したうえで、曖昧性と不確実性のマネジメント方法を説明する。

6.1 プロジェクトの曖昧性と不確実性

ITプロジェクトには、常に多くの「曖昧性」が内在している。関与するステークホルダーが多く、成果物を言語や図で表現することに限界があるからだ。さらに、近年の発注側のビジネスの変化が激しく、発注者自身も新しいビジネス・モデルを読みきれないことからくる「不確実性」が内在している。しかも、その質と量は、他業界、他分野と比べても圧倒的に多い。このことがITプロジェクトのマネジメントを難しくし、ITプロジェクトの成功率が低い原因となっている。

ここでいう「曖昧性」とは、知識が不足していて分からないこと、解釈が何通りもあって分からないこと、誤りが含まれていて正確でないことなど、複数の状況を指す。これは、文章、図およびコミュニケーションが持っている本質的な欠陥であり、プロジェクトの失敗原因となり得る。

一方、「不確実性」とは、主に環境の変化によってもたらされ、想定したことが起こるかどうかがその時点では分からないことを意味する。

ITプロジェクトにおいて、曖昧性と不確実性が存在する個所の例を**図表6-1**に示す。

この「曖昧性」と「不確実性」を内包しているプロジェクトを成功させるには、プロジェクトを取り巻く状況を的

図表6-1●ITプロジェクトにおける「曖昧性」「不確実性」が潜む個所の例

項目	曖昧性	不確実性
プロジェクト	●ステークホルダー間の視点、価値観 ●対象とするシステムの特性と関係性の理解 ●プロジェクトの位置付け	●潜在的な問題の数と質が確定できない ●IT技術、プロダクト品質などの変動
マネジメント	●要件定義、品質管理などの非定量性 ●発注側のプロジェクト・マネジメントと受注側のプロジェクト・マネジメントのレベルの相違や整合性	●プロジェクトの目的自体の変化 ●要求仕様、人的質・量の変化 ●ステークホルダー間のコミュニケーション量
IT技術	●ビジネス/サービスなどの利用形態の多様性 ●ビジネス/サービスなどのモデル/業務仕様、要件定義の表現	●外部環境の頻繁かつ俊敏な変化 ●モデル/業務仕様/要求仕様の変化 ●IT技術の非連続的な進化

確に認識したうえで、ドミナント・アイテムを把握し、そこに焦点を絞ったマネジメントと複数の選択肢を活用するマネジメントが有効である。

6.1.1 「ドミナント・アイテム」中心のマネジメント

プロジェクトは、その実行過程において表面化しないものを含めると、様々な問題が発生する。プロジェクト・マネジメントは、顕在化あるいは潜在化している問題を取捨選択して解決し、プロジェクトに課せられた変革の目的を達成することにある。

そこでプロジェクト・マネージャが最も避けなければならない状況は、プロジェクトの成否にかかわる本質的問題（ドミナント・アイテム）を見失って、プロジェクトを失敗させてしまうことである。

ドミナント・アイテムの数は、制御不可能なほど多くはない。定められたプロセスで問題管理の基本的なマネジメントができているのであれば、結果的には「80対20の法則（パレートの法則）」にしたがっている。

ただし、プロジェクト・マネージャがいかに有能でも、一般に人間の短期記憶で同時に管理できる情報は7±2個とされている。ある程度の限界があって、プロジェクトに発生するすべての問題に対応することはできない。そこで、ドミナント・アイテムに的を絞るわけだ。

リスクを組み込んだマネジメントの概念を「変化に呼応する予測／予見のマネジメント」として**図表6-2**に示す。この図は、縦軸にプロジェクトに課せられたミッションの達成度合いをとり、横軸には時間をとる。曖昧性・不確実性を内包したままプロジェクトを開始し

た後、ある不確実な要素が明確になったときに、ミッションの達成度合いが高まることを示している。図では、将来のミッション達成度合いの振れ幅を予測して、Best-Moderate-Worstと表している。

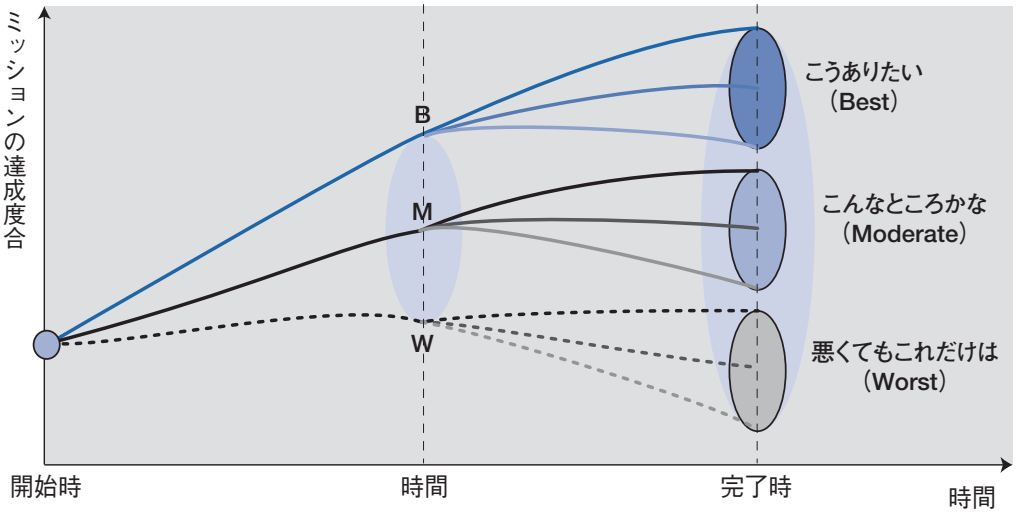
重要なことは、まずドミナント・アイテムの一つになり得る不確定要素が何であるかを早期に認識すること。次に、その不確定要素が明確になった時点で、何をどこまで決めておくべきなのかというシナリオを、プロジェクト・マネージャが描いておくことである。そうすることで、環境の変化が発生するたびに場当たり的な検討を繰り返さずに済む。プロジェクトの位置付けを見失わないマネジメントへの道が開ける。

6.1.2 選択肢(オプション)を用いたマネジメント

プロジェクト・マネジメントは、予測／仮説の下で実施されている。例えば、要求仕様が不確実で曖昧な状況のとき、スケジュール優先で次工程に進むべきか、仕様が明確になるまで現工程での検討を継続すべきか、選択に迫られることがある。

スケジュール優先で次工程に入ると、見切り開発による「手戻り」の増加がリスクになる。一方、仕様が明確になるまで待つと、プロジェクトの「遅延」がリスクになる。このように不確実な状況下で後戻りできない意思決定をする場合には、「意思決定者がその決定を先延ばしにできる自由度を最大にするマ

図表6-2●変化に呼応する予測/予見のマネジメント



図表6-3●ITプロジェクト・マネジメントで活用できるリアル・オプション

オプション名	内容
拡張オプション	達成目標などの拡張を選択する
撤退オプション	撤退（取りやめる）オプションを選択する
縮小オプション	達成目標などの縮小を選択する
切替オプション	異なる技術などへの切替えを選択する
選択オプション	拡張、撤退、切替、縮小などのオプションを選択する
バリア・オプション	特定の条件（バリア）を満たすかどうかで、次のオプションを選択できる

ネジメント」が正解である。

例えば、「仕様が8割がた明確になる時点まで次工程への進行を待ち、重要な検討事項に解を得たうえで次工程に入る」といったマネジメントである。ベテランのプロジェクト・マネージャは、作業ロスを少なくするこのようなマネジメントを行っている。当初の不確実な状況が明確になったときに、あらかじめ設定したオプションを選択して、方向転換する。方向転換する際には、新たなオプションの設定を行うのである。

ここで述べた「不確実な状況下で不可逆的な（後戻りできない）意思決定をする場合に、意思決定者がその決定を先延ばしにできる自由度を最大にする」マネジメント手法は、『リアル・オプション』と呼ばれ、他業界のプロジェクト・マネジメントでは既に用いられている。

リアル・オプションには、**図表6-3**のような選択肢がある。例えば、要件定義工程で要件が的確に定義できていないため、業務部門からの承認が得られないという状況を想定してみよう。ここでは、次のようなリアル・オプションを用意することが考えられる。

拡張オプション

発注側としては、業務部門からの要望や要件を的確にかつ十分に説明できていないことが問題となる。オプションとして、「発注側の担当者を、要件の説明が十分にできる人に変更する」を選択する。

バリア・オプション

受注側としては発注側の業務知識が不足しているために、これまでの説明を十分に理解できていないことが問題となる。近日中に要員増強が必要であり、かつそれが満たされないとき、次に選択するオプションとの連続性が必要である。バリア・オプションとして、

「10日以内に業務を熟知した担当者に交代することを条件に設定する」を選択する。

撤退オプション

バリア・オプションで設定した条件が満たされない場合には、破綻するプロジェクトを継続するよりも、オプションとして「次工程に入らず、プロジェクトを中止する」を選択する。これらのオプション選択の流れを**図表6-4**に示す。

6.2 プロジェクト・マネジメントの勘所

6.2.1 要件定義工程

要件定義は、上流工程における中心的な活動である。その際、プロジェクト・マネジメントの観点から是非とも押さえておきたいポイントは次の通りである。

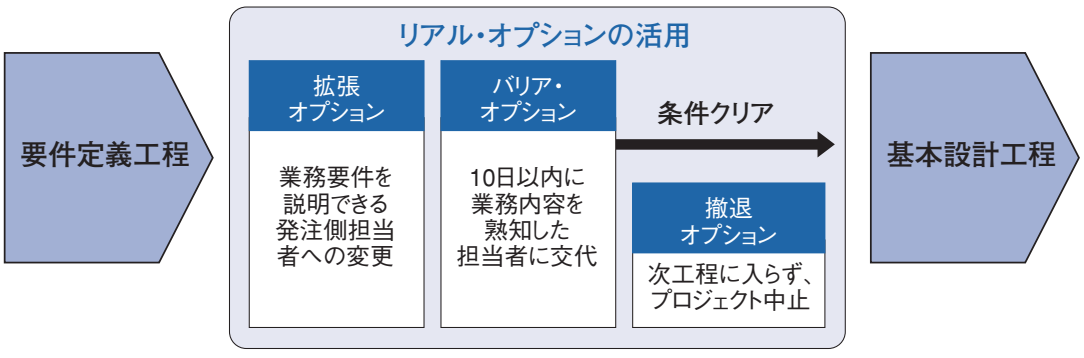
(1)要件が確定できない案件への対応を強化する

上流工程の成果物として、プロジェクトの方向を決めるプロジェクト計画書を作成する。一般的に、プロジェクト計画書の作成作業は、次のように進める。

① スコープ定義
② 作業定義・資源計画作成
③ 作業順序設定・所要期間見積もり・コスト見積もり
④ スケジュール作成・予算設定
⑤ プロジェクト計画策定

要件定義工程において、スコープ定義（業務、アーキテクチャなどのシステム、システム運用、移行や非機能要件）を行うが、ここで定義されたスコープが

図表6-4●ITプロジェクト・マネジメントでのリアル・オプション活用例



プロジェクトのスケジュール、コスト、体制などの計画を立てる際のインプットである。

要件を整理した段階で発注側と合意する必要があるが、ステークホルダーの範囲が広がり、人数も増えているので、発注側だけでは決められないケースも多くなっている。特に、システムが社会インフラとなっており、障害発生時の影響範囲、影響規模が大きく、経営上のリスクにもなっている案件で顕著である。

このような場合、受注側が積極的に、ステークホルダーの合意をとる働きかけをしたり、他案件での経験を基に適切な要件を提案して、発注側の決定を促したりすることが必要である。また、ステークホルダーが多く、複雑な要求がある場合、要件定義が不明確あるいは間違いであることがないように業務とシステムの要件定義それぞれに専門家を投入する必要がある。さらに、担当者では決定できない場合には、発注側の経営者を加えて、経営リスクに応じた要件を決めてもらうといった対応が必要である。

(2) 発注側のキーパーソンを把握するチャンスを活用する

要件定義工程においては、発注側の参画が必須であり、必然的に発注側とのやり取りが多くなる。このときが発注側を知るチャンスでもある。発注側の属人性、会社・組織としての文化、およびステークホルダー、キーパーソンを観察するとよい。

キーパーソンを把握し、組織の文化に応じた対応（合意形成）をとること、および発注側の信頼を得ておくことは、その後の工程での発注側との検討作業、結果の承認をスムーズに進めるうえで必須条件である。

(3) 要件定義の決定責任はあくまで発注側にあることを明確にする

発注側と一体となって活動する要件定義工程では、両者の役割が不明確なままに作業を進めることがあり、そのことによって責任の所在が不明確になることがある。

要件の決定をサポートすることは必要だが、要件全体は発注側しか把握できないので、最終決定責任は発注側にあることを明らかにしておく必要がある。責任を明らかにしないままに要件が決まると、要件定義を終了させることができても、多くの要件漏れが隠される結果となり、その後の工程で要件変更多発の原因となる。

結果、責任の所在が不明確であることが明らかになり、納期遅延・コスト増大となる。

(4) 要件定義の不備、特に曖昧性を徹底的につぶす

要件定義の不備は、この段階で徹底的にチェックしておかないと、下流工程で大きな手戻りを生じる。要件定義の不備を防ぐための方法を**図表6-5**に示す。要件が決定した後でも、その記述・表現が曖昧だと、後工程に大きな影響を及ぼす。

6.2.2 プロジェクト計画書作成

上流工程における重要な成果物としてプロジェクト計画書がある。プロジェクト計画書を作成する時点では、次のことを見える化する必要がある。

(1) プロジェクトのドミナント・アイテムを把握する

プロジェクト計画書を作成する前に、俯瞰図を用いて、プロジェクトのドミナント・アイテムを把握しておく。

(2) プロジェクトの目的、成果物、成功基準を明確にする

プロジェクトの目的、成果物、成功基準という重要な要素は、プロジェクトの目的、成果物の品質・納期・コストへの影響が大きい。

成果物が不明確なプロジェクトは稀であるが、成果物の作成にとらわれて目的が見失われているプロジェクトを

図表6-5●不備のある要件定義の防止策

不備の内容		防止策およびチェック方法
誤り・欠落	誤った情報	●不完全なスコープとならないように、複数の立場の人によりスコープの食い違いがないことを検証する ●システム利用時のコンセプトが明確になっているか検証する ●検証の成果に論理的根拠を盛り込む ●ステークホルダーを巻き込んで要件の検討、成果物のレビューなどを実施する
	欠落	●標準的なテンプレート/チェックリストを用いる
曖昧性	不明確	●標準的な記述方法から逸脱していないかチェックする （例：二重否定や「など」という表現は極力避ける） ●妥当性を確認する
	欠落表現上の複雑さの排除	●単純なフォーマットを用いる （例：区切りが分らない長文（100文字以上）になっていないか） ●エディタを使用する
	見当違い	●標準的なテンプレートに従って検討成果を記述する
実行や運用での発見方法		開発者は、「なぜ？」と尋ねる習慣をつける。 要件定義の記述者は、「証明に何が必要か？」を考える。 これらのやり取りが必要である。

目にすることがある。目的が明確でない場合、当然ながら、成功基準も定義されていない。

(3)スコープの確定の度合いを把握する

発注側が参画するスコープ（要件・機能）の確定作業におけるステークホルダー、発注側の対応から発注側の特性やプロジェクトの問題を観察できる。また、要件定義の発注側の承認段階における状況を観察することによっても、スコープの確定度を推測できる。

承認に際して発注側内部が紛糾したり、納得できる説明がないままにスコープが変更されたりする場合には、スコープの確定度が低いと判断する。

その場合、スコープ変更に対応できる契約になっていることを確認し、対応できていない場合には必要な条項の追加を交渉する。そのうえで、スコープ変更を前提としたスケジュールを計画しなければならない。

(4)クリティカルなスケジュールの状況をマネジメントする

まず、スコープに照らして適切なスケジュールが計画されているかを見る。

不適切なスケジュールとは、納期から機械的に逆算して作成され、不足する日数のつじつまを合わせるために、滞りなく上流工程を終わらせることを想定して作成されたものである。このようなスケジュールでスタートすると、実際には上流工程の期間が短すぎて、成果物である要件定義書、基本設計書の品質が悪く、次の工程の進捗に影響する場合が多い。

また、作業量が莫大な大規模プロジェクトでは、プロジェクト・マネージャがすべてのスケジュールをマネジメントできないし、非効率である。そのような場合、プロジェクト・マネージャがマネジメントすべきクリティカル・パスに関連する7±2個に絞った「スケジュール俯瞰図」を作成し、そのスケジュールを重点的にマネジメントするとよい。

(5)プロジェクトの制約条件を把握する

プロジェクトの制約条件とは、社内のリソース確保、チーム・ビルディングの状況、コスト、契約条件などである。これらの要素により、プロジェクトが成功するかどうかが決まることが多く、問題への対応（制約を受け入れる、制約を回避する、制約をやわらげる）を状況に応じて変える必要がある。

6.3 リスク・マネジメントの勘所

今まで失敗したプロジェクトを概括

すると、リスクが内在しているにもかかわらず、問題が表面化するまで手を打たなかった、または、打てなかったケースが多い。いわゆる、リスク・マネジメントが不十分であったわけだ。リスクを回避するには、プロジェクトでリスクが問題となって顕在化する可能性のある要素について、①早期にリスクをリストアップする、②万一リスクが顕在化した場合の回避策を事前に用意する、③リスクが顕在化する状況にあるか否かを検証する、といった対応が必要である。

6.3.1 リスクの識別

(1)リスクの性質

どこにどのようなリスクが潜んでいるかを識別するためには、リスクがどのようなものであるかを整理しておく必要がある。リスクがどこに発生するか、またいつ発生するか、何をリスクとして見るか、といったリスクの起因内容によって整理することができる（**図表6-6**）。

さらに、リスクがどのような性質を持つものなのかを説明する。

まず、「システム開発にはリスクが常に伴う」ということである。システム化の要件という抽象的な事柄をシステムという具体的な形で実現するので、システム化することによる要求の漏れというリスクが常に存在する。

また、「リスクは現時点より必ず後工程にて顕在化する」。上流工程では、サービス・インまで時間的余裕があり、リスクが現実的な問題となって発生しないために、リスクの認識度が低くなる。

「想定していなかったリスクが後工程で顕在化した場合、必ず大きな損害となる」という性質もある。リスクが顕在化した場合、予定の作業を継続することに加え、問題処理のために、体制、工数、期間、費用、メンバーのモチベーションを度外視して対処しなければならない事態に陥る。対症療法でマネジメントを進めるために、問題の根本的な解決がされないままになる。また、プロジェクト・メンバーの心理的作用

図表6-6●リスク起因区分

起因区分	起因するもの
発生する場所	●市場、業界 ●発注側 ●個人 ●プロジェクト ●その他自然災害など
発生する時期	●受注、契約段階 ●要件定義段階 ●設計段階 ●製造段階 ●テスト、移行段階 ●運用開始以降
リスクとして見る内容	●金銭 ●信用、社会性 ●人材 ●ビジネス ●企業の存亡

として問題を隠ぺいしたり、過小評価したりするケースが多い。

リスク・マネジメントは最終的に発注側、受注側両者にとって有効でなければならない。リスクは、発注側、受注側の立場によって相反する場合があり、リスクを回避するためには、最終的に両者にとって有効な内容でなければならない。一般的には、受注側が過負荷になることが多いが、これは本来の姿ではない。

(2)リスク要因のとらえ方

プロジェクトにおけるリスク要因を決めるキーポイントは、次の3点に集約される。①決定すべきことは、すべて決定しているか否か。また、以降、決定事項に変更がないか否か。②決定した内容に客観性があるか否か。③決定した内容が計画通り着実に進捗しているか否か。

現実には、この3点が問題なく遂行されているプロジェクトが少ないため、これらが守られないことを想定して、④課題を速やかに表面化させ、問題発生以前に対処することが可能か否か、が加わる。

この①～④のキーポイントをリスク回避のための基本行動に照らし合わせると次のようになる。

物事の決定とそれ以降の変更をなくすために、基本的計画、すなわち、開発範囲、費用、体制、負担、開発手法、開発期間などを明確にする必要がある(①)。また、決定事項の客観性を担保するには、レビュー（承認を含む）を徹底させることが欠かせない（②)。決定事項の着実な推進のためには、プロジェクト・マネジメント標準の確立とメンバー全員への浸透、進捗フォローが不可欠である(③)。そして、課題に対処するためには、迅速なエスカレーションが必要である(④)。

(3)リスクの認識

リスク・マネジメントを難しくしている理由の一つは、リスクを「○○不備」、「○○不足」というように認識しなければならないことである。すなわち、現在は上流工程に位置しながら、中流・下流工程での結果を想定しなければならない。

これは、あらかじめプロジェクトの先をすべて見通さなければならないため、経験不足のプロジェクト・マネージャには困難である。大問題が発生することが分かっていれば、手を打たないことは通常あり得ない。分からないか、または発生しないだろうという認識から手を打たなかったのである。その背景として、「計画がきちんと明確になっているか」、「体制が整っているか」、「費用

は妥当か」、「開発期間は十分か」、「発注側との関係は良いか」など、概念的で、かつタイミング的に疑問のあるリスク認識のみがリストアップされていることが多い。

また、数多くのリスク認識項目がリストアップされると、数の論理に気を取られ、あたかもリスク・マネジメントを十分に行っているかのような錯覚に陥りやすい。

しかし、本質的にリスク・マネジメントを行うということは、「次の工程がうまくいくために直前の工程がどういう状況にあるべきか。何が明確になっていなければならないか」を考えることだと言ってよい。つまり、次の工程をスムーズに進めるに当たり、前工程の何が阻害しているのかということを単純にリスク要因として認識するということである。

6.3.2 リスク分析

(1)リスク影響度の考え方

システム開発は、出来合いのものを同じ環境や条件で適用するよりも、カスタマイズするか、個別にシステムを構築するケースが多い。システム開発の上流工程におけるリスクは、①目的、要件の明確性、②計画、体制、組織の具体性、③技術、進捗の確実性、④プロジェクト・マネジメントの客観性、という4つの要素に分類できる。

一方、もう1つのリスク回避を阻害する要因の分類方法として、外部要因と内部要因がある。これらの分類に、①～④を組み合わせると次のようになる。

外部要因は、目的、要件、計画、費用、体制に密接に関連する。プロジェクトにとって根幹を成す要因であり、影響範囲は大きい。

内部要因は、計画、組織、技術、進捗、管理に密接に関連する。外部要因に比べ、影響範囲は小さいものの、具体性という点で結果への関連は大きい。

したがって、リスクを分析する場合には、外部要因に力点を置くべきである。

(2)リスクの大小の判断方法

リスクの大小判断は、プロジェクトの特質により異なるが、いくつかの視点があるのでここで紹介する。

まず、連携動作する個所が多い場合、リスクが大きいと判断する。例えば、関連するサブシステムが多い場合に、サブシステム間のインターフェースの仕様変更を想定してみよう。たとえプログラムの変更量が小さくても、設計の見直し、運用の変更、テストのやり直しなど影響が大きいため、リスクも大きくなる。

また、後工程になるほど、リスクが大きいと判断する。システム開発の途中での変更、追加はやむを得ない事象としても、早い時期での変更、追加と、サービス・イン直前の変更、追加ではリスクの大小が異なる。

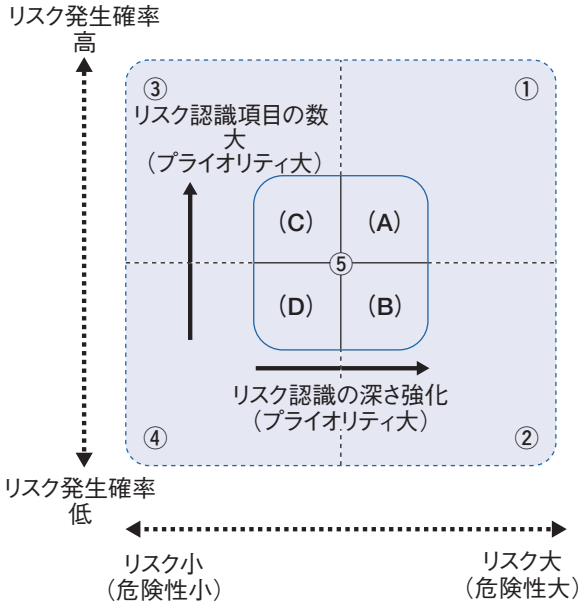
例えば、テスト段階で設計にかかわる変更、追加は、工数、進捗に関係するだけでなく、作業の完成度にも影響する。当然リスクは大きい。一方で、設計作業途中での変更、追加は、関係者が設計に注力しているため、一般的にリスクは小さい。

他のシステムや工程に対する影響度合いが大きい場合、リスクが大きいと判断する。過去の事例から考えると、①品質（正確性）、②性能、③機能（広義）、④納期、⑤費用の順でリスクの大小を判断するのが妥当と考える。ただし、代替案があり、かつ代替案による対応が容易な場合にはリスクが小さいと判断する。

⑶リスクの重み

リスクを認識してリストアップすることは、経験を積み重ねれば比較的容易だが、これらをすべて同等に扱うことは非現実的であり、有効なリスク・マネジメントとはならない。リスク・マネジメントを過不足なく偏りなく実施するためには、リスクを**図表6-7**のリスク要因マトリックス上に整理する必要がある。

図表6-7●リスク要因マトリックス



現場でどのようにリスク・マネジメントがなされているかを見ると、①の領域（リスク大、リスク発生確率高い）は危険性が高いことから、企業、または組織として強く注目される。半面、中にはプロジェクトだけで解決できないリスクが含まれているため、手が打たれないままになる。特にコミュニケーションの良くない組織においては、後工程で大問題が発生するケースがある。失敗として公表されるプロジェクトの大半は、この領域のリスク・マネジメントが不十分だったと言っても過言では

ない。

②の領域は、危険性が大きいと理解しつつも、発生確率が低いため、リスクと認識することが難しい。しかし、無理をすれば解決できる場合もあり、リスク・マネジメントが最も難しい領域である。

③の領域は、リスク発生確率は高いが、リスクが小さく、リスクの認識が容易である。このため、集中的なリスク・マネジメントが行われることが多い。

④の領域は、リスク発生率が低く、リスクも小さい。リスク・マネジメントというより、課題管理の対象として扱うことが多い。

⑤の領域は、リスク発生確率の高低、リスクの大小の視点と異なり、曖昧性に起因している。例えば、設計書のレビュー、合意の重要性は誰でも理解しているが、レビュー、合意の内容、深さがどの程度で十分かという物差しの定義は困難である。この「十分な合意のレベル」をできるだけ明確にして見定めることが⑤の領域でのマネジメントである。この領域のマネジメントが不十分であると、いずれ①～④の領域にも波及する。

⑤の領域についてのキーポイントは、多くのマネジメント手法で述べられている。しかし、リスク・マネジメントの物差しが不足しているか、その物差しが現実に即していない場合がある。この領域は、プロジェクト・マネージャとしての経験、資質、センスなどがより多く問われている。極論すると、プロジェクト・マネージャは、日々、この領

図表6-8●リスク要因の概要

リスク・マネジメント領域			領域の特徴	
領域	発生確率	危険性		
①	高	大	●組織的に注視される ●高い地位者が対応するためエスカレーションが重要	
②	低	大	●リスクと認識し難い ●無理することにより解決できる場合があり、リスク・マネジメントが難しい	
③	高	小	●リスク・マネジメントとして注力しやすい（①、②がなおざりになりやすい）	
④	低	小	●プロジェクト・マネジメントとしては、見切る場合が多い ●課題管理として対処するケースが多い	

域のリスク・マネジメントに追われているのが実情である。

プロジェクト・メンバーは、リスクとなる問題とその発生確率を比較的早い段階で、感覚的にとらえていることが多い。プロジェクト・マネージャ、プロジェクト支援組織は、それらの内容を聞き出してリスク識別・リスク分析をする必要がある。

6.3.3　リスクの評価方法

リスク要因マトリックスの各領域の特性に応じてどう対応すべきか、方法を説明する。

リスク要因マトリックスの領域を2分割した縦軸（リスク発生確率）と、2分割した横軸（リスクの大小）によって（A）～（D）の4つの領域に分け、それぞれの領域におけるリスク要因と、対処策を整理したのが**図表6-8**である。

リスク要因領域（A）

この領域のリスク要因は、基本的事項（特にQCDおよび法律、信用にかかわる重要な項目）が合理性を持って確定しているか否かである。

上流工程だからといって、要件、仕様、方式、体制、費用、計画などの基本的事項が、この段階で決定していない、または確定していない状態を決して「やむを得ない」と容認してはならない。このような基本的事項の合理性に立脚し、このリスク要因領域におけるリスク認識を行う。

現実には、基本設計段階になっても要件が決定しない、または決定したと

	リスク要因領域		
	領域	リスク・マネジメントに必要な基本行動	リスク発生の兆候と対応
	（A）	●基本的計画の明確化（開発範囲、費用、体制、開発手法、納期などの基本計画）	●計画通りの進捗が逆に問題 ●容易にエスカレーションできるルール作りと、その運用
	（B）	●原則、要件、計画を変更しない ●レビューの徹底	●些細な変更が多発 ●無理、無茶の防止 ●どんな些細な変更も発注側、受注側の責任者の承認
	（C）	●プロジェクト管理標準の確立とプロジェクト・メンバーへの浸透	●結果がプロジェクト管理標準の期待値から逸脱する場合が多い ●リーダーによる管理結果のフォロー
	（D）	●プロジェクト管理標準の確立とプロジェクト・メンバーへの浸透	●計画通り進捗するが懸念が残る ●管理結果のフォローをプロジェクト・メンバーに任せる

しても仮決定の意味が含まれていることが多い。しかし、ある時期に物事が確定していることを前提において、規模、生産性、工数、期間など綿密に計画を立てているにもかかわらず、実際のプロジェクト運営が未決定、未確定、変更やむなしととらえることは誤っている。

発注側、受注側もそれが当然と考え、また、それら不確定要素に柔軟に対処することが受注側の実力と解釈する風潮があるとするならば、それはリスク・マネジメント上、非常に危険である。最初から失敗を受け入れているようなものである。

仮に、未決定、未確定事項がある場合、どんな些細な未決定、未確定事項でも問題視すべきだ。発注側、受注側の社内で問題提起（エスカレーション）し、公にすることが重要である。その後、計画を見直して、発注側、受注側の責任者の承認を得るなどの施策を実施する。これは、プロジェクトという立場を超えた発注側と受注側という企業間の問題だからである。

無理をして決定、確定しないことも重要である。計画通りに進捗しない場合、決定、確定に向けた行動をとることは当然である。しかし、この段階で未決定、未確定があるということは、その背景に根深い問題が潜んでいることにほかならない。それを無理に決定、確定しても、必ず後工程で変更、追加が発生し、混乱する。

プログラム製造、テストの消化などは、ある程度、増員、設備強化など物理的対応により進捗が促進される可能性がある。だが上流工程においては、最低限必要な期間、巻き込むべき組織、採るべきプロセスがあり、これを無視できない。

一方、問題を複雑にする理由として、納期まで時間的余裕があるため、「後工程で何とかなるだろう」という希望的観測、精神論的になることである。特にこのリスク要因領域は、現工程で予定通り進捗しないにもかかわらず、後工程で計画通り進捗する、あるいは遅れを取り戻すなどプロジェクトの進捗が向上することは稀であることを認識すべきである。

リスク要因領域（B）

この領域のリスク要因は、以降の工程で、ジワジワと痛手を被るものがないか否かである。

この領域には、内容、質という若干抽象的な要素が問われるものと、抽象的要素を具現化するための組織的バックアップ体制が整っているかという要素がある。それらを次の視点で認識する必要がある。「計画に曖昧さがある

か」、「計画実行に曖昧さがあるか」、「コミュニケーション不良、モチベーション低下など人間的な問題が内在しているか」、「上記の課題をひたすら頑張ろうなどという精神論で解決しようとする姿勢があるか」、「第三者的立場による検証など客観的評価ができる組織か」、「エスカレーションのルールが明確か、エスカレーションの効果は十分か」などである。

例えば、性能はシステムにとって重要である。そのため、リスク要因領域(A)での必要性能を確定、試算、検証する手立て、つまり性能確保については誰でも認識する。

仮に、毎秒100件のトランザクションを処理する要件の場合、設備、ツール、評価体制などを整え、毎秒100件の性能要件を満たそうとする。問題は、実システム、あるいは実運用前のテストで、毎秒100件以上の負荷がかかった場合、この性能値の扱いについて発注側、受注側で大きく意見の分かれるところである。

発注側は、毎秒100件以上の負荷がかかった場合、若干の処理遅れがあっても、システム障害に至るとは考えない。一方、受注側は、毎秒100件の負荷まで検証するとしても、それ以上の負荷がかかった場合にシステム全体にどういう影響が出るかは、設備、工数、ツール、期間などの制約、困難が発生するため、特別な場合を除いて検証しない。その結果、一般的に性能改善には大きなリスクが伴う（危険性大）と理解しているにもかかわらず、リスク発生確率が低いため、見落としやすい。これを防ぐためには、第三者検証による先を見越したリスク・マネジメントが重要である。

リスク要因領域(C)

この領域は、プロセスが目的に沿って正常に進展しているか否かが視点となる。

ただし、この領域は、発生確率が高いためリスク認識として欠落することは少ない。何がリスクか、この領域でのリスク・マネジメントを怠ったらどうなるかは、参考文献が多く、マネジメント手法も確立されている。プロジェクト・マネージャ自身も体験的に知っている。特に、プロジェクトで定めたルールの順守をプロジェクト・メンバーに任せきりにせずに、プロジェクト・マネージャ自身がマネジメントしなければならないことに注意する必要がある。

例えば基本設計工程なら、要件、規模、機能などに基づいて、設計書の目次、書き方、レビューの仕方、是正処置の手順を決めたルールを作るか、標準のルールを利用する。一般的に、進

捗会議での報告は、基本設計書の予定総項目数、予定総機能数、予定合計ページに対する実績総項目数、実績総機能数、実績合計ページ数などの数値から、進捗率○○%という報告が行われている。

その報告を受けたプロジェクト・マネージャは、進捗率に興味を示すものの、基本設計書の深さ、基本設計書全体の整合性、あるいは基本設計に潜んでいるリスクに目を向けることが少ない。特にスケジュール的に厳しい状況下では、なおさらである。

そのため、レビュー、または第三者機関による検証を行う場合に、定量的検証を通じて定性的内容の検証に注力する必要がある。

同様に基本設計工程を例にとると、レビュー、第三者機関により検証を行う場合、「要件に過不足がないか」、「基本設計書に正しく反映されているか」、「要件に定義された内容から、仮に逸脱した場合、その扱いが基本設計書で明確になっているか」といった検証項目を重視する必要がある。

リスク要因領域（D）

この領域は、マニュアルの体裁、分かりやすさ、画面のレイアウト・配色、報告書の様式、プロジェクト・ルームの環境、プロジェクト・メンバーの資質など、重要ではあるが経験、説明、教育により課題を解決できる内容である。

一般的にリスクが小さいため、リスク・マネジメントとしては、重要視されていない。しかし、この領域は、当事者の趣味、嗜好、または企業文化というような価値観により、リスク認識が全く異なるため、扱いが非常に難しい。

リスクを認識するうえで、要件の分かりやすさ、見やすさ、判断の容易さなど、概念的なものがあるときに、本当に要因領域（D）のリスクであるかを検証することに注意しなければならない。

半面、この領域のリスク・マネジメントは、計画に反映させ、地道に改善しておかないと、中流・下流工程で費用、工数、体制、納期、納品に大きく影響することを意識する必要がある。同時に見切りも必要である。

6.3.4 リスク・マネジメントの事例

上流工程におけるリスク・マネジメントは、プロジェクト運営にとって最も重要である。受注側の組織の位置付け、ビジネス戦略、プロジェクトの方針、リスク項目、内容、リスク認識、リスク判断基準、リスク発生時の回避の方法など、多くの要素によってリスク・マネジメントの方針が決まる。

プロジェクトを推進してきた企業の中には、過去の経験、知識、失敗・成功事例を活かし、独自のリスク・マネジメントを確立し、運用しているところがある。ここではリスクの「定量化方法」と「分類方法」の例を紹介する。

リスクの定量化方法には、「直接リスク評点を付ける」という方式と、「係数を加味したリスク評点を付ける」という方式がある（**図表6-9**）。前者は、開発規模、実現性、開発規模の増加要因などを分類してあらかじめ点数化して

図表6-9●リスクの定量化事例

評価方式	評価方法	評価例	特徴
直接リスク評点を付ける	●あらかじめリスク項目の重要度を点数化しておき、それを選ぶことにより評点を求める ●評点＝評価項目の点数	（規模） 規模大　1.0 規模中　0.5 規模小　0.3 （実現性） 困難　0.7 やや困難　0.5 容易　0.3	●比較的多くの人が採用しやすい ●リスク・マネジメントの概念が浸透しやすい
		（開発規模増加） 品質悪化　0.3 納期延期　0.4 収支悪化　0.3	●具体的な管理が可能 ●成熟したプロジェクト・マネジメント文化が必要
係数を加味し、リスク評点を付ける	●発生確率、リスク発生時の影響を計数化し、評点を求める ●評点＝発生確率×影響度	（発生確率） 高い　1.0 普通　0.5 低い　0.3 （影響度） 影響大　1.0 影響中　0.6 影響小　0.3	●リスク評点のバリエーションが大きく、リスク・マネジメントが深まる
	●リスクの重要度に、リスク・マネジメント状況を加味し、現実的に、どの程度のリスクが発生するか評価する ●評点＝発生確率×影響度×緊急度×対応策×リスク状況	（緊急度） 緊急（即時）　1.0 中（1カ月以内）　0.5 低（1カ月以降）　0.1 （対応策） 対応済み　0.1 対応中　0.5 未対応　1.0 （リスクの状況） 結果（発生中、発生後）　1.0 プロセス（リスク発生の恐れ）　0.5	●プロジェクト・マネジメントと一体化したリスク・マネジメントが可能 ●リスク・マネジメントが若干難しい

図表6-10●リスク項目の分類事例

方法	分類	
	大分類	中分類
リスク項目をビジネス全体の視点で捉える（ビジネスの引き合い、見積もり、受注、再見積もりなどの観点で捉える）	外部要因 内部要因 プロジェクト固有	引き合い経緯、発注側情報、 システム情報、契約条件 見積もり条件
計画時以降の プロジェクト・マネジメントの 視点で捉える	プロジェクト・マネジメント その他 プロジェクト固有	結合、スコープ、タイム、コスト、品質、 人的資源、コミュニケーション、リスク、 調達 技術
システム開発の計画、 実行の視点で捉える	立上げ、計画 コントロール、実行	統合、発注側、契約、スコープ、業務、 機能、技術、タイム、コスト、品質、 組織、コミュニケーション、調達 統合、機能、技術、タイム、コスト、 品質、組織、コミュニケーション、調達
リスク・マネジメント上 重要と思われるものを 重点的に捉える	契約、約束 発注側 システム プロジェクト・マネジメント 要員管理 調達管理 その他	前提条件、契約 発注側情報、体制 開発、環境、信頼性、性能 プロジェクト・マネジメント 組織、体制 調達管理
プロジェクト・マネジメントに 集中して捉える	財務 発注側 レビュー 進捗 品質 体制 リスク・マネジメント	プロセス、結果 プロセス、結果 プロセス、結果 プロセス、結果 プロセス、結果 プロセス、結果 プロセス、結果

おき、その合計点でリスクを定量化する。比較的多くの人が採用しやすい。後者はリスクの発生確率や影響度を基にして、各リスク項目の点数を付ける方法となっている。

次に、リスクをどのように分類し管理しているのか、**図表6-10**にリスクを5つに分類した例を挙げる。プロジェクトの引き合い経緯や発注側の企業情報など、ビジネス全体の視点から見た分類や、システム開発の計画／実行の視点で項目を整理した分類方法があるなど、どの時点で、どの範囲のリスクを洗い出そうとするかによって、様々な分類の仕方がある。

第7章

見切りながらのプロジェクト推進

7.1 実情に合わせたマネジメント

俯瞰図を使った「見える化」で述べたように、プロジェクト・マネジメントで最も避けなければならないことは、ドミナント・アイテム（プロジェクトの成否にかかわる本質的問題）を見失うことにより、プロジェクトが失敗に帰することである。

ドミナント・アイテムの数は、制御不可能なほど多くはないが、プロジェクト・マネージャがいかに有能であっても、一度に処理できる数には限界がある。プロジェクトに発生するすべての問題には対応できない。

そこでドミナント・アイテムに関する次の時点での「見切り」が必要になる。① 最初にドミナント･アイテムを選定するとき、② 新たにドミナント･アイテムを追加し、不要になったドミナント･アイテムを削除するとき、③ ドミナント･アイテムに変化が生じたとき、である。

さらに、ドミナント・アイテムに合わせてオプションを設定している場合には、次のような情報収集とオプションの選択を決定する必要がある。まず、どのような状況になったときに、オプションを実行するのか。そして、不要になったオプションは何か、新たに設定したオプションは何か、ということに留意する。ドミナント・アイテムの変化への対応やオプションの実行は、ほとんどの場合、リスク・マネジメントの対応策と重なる。

7.2 良い見切りと悪い見切り

「見切り」という言葉には、① 全部見る、② 見込みがないとしてあきらめる、③ 見極めをつける、④ 見かぎる、といった意味がある。できる限りの状況認識を行ったうえでのギリギリの判断という意味であれば③の見極めに近いが、一般には、②④の見かぎるに近い使われ方をしている。

上流工程においては、2つの意味で

「見切る」場面が見られる。1つは、プロジェクト・マネージャが得られる限りの情報を駆使して、自己責任のもとにギリギリの選択をしているケースである。最悪の場合も見極めているので、結果がどう出ても対応ができる。

もう1つは、情報不足のまま「エイヤッ」と決断したり、「その時点では先が読めなかったのだから仕方がない」と成り行きに任せたりするケースである。実際には、こうした見切りが圧倒的に多い。

2つの見切りには大きな違いがある。ここで、「良い見切り」と「悪い見切り」という問題が生じてくる。

プロジェクト・マネージャはとにかく決断しなければならない。決断しないことには、物事は常にその場に留まり前進しない。要所での決断に当たって、自分の判断が常に正しいなどと考えている“幸福な”プロジェクト・マネージャはいない。苦渋の選択をしているはずである。

重要なのは、決断の結果が「悪いほうに傾くことがある」というリスクを考えておくことだ。一言で言えば、この判断をしたら何が起きる可能性があるか、それが起きたらどの程度のロス（損失）が出るか、また、回復の道にはどういうことがあるかということを考えておくことである。これが「良い見切り」であり、リスク・マネジメントそのものである。

大きな危険を伴う可能性があるということを想定して見切っているなら、いわゆる“想定内”の出来事である。想定内であれば、なんらかの危険予知と対応策が考えられているので、それほど怖さはないだろう。

怖いのは、「そのケースは考えていなかった」という“想定外”のケースである。想定外とは、全くリスクに気付いておらず、対策を考えていない場合を指す。想定外の事象が発生したときにはパニック状態に陥り、メンバーが混乱に陥る。プロジェクトが失敗に終わる可能性も高くなる。

想定外の事象を皆無にするのは無理な話だが、持っている情報を基に、起こり得る様々な事象を想定内に持ち込む必要がある。事前に、「このプロジェクトでは、ここまで想定しておけば破綻することはない」と判断できることが重要である。

悪い見切りは、前述したような情報不足の状態や成り行きに任せた判断である。概して、独りよがりの決断になる。相手は悪くしないだろう、そんなことは起きないだろう、といった類であり、この「だろう」が危険なのである。失敗は、独善的になったときに多く発生している。

プロジェクトの状況は時々刻々と変

化する。したがって、リスクは定期的に評価し直すことが必要である。最初に考えたリスクの中には、時期的には過去形になってしまって、もう起こることがない項目もあるし、当然新たなリスクが発生している可能性もある。そのために、例えば、工程の切れ目での要所で見直しが必要である。

正しい情報からドミナント・アイテムを得て、それに対する様々なケースを想定して、根拠をもって判断すること（見切ること）ができるプロジェクト・マネージャであるなら、プロジェクトが成功する可能性は高い。

7.3 「見切り」のケーススタディ

本書の付録にある事例集の中から、いくつかの案件を選んで、実際のプロジェクトにおける上流工程のある時点で、どのような状況下でどのような見切りを行ったかを振り返り分析する。

7.3.1 スケジュールの見切り事例

まず、事例集の事例番号20のケース（**図表7-1**）を取り上げる。事例について一読したうえで、以下のケーススタディを読んでもらいたい。

この事例では、プロジェクトとしてはスケジュールの延期となり、納期に影響を及ぼしたという意味では失敗プロジェクトと評価せざるを得ない。しかしながら、プロジェクト・マネジメントの視点からどのような見切りを行ったかを分析すると、「良い見切り」を行ったと評価することもできる。順を追って考察する。

(1)状況の把握

上流工程では、要件定義、仕様確定といった発注側の意思決定が成果に大きく影響するタスクが活動の中心となる。特に、「利用者が多岐にわたる」、「仕様確定に組織的な承認が要る」など、発注側にとっても未経験な事項が含まれているプロジェクトでは、意思決定の弱さが容易に露呈することがある。

この事例では、発注側の経験の浅さ、決定権の不在などがあらかじめ把握できていて、当初用意した仕様確定期間では決着しないかもしれないリスクが見えている。

プロジェクト・マネジメント上の考慮点としては、「仕様確定に必要な期間は本来どれくらいにすべきだったのか」をプロジェクト・マネージャがきちんと説明できるか、という点にある。仕様確定の終了基準が何であるかという、この事例に限らない共通の問題となってしまう可能性もある。だが、あえて指摘するなら、決めるべき事項の数、調

整先の数、作成するドキュメント量などから、与えられた期間では決着しないことを定量的に示し、論理的に説明する必要があった。その中で、発注側の意思決定の弱さが、どれくらい期間延長に影響するのかを説明できなければいけない。

しかしながら、現実的にはそれができないため、仕様確定の期間を延長するという選択肢を採りづらい。これが問題を大きくしてしまう。

3カ月かけて決められない仕様を、5カ月かければ決められるかというと、答えは「ノー」である。発注側の意思決定の弱さは、承認会議を通過するなどの手続きレベルにまで分解して対処しない限り、時間を費やすだけの結果に終わる。むしろ期限を設定して、「いつまでにどこまで決まらないと、全体の進行が妨げられる」と説明し、発注側にも切迫感を持って検討してもらう方が、意思決定が進みやすい。

先に記したことと矛盾するようであるが、現実的には「仕様確定に3カ月かかる」という見方でスケジュールを引いているのではなく、「3カ月で仕様を確定する（したい）」というロジックでスケジュールを設定していることに問題がある。

(2)見切りの内容

この事例では、与えられた期間内で

図表7-1●事例番号20のケース

20 顧客側メンバーの体制が弱く、仕様が詰められない

顧客側メンバーは、システム化の経験が浅く、現場部門に対する仕様決定権も弱いので、仕様確定には計画より期間がかかる可能性があったが、ベンダー側の戦略上、受注する必要があった。プロジェクトが始まると、顧客側メンバーが多忙との理由により、仕様検討の稼働や時間が十分に取れず仕様確定が遅れ始めた。顧客側の要件の提出期限をベンダー側から提示。顧客は、遅れをとりもどすためメンバーの増強を行なった。また、スケジュールを延期することとなり、当初予定していた納期は守れなかった

事例における見切りの内容	本来の見切りの考え方
当初ベンダー側としては、顧客側メンバーのスキル不足を補う形で、ベンダー側で作成した仕様案をもとに打ち合わせすることにより、仕様確定をスムーズに行い計画通りに納めることができると考えて受注した	・顧客側の体制やスキルの不足が事前に分かっているのであれば、顧客側の体制強化を依頼する先（相談相手）を見極めておくこと ・仕様のたたき台が作成できるほどのスキルがベンダー側にあり、現行システムのリプレース案件である場合は、現行システムのドキュメントをベースに仕様詰めする部分を絞るなどの可能性を検討しておくこと ・契約時に顧客側起因のスケジュール遅延や予算オーバーについての免責事項を入れておくべきである
捉えるべき兆候	**対処例**
・顧客側メンバーの経験が浅く、決定権も弱い	・顧客側の責任を明確にした上で、顧客側の体制強化を依頼する ・システム検討範囲の縮小や工期延長について協議する ・現行システムのドキュメントなどをもらって、それをベースに検討を進められるかを判断する

効率的に成果を上げようとするプロセスの考案と、その方法による完成見込みを立てている。仕様確定に関するタスクの取捨選択においても、発注側の意思決定のスピードアップという視点から見切りを行っている。

また、「事例における見切りの内容」欄で、「仕様案を作成して、それをベースに打ち合わせする」という意味は、発注側との仕様確認に必要なドキュメントと、次工程のインプットとして必要な確認済みの仕様書、という最低限必要な成果物を取捨選択するという見切りも行われている。

プロジェクト・マネジメント上の考慮点としては、起こり得る事象の1つとして、ワースト・ストーリーを想定できていたか、という点が重要である。発注側の意思決定が弱い場合は、受注側が仕様案を提示して、発注側には「イエス/ノー」に近い形で回答してもらい、仕様を確定していく。受注側が発注側の業務内容や現行システムの構造などを十分に把握しているなら、スピード重視の最善の進め方であり、実際、発注側もそういう進め方を期待するから、現行システム担当の受注側メンバーが重宝がられるのである。本事例の場合は、少なくとも発注側の業務に明るく、仕様提案型の進め方ができる受注者であると思われる。

ここでのワースト・ストーリーは、仕様案の了解を「イエス/ノー」で積み上げていく途中のどこかで方向性を誤ることだ。受注側が提案する仕様は、ときにシステムの都合を優先した内容に陥ることがある。こうした場合、つじつまを合わせて仕様を確定したとしても、「そもそもの目的を達成できていない」という発注者視点の議論でひっくり返されてしまう。検討の途中で、発注側と改めて方向性の確認ができる場を設定しておくことが重要である。

発注側の経験不足と関連して、前述のような仕様案を提示して決めていく進め方を行うと、ますます受注側への依存度が高くなる。どの段階で何がどれくらい決まっていればよいのか、発注側が主体的に判断できなくなってしまう可能性が出てくる。

つまり、発注側が何種類の文書（図面）を作成すればよいのか、どれくらいの記述レベルがその時点で求められるのか、どれくらいの量が妥当なのか、などが分からなくなってしまうということだ。社内調整のために必要な資料も受注側が作成しないと、発注側の組織的な仕様の承認がなされなくなってしまう、などが想定される。プロジェクトに関して、発注側を「指導」できるレベルのマネジント力が発揮されなければならない。

(3)見切り判断

この事例では、仕様確定に際して、発注側に期限付きでタスクを渡すことができている。つまり、前述した受注側への過度な依存を回避できている。また、回答期限が守られないと、全体スケジュールが遅延することを発注側に説明できていて、上流工程のクリティカル・パスが発注側と共有できている。発注側が増員したということは、発注側も担当者が奔走するだけではなく、仕様確定に組織的に注力していることを意味している。

こういったレベルで発注側と課題を共有できたのであれば、最終的に期限延長という結果ではあっても、プロジェクト・マネジメントの質としては良好だったと評価することもできる。とはいえ、Q（品質）、C（コスト）、D（納期）に影響を及ぼす結果となったことは事実だ。結果を重視する評価をするなら、やはり失敗プロジェクトということになる。

この事例において最も重要なことは、「説明しながら対処する」という一言である。受注側が発注側に対するプロジェクト状況の説明責任が果たされていることが大事なのである。

これによりプロジェクトを推進する発注側と受注側のベクトルを合わせることができれば、極端な言い方ではあるが、少々の見切り誤りによる課題を残したとしても、プロジェクトを終結させる力として作用する。

プロジェクトは期間延長という結末を迎えているが、上流工程の仕様確定に関する期間延長は本当に悪なのだろうか。課題を残したまま次の工程へ進む方がむしろ悪であると思えるし、潜在的な課題が見えないまま次工程へ進むことは最悪である。必要な工数、日程を割り出したうえでの期間延長は、検討対象のスコープが明らかになったことを意味しており、前向きにとらえるべきである。

7.3.2 機能の見切り事例

次のケーススタディとして、事例集の事例番号58のケース（**図表7-2**）を取り上げる。この事例はパッケージ・ソフトを利用した開発で、パッケージの機能と業務機能のギャップ（パッケージで実現できないこと）が製造工程になって判明したため、大幅な方式の見直し、大きな手戻りが発生している。

失敗の原因は、パッケージ機能の検証が不十分な状況で、パッケージでは実現できない発注側の要望を受け入れてしまった点にある。なぜ、このような状況に陥ってしまったか、どのような点を考慮してマネジメントをすべきだったのか、考察してみる。

(1)状況の把握

受注側が戦略的な受注を行う場合は、不十分な体制や条件で受注することが多く、いろいろなリスクがプロジェクトに潜んでいる。また、各ステークホルダーの思惑により、回避できないリスクも存在する。

戦略的な受注案件の場合は、プロジェクト・メンバーに任せっきりにするのではなく、受注段階から組織あるいは会社として、プロジェクトを監視する必要がある。

サブベンダーのプロジェクト・マネジメント上の考慮点としては、「パッケージ利用のシステム開発を戦略的に受注した意図が共有できていたか」が焦点の1つとなる。自社内に経験者がいない状況で、パッケージ・ベンダーのサブとして受注するのであれば、プロジェクト自体が「自社のシステム開発メニューの拡大に向けた試験的要素を含んでいる」ということが想定できる。それがプ

図表7-2●事例番号58のケース

58 パッケージ開発では業務要件とのギャップが重要な見極めのポイント

パッケージ製品による開発に際し、プライマリーベンダーを支援する形態でプロジェクトに参画したが、開発体制にパッケージ製品に関するエキスパートを組み込めないまま上流工程を進めたため、パッケージ機能と業務要件とのギャップが製造途上で判明し、方式全体の見直しを余儀なくされ、設計及び製造に大幅な手戻りが発生した

事例における見切りの内容

＜プライマリーベンダーの見切り＞
顧客のビル移転との関係で稼働時期が決められていたこともあり、パッケージ製品の適用による短期開発こそが最善の開発方式であると思い込んでいた。フィット＆ギャップ分析を意識はしていたものの、顧客側の仕様決定が度々滞る事態も発生したため、パッケージ機能を活用することで、顧客の要求仕様は概ね実現できるという直感的な見切りを行うことで、上流工程を完了させてしまった

＜サブベンダーの見切り＞
採用しているパッケージ製品の製造元がプライマリーベンダーであったこともあり、プロジェクト推進過程で発生した問題解決は迅速に行えるはずという判断から、上流工程におけるエキスパート投入を重視しなかった

本来の見切りの考え方

・パッケージ製品の機能に依存したシステム開発を行う場合は、製造工程を圧縮してでも、上流工程で業務要件とのフィット＆ギャップ分析を適切に実施すべきである。分析結果という明確な証跡をもって上流工程を完了することで、例え上流工程では見えない業務要件などが潜在していたとしても、中・下流工程で方式全体の見直しを行う場面で、コストや納期を含めた顧客やプライマリーベンダーとの折衝が容易になるものである
・自社、他社を問わず、パッケージ製品にかかわるエキスパートの投入や迅速なQ/A対応を実現できる体制作りは、パッケージ機能に依存したシステム開発の場合、避けて通ることができない重要な準備タスクである
・自社製品とはいえ、企業の合併、吸収が頻繁なIT業界では、必ずしも製品のサポート体制が国内になるとは限らない。製品のサポートレベルについては、具体的な水準や実現時期を明確にしたうえで、プロジェクトへの参入時点で、プライマリーベンダーと体制に関する調整を済ませておくことが重要である

捉えるべき兆候

・機能設計やプログラム設計書の記載内容に曖昧さが残る
・パッケージ機能と業務要件とのマッピングに予想以上の時間を要し始めた

対処例

・業務要件とのギャップを埋めるノウハウを有する技術者を最優先で確保するようプライマリー・ベンダーに要請する
・上流工程で決定した設計内容の実現の可否を評価できるエキスパートを開発体制に組み入れることで、製造工程以降の手戻りを最小化するよう、プライマリーベンダーに要請する
・プライマリー・ベンダーとの契約(分担、責任など)を明確にしておく

ロジェクト・マネージャと共有できていれば、パッケージの技術要素、フィット＆ギャップの進め方の両面について、ノウハウ吸収の体制やチーム運営が当初から組み込まれているはずである。

別の考慮点として、「パッケージに適した使い方や制限事項について、発注側と意識合わせしているか」という点も重要である。パッケージを利用してシステムを構築する場合は、パッケージが前提としている使い方や条件、制限に沿うように仕様、あるいは業務フローを決めていく必要がある。

本事例の場合は、発注側がパッケージをよく知らないながらも、大きな期待を寄せている。このような場合は、可能な限り早い段階で、発注側とパッケージ適用における考え方、制限を共有し、パッケージが適用できない場合、どう対応するか意識合わせをしておく必要がある。

(2)見切りの内容

要件定義が遅れながらもスケジュールを死守するために、機能設計と製造工程を並行させることはよく見られる。ただし、それは先行する部分が、サブシステムとしての独立性が高く、他システムへの影響が少ないと判断した状態にのみ可能であり、大きな手戻りが発生しないことを前提としている場合である。

本事例では、パッケージで実現できる機能を的確に整理できていなかったにもかかわらず、要件定義、機能設計、製造を並行させる事態となってしまった。ここでは、「パッケージで実現できない場合のリスクを認識していたか」という点が大きな意味を持つ。

プロジェクトとしてはリスクの認識はあったものの、パッケージが提供する機能の網羅性について受注側も過信があり、ギャップを埋めるために変更すべきは業務側であるという認識から、リスクを過小評価している。

また、納期は変更できないといった厳しい状況のなかで、開発生産性を高めるためにもパッケージを適用すべきであると信じていた。現実的にはプロジェクト・マネージャの視野が狭くなり、リスクが見えなくなって希望的観測でマネジメントをするようになっていたと考えられる。

(3)見切り判断

サブベンダーのプロジェクト・マネージャとしては、パッケージを利用することで、ソフトウェア機能の洗い出しが簡便に済ませられることを効率化のための見切り根拠としたかったようである。

しかしながら、プロジェクトに課せら

れた戦略的なミッションとして、「自社内に当該パッケージ・ソフト利用者を育成する」、「パッケージ・ベンダーとの協業体制を確立する」などがあることを考慮すると、プロジェクト運営のために必要な事項が明確になっておらず、見切り判断を誤っていると言わざるを得ない。

また、プライマリ・ベンダーが提案したパッケージでありながら、そのパッケージで実現している機能の検証にプライマリ・ベンダーからのサポートが十分に得られない状態であった。結果的にはプライマリ・ベンダーに対して、パッケージの検証に関する課題や対策を申し入れ、パッケージの機能充足度を検証するための体制を急遽、強化する対策を打っている。プロジェクト進行上のプライマリ・ベンダーとの役割分担がより早い段階で明確になっていれば、プライマリ・ベンダーへの要請事項もより早く明らかにできる可能性が十分にあったといえる。

7.4 熟練マネージャによる座談会

上流工程にあるプロジェクトについての問題認識や現場の様子、進め方のコツについて、プロジェクト・マネジメントの豊富な経験を持つプロジェクト見える化部会の委員で座談会を開いた。企業に所属し、プロジェクトの現場で手腕を発揮されているマネージャの生の声を参考にしてもらいたい。出席者6人のプロフィールは**図表7-3**の通りである。

――上流工程におけるプロジェクト・マネジメントの難しい点は？

D氏　顧客の要件が具体的ではないなかで、要件を特定していくところ。また、人が揃わずプロジェクトの全体が把握しきれていない段階で、いろいろなことを決めながら進めなければならないところ。顧客の要件を把握するということと、社内のメンバーをアサインして体制を作っていくという同時進行自体が難しいと思います。

C氏　上流工程と言っても、例えばシステム化企画からまとめていくように最初から要件を掘り起こしながら進めるパターンと、顧客の要件がすでに定義されていて基本設計から始めるパターンの2つがあります。その2つのパターンで、やり方の違いは何でしょうか？

F氏　やり方はそれほど変わらないでしょう。ただ、後者のパターン、すなわち要件定義が終わり、そのあとにプロジェクトを受注する場合のほうが、リスクが高いと思います。要件定義をやった会社がそのあとの設計を受けていな

い理由が何かあるはずなのです。

私の経験上、後工程をこちらが受注する段階で、要件定義がきちんとできていたというケースは多くありません。一般的に要件定義が完了しているほうが楽なように思われがちですが、現実的には体制を作る時間がずっと短くなっているし、「もう要件定義は終っているのだからすぐ次工程ができるだろう」と顧客も思っているので、むしろリスクが高いのです。例えば、方式設計や非機能要件などが、すっかり後回しになっていることもあります。

C氏　上流工程の難しさとして、「顧客に決めてもらわなければならない」という点もある。決定権が顧客側にある中で、どのようにコントロールして、スケジュールを決めていくのか、というところはやはり難しい。要件をきちんとドキュメント化すればよいが、「基本設計のフェーズも自分たち（受注側）がやる」というプロジェクトの場合、受注者側のプロジェクト・メンバーがドキュメントをしっかり書いてくれないこともある。

F氏　要件定義は顧客に納めるものである、という考え方が必要になると思います。そういう関係を顧客と作っていくことが大事。成果物を何のために作っているかというと、自分のために書いているのではなく、顧客に納めて、

図表7-3●座談会出席者プロフィール

年齢	47～60歳
IT業従事年数	20～39年
所属する企業のカテゴリ	システム・ベンダー、ハードウェア・ベンダー、教育・研修サービス
職種カテゴリ	品質保証部門、PMO、教育研修、経営企画
担当経験規模	50人月～3000人月
担当プロジェクトの特性（業界）	銀行、証券・投資信託、生命保険など14業界

「できました。できていないところはここです」ということが言えるようにするためのものなのです。

B氏　ソフトウェア開発はエンジニアリングであるべきですが、実態はマニュファクチャリングの域を出ていません。やはり属人的な力に頼っているところが多分にあります。ドキュメントなども、個々人の成功体験に基づいて作られているものが横行しています。それで問題なくできればいいのかもしれないが、会社としてもどこまで個人に責任を問うのかということを考えると、成り立たなくなるし、業界の発展という観点からも、エンジニアリングの考え方を導入しないといけない。

顧客の要望が100あるとしたら、要件をヒアリングしてプロジェクトがアウトプットできるのは100以下。そこには差分が必ずあります。先ほどの話のように、後工程を受けるということは、そ

の差分というリスクをかぶっているということを、顧客もベンダーも双方で理解しておくべきです。上流工程だけを請負って成果物を納めているコンサルティング会社がたくさんありますが、それらのアウトプットには、これで本当に次工程ができるのかって思われるものがたくさんあります。

A氏　公共系のプロジェクトでは、通常、工程を分離して発注します。上流工程の要件定義のアウトプットは、あえて解釈が幅広くできるようになっていて、100%以上のことが読み取れるようになっています。もともと実現しようとしていることよりも、大きく書いてあることがある。システムを構築する際、要件のブレを抑えるために、ベンダー側のビューを通して顧客のスコープを置き換えてしまうということも1つの手段です。そもそも実現すべきことが数億円規模であるのに、数十億円規模のことが読み取れる内容で書いてあるわけですから、自社の実績と提案で顧客と折衝し、必要な読み替えを行うことで対応可能な範囲を特定して、顧客を説得していくというのが重要です。

ベンダー側のビューで説得するというのは、具体的には実際に構築したその顧客の業務に一番近いシステムのデモを見てもらい、顧客側の要望を満たしているかどうかを検討して枝葉をそいでいくこと。大きい案件では、最初に要件定義をしたコンサルティング会社にも入ってもらうようにします。

E氏　少し話しを戻しますが、上流工程の難しさは「サービスインまでに時間がある」ということではないかと思います。契約が結ばれていませんでした、要件が決まっていませんでした、という話はたくさんありますが、サービスインのときに要件が決まっていない案件はない。

なぜ、上流で決められないかというと、「時間があるから、これでいいや」とか「まだ変えられるだろうから仕様を変えてくれ」という話が横行しているからです。現実問題として、そういう不確定さがあるなかで、その後、「本当にやっていけるのか？」と自問すると、実はちゃんと見通せていなくて、受け入れてしまったあとで破綻してしまうことも多い。

例えば家を建てるときに、間取りが決まらないまま進めたらまずいですが、壁紙の色などはあとで決めてもいい。そういう不確定要素はあってもいいことだと思いますが、ソフトウェア業界では、どこまできっちりやったらいいか、という部分がはっきりしていません。

F氏　契約を取り交わしたうえで受注生産品を作るのだから、要件や仕様が決まっていなければできないっていうこ

とを「一般的な商慣習」として組立てていかないと、ダメなのではないでしょうか。

B氏　一言で言ってしまうと、業界的に買い手も売り手も成熟度が低いということなのですね。SLCP（Software Life Cycle Process）とか出てきているものの、相変わらず買い手側と売り手側の常識に乖離があるので困っています。ソフトウェアが見えないからなのかもしれません。

E氏　そこには国民性みたいなものもあって、物事はっきりさせるのはいかがなものかというような文化があるし、頑張って何とかしますというのを良しとするような傾向もあると思います。別のベンダーがあきらめた案件を「我が社が頑張ってやらせてもらいます」なんて、胸を張って言うのはおかしいと思うのです。別のベンダーが、どういう理由でやめたのか、技術面や戦略面などいろいろあるとは思いますが、なぜ営業部隊はそれを受けたのか戦略的な理由が分からないことが多い。そのような営業の仕方しかできないところが、まさに成熟度が低い証拠なのかもしれません。

――顧客側も含めて、上流工程できちんと決めるべきこと、建物でいうところの“間取り”を決めないといけないということが分かっていないということですね？

B氏　売り手のコンセプトがはっきりしていない、という面もある。建築であれば標準的な間取りならお安いですよ、フリー設計だと高くなりますよという売り方もある。高くなる理由が付けられる。

　ソフトの場合は、そもそも何もないところにあるように見せて、高く売ろうとしている。例えば、基準ソフトウェアがあるのだったら、そこから、標準に対して追加機能を付けるなど高くなる要素、安くなる要素を整理すれば分かりやすくなる。そういうことができずに、しのぎを削って赤字でも取りに行くから、ベンダーのお互いのつぶし合いにしかならない。

E氏　ソフトウェア製品を作って発表するときに、それを使うと開発工数が半分になって、効率が2倍になるなんて宣伝しますが、実際世の中にそんなにうまくいったプロジェクトなんて見たことがない。自分たちがやっていることに、いったいどれだけ責任を持っているのかというような気持ちになる。

――ソフトウェア業界の問題がそういうところにあるのかも知れませんが、そういう中でもプロジェクト・マネージャはプロジェクトを進めないといけないですね。

F氏　要件が決まらない難しさは皆さんの認識の通りなのですが、もう1つ重要な要素があると思います。それは、プロジェクトが始まったばかりなので、適した人材が分からないということです。しかも、上流工程は分業ができないという難しさがある。実は、下流工程になっていくといろんなことが決まってくるので、サブシステムに分けるとかして分業ができる。しかし、上流工程は分業ができないし、スキルを持った人材も少ない。それなのに要件定義の時間を短くするために要員を投入してしまうところがある。そうすると大抵失敗する。

実は上流工程の中でも、その前段には商談があって、「来週までに発注を決めてもらわないと困ります」と言っても、顧客は、「そうは言っても常務会は来月だから」という調子でどんどん遅れたりする。そうすると、本当は要件定義に5カ月あったものが3カ月になったりするのです。そういうとき、どうするかというと、最初は3人でやろうと思っていることを5人でやろうとします。そのアサインされた5人のうち、何人かはそのプロジェクトに合ったスキルやノウハウを持っているわけではない。でも、人月数の世界で数字を合わせるから「できるでしょ？」っていう話になる。最終的には、「まだ先があるから大丈夫」というような言い方になって、未確定要素をさらに先送りして、結局“きりもみ状態”に陥ってしまう。たとえスキルのある人が入ったとしても、上流工程は分業できないから実は思った通りには改善できない。つまり、「3人で5カ月」と「5人で3カ月」とは違うってことを、皆忘れてしまっているのです。

B氏　その辺りはプロジェクト・マネージャの力として、どこまでで見切るかっていう判断力が必要になるのでしょうね。課題の棚卸しができるようなツールリストを作って先に進む。そこで決めておくべきことがあるにもかかわらず、「まだ時間があるから」という発想に巻き込まれてオーバーランしてしまう。そこの基準がはっきりしているかどうかですね。その見切りの基準が、誰もが同じ基準でやっていかないとおかしいのではないかと思うのですが、ソフトウェアの世界でははっきりしていない。間取りは必要だが、壁紙の色は後でもいいという基準がない。エンジニアリングが導入されていない。人の経験に依存しているから、常に失敗プロジェクトは繰り返されるということになってしまう。

――ダメなプロジェクトの特徴について教えてください。

E氏　よくある現象が2つあります。1

つはやたらと管理項目が多いということ。もう1つは「うまくいってます」という報告。これが危ない。つまり、嘘をついているか、分かっていないかのどちらかだと思うのです。

管理項目が多いというのは、いろんなチェック項目があったり、いろんな指標でチェックがされたりしているということ。上流工程に必要な指標がそんなにたくさんあるわけがない。結局、いろんなチェックをして自分はリスク管理やプロジェクト管理をやっているのだと誤解している。というか、思い込んでいる。自己満足の世界なわけです。

F氏　プロジェクト計画書が、最初からしっかりまとまっているのって、危ないのではないかと思ったりします。あるいは、別のプロジェクトで書いたのをそのまま引っ張ってきて、「前回はこれでうまくいきましたから」って言って、できたようなつもりになってしまうケースです。プロジェクト計画書はある期間とかフェーズによって見直して、きちんとしたものに組み上げていかなければならない。そういうことが分かっていないと、やたら管理項目ばかりが多くて、ミーティングばっかりやっていることになる。

むしろ、「できてないところがあります」という報告のほうが、きちんと見えているのだなってことが分かって安心です。それを顧客と共有できているかどうかという事のほうが重要なのです。逆説的な議論になってしまいましたが、もちろん計画がないのはNGだし、この工程の終了基準が決まっていない、さらには目的が定義できていないというのはダメです。あとは、要所で「振り返り」をしているということが分かればよいと思います。

B氏　自分の考えやサクセス・ストーリーを語ってくれる人がプロジェクトにいると進めやすいですね。そういう人がいないときは危ないなと感じます。プロジェクト・マネージャの資質の観点から見ると、失敗したプロジェクト・マネージャは必ず失敗を繰り返す。だからそこをしっかり見ないといけない。

A氏　プロジェクトのアセスメントでは、現地、現物を見て、プロジェクト・マネージャの報告とのギャップをいかに早く見つけるかということが重要だと思います。顧客との交渉ごとをいつまでにどう決着つけるのかということがマイルストーンとして管理されているかどうかということです。プロジェクト・マネージャが自分に責任転嫁されないように考えているのは全くダメです。プロジェクト・マネージャをモニタリングして、いろいろな人のクレームを聞けるようにしておくとよいでしょう。たとえば、「顧客の要望に対して、プロ

ジェクト・マネージャは何もしていない」といった意見です。

E氏　会社で失敗プロジェクトを分析したところ、1つの傾向として、プロジェクト・マネージャを除いて「7〜8割のメンバーが比較的早い段階で、このプロジェクトはうまくいかないと感じていた」という皮肉な結果が出たことがありました。感覚的な意見ですが、先ほどの現地、現物のほか、現場の人の話というのも、適当なところを聞いて回ると問題の所在を探ることができます。メンバーの話を聞くことがアセスメントでは重要になります。

――プロジェクト・マネージャの資質で必要なものは？

D氏　チーム・ビルディングができる人が望ましい。徐々に人集めをして、目標を共有することができて、チームとして取り込むことができるということ。部下のことも分かっていて、状況も把握できるということになる。顧客との課題解決に一人称で動けるというようなことです。「計画書や手続きに従ってやればいい」ではNGですね。あの人だったら、動いてくれるという信頼感があるということです。

E氏　チーム作りという観点ではいろいろ方法があるとは思いますが、コミュニケーションのとり方が難しくなってきた時代に、ルールやマニュアルでそれを補おうというような発想ではダメです。あまりにも数字になっていて、本当のところが忘れ去られているような気がします。

――プロジェクト・マネージャが自分で背負ってしまったり、逆に部下に丸投げしてしまったりということもあるようですが？

F氏　率先垂範するから認められて、部下もついて来るという発想がある。小さいプロジェクトならそれでもいいのですが、ある程度大きくなると、プロジェクト・マネージャがどんどん下に降りていってしまって、まわりを見られなくなってしまったりする。だからといって、プロジェクト・マネージャが下に降りていかずに、上から「おまえダメじゃないか」と言うだけではダメなのです。ライン・マネジメントとプロジェクト・マネジメントの違いになるかもしれないが、マネジメントというプロセスの中で、リーダーシップもとることができるバランス感覚のよいプロジェクト・マネージャを育てていくということが、業界的に求められているのだと思うのです。

A氏　日本のライン・マネジメントって一直線に縦のラインになっていて、マトリックスとかになっていないですよね。ライン・マネジメントって、「おま

えに任せたぞ」ということが実は丸投げのことで、その中で部下がもがいている。任せられた方から見ると、「できません」と言うことが、期待を裏切ることになっている。きちんと横のラインがあって、別の比較感とかがあれば、そこがおかしくなっているということが分かるはずなのですが、そういう柔軟性のなさが問題であり、怖いところだと思います。

E氏　体育会系の動きのマネージャは要注意ですね。ある目的に向かってまっしぐらになるのは、一見頑張ったように見えるが、本来はそうではないのではないか、と思うところがあります。

F氏　率先垂範して成功しているマネージャに、いったいどこまで任せたらいいのかというのが難しい。インタビューなどの制度で、プロジェクト・マネージャを育成するという観点が重要になると思います。

B氏　プロジェクト・マネージャが現場を見ているかどうかがポイントです。大丈夫だと口では言えるけれど、いろいろ聞いてみると、発言に「多分」という言葉が入ってくる人は信用できない。PMOがプロジェクトのすべてを見に行くことは難しいけれども、仕組みや肝が分かっていることが大事です。その肝が分かっているのが本当のプロジェクト・マネージャでしょう。

F氏　上流で、そのシステムのミッションクリティカルが何なのかを定義できることが重要だが、それができないプロジェクト・マネージャが結構たくさんいる。

A氏　そこでアカウンタビリティという言葉が出てくる。システム・サービスを提供するときに、もしくは動かなかったときに、企業であれば経営者が株主に、自治体であれば知事が住民に説明しないといけない。そういう説明責任を顧客が担っているということを、プロジェクト・マネージャが認識できていないということです。

――上流工程は時間があるために、なかなか危機意識を持てないとのことですが、プロジェクトに危機意識を持たせるコツは何かあるのでしょうか？

B氏　ビジネス・マインドが非常に強くて、ビジネスライクに仕事をしていくというのも重要。「利益を確保するために何でもする」ということが危機感を持たせるには最適な気がします。そういう能力を要求される企業は強くなる。なんとかなるさっていうような企業文化の中では、そういう風にはなれない。

F氏　最後は評価の仕方になってしまうかもしれないけれど、プロジェクト・マネージャの評価の仕方を変えるやり方もあります。ある会社では納期より

も早く終るとボーナスが出る。もちろん逆に納期遅れでボーナスが減ることもあります。短期的な見方で評価しては危ない部分もありますが。

一方、プロジェクト・マネージャをプロジェクト・マネージャとして育てる仕組みがあるのか、という疑問があります。SEとして優秀だと、「次はプロジェクト・マネージャをやってみるか」という話になるのですが、それって本当はあまり良くないのかもしれません。プロジェクト・マネージャとしてのキャリア・パスが定められてプロジェクト・マネージャとして育成されれば、意識が違ったものになるかもしれません。そういう事例はありませんか？

E氏　ある会社では、資格認定制度が導入されています。プロジェクト・マネージャの能力、資質、業績、向上心などを他部門長、役員がヒアリングし、総合的に判断している。また、非常に難しいプロジェクトを担当しているプロジェクト・マネージャ、キャリアを積ませたいプロジェクト・マネージャに対しては、組織の枠組みを超えたメンター制度により育成を図っている。従来のようなルール、マニュアルの順守というプロジェクト・マネジメント・スタイルに加え、経験、勘所というような、曖昧ではあるが重要な内容を事例を通して伝授し、育成を図っている。

B氏　日本の文化としては、山がないけど、谷もないから、やらずに済ませようというところがある。一所懸命やって失敗するより、何もしないで失敗しない人の方が上になっていたりするところがある。

F氏　個人が実績通りに評価されるのは良いと思います。一方で、あまり個人の評価に結びつけ過ぎると、いままで日本的にうまくやっていた「みんなで協力して達成する」という、良い部分がなくなる恐れがあります。どうやって、そういう良い部分を残しながら、制度として成熟させていくかということが課題だと思います。

――顧客に危機意識をもってもらうコツは？

C氏　危機意識の醸成というわけではないのですが、今ある会社でやっていることは、過去の具体的な事例を分析して、「要件定義など、上流をきちんと押さえておかないと失敗する」ということを知ってもらうようにしています。そのうえできちんと設計書をレビューする、プロジェクト・マネージャが具体的レベルで報告するといったことを課しています。現場にPMOが入って、アセスメントを進める際に、アセスメントの結果よりもそういう活動そのものが効果をあげるきっかけになっている。

昔は各人が自分でそういった問題に気付こうと努力していたように思うのですが、今は、気付かせてもらえると思っている人が多い気がします。昔は、自分が知識を得るためにいろいろ探し回った。今は、与えられたものを理解することしかやってないような気がする。そういう人に何を言ってもダメな気がするのです。

F氏　プロジェクトを止める権限をプロジェクト・マネージャに持たせることがポイントではないかと思います。そうするとプロジェクト・マネージャの意識付けになるのではないでしょうか。やること、進めることが至上主義になっているが、プロジェクト・マネージャが止める権限は自分にあるのだということを認識したら、変わるように思えます。

――では、上流工程のプロジェクトを任されているプロジェクト・マネージャのみなさんへのアドバイスをお願いします。

A氏　説明責任が重要。単に、説明するということではなく、自分たちがやったことを皆に説明できること。何か起こったときに、どう説明できるのかということ。

B氏　重点志向、説明責任が重要。一歩先を見た段取りのうまさ、一歩先に何が起こるのかという風を読むうまさを養ってほしい。意思を持って見るように。

C氏　無理な計画は立てないでほしい。詰めれば詰めるほどできそうな気になるが、ダメなものは早くダメだと言えるようになってもらいたい。

D氏　チーム・ビルディングという意味で、客も上司も含めて、同じ船に乗ってくれる人を集められることが大切でしょう。

E氏　物事にはプライオリティがある。全方位ではなく、プライオリティ付けをして3つだけでもやるようにするとよい。

F氏　時間があると思わないこと。後々、どう響くのか、先送りしていいのかってことを気にしながら進めてほしい。

第8章

プロジェクトを取り巻く環境変化と課題の俯瞰

近年、プロジェクトの失敗あるいは金融・通信業界などの社会インフラを構成するITシステムの障害が多発している。その社会活動への影響の大きさから、社会的問題にすらなっている。同時に、IT業界の企業経営にも多大の影響を及ぼしている。

この章は、IT利用者/IT業界双方に影響を及ぼしているプロジェクトの実情と課題を、プロジェクト失敗の実態から分析・把握し、問題の所在を明らかにすることを目的とする。さらに、プロジェクトが抱える課題の「見える化」を図り、今後のプロジェクトにおける変革の方向を俯瞰しようとするものである。

8.1 プロジェクトを取り巻く環境の変化とその特質

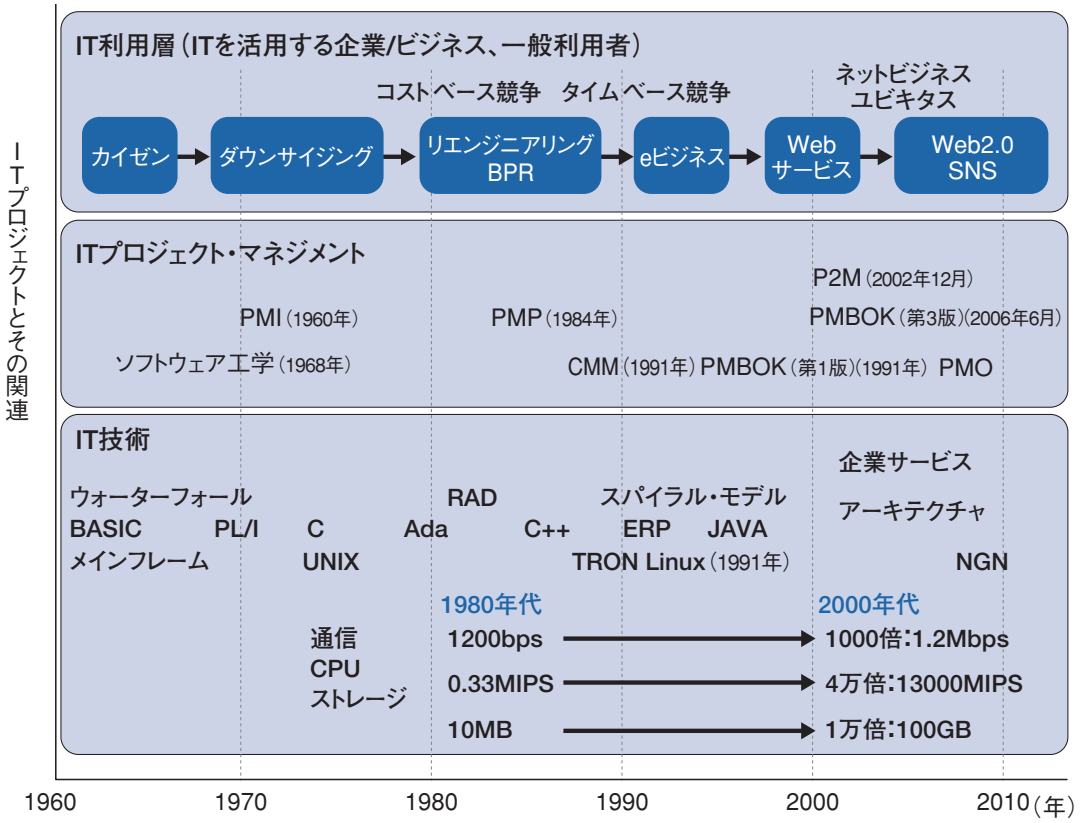
長い歴史と経験を持つ他業界と異なり、IT業界は1940年代のコンピュータ出現以来、まだ60数年しか経っていない。しかし、その間にITプロジェクトを取り巻く環境は、ITとその利用層の相互連鎖を伴いながら、他業界に類を見ないスピードで変化・拡大を遂げてきた（**図表8-1**）。

IT利用層の目的達成を使命とするITプロジェクトはいまだ成長途上にあって、乗り越えるべき課題は多い。また、今後も続くIT業界の変化とそこから派生する問題解決のために、プロジェクト・マネジメントが果たすべき役割は多く、その使命は重い。

8.1.1 ITプロジェクトを取り囲む環境の変化

ITプロジェクトの使命は、図表8-1に示したように、ITとIT利用層の間にあって両層の整合をとり、IT利用層が目的とするビジネス/サービスを実現することにある。情報システムの基盤となるITは、その基本要素であるハードウェアがムーアの法則に象徴されるように、この20年間で数千倍から数万倍のスケールで進化してきた。このハードウ

図表8-1●ITプロジェクトを取り巻く環境の変化



ェアの進化を受けて、（残念ながらハードウェアの進化のスピードに追従できていないが）ソフトウェア、開発言語、開発方法論なども進化を遂げてきた。

一方、IT利用層は、1980年代に米国でブームとなった「カイゼン」を皮切りに、1990年代以降、ITの導入も含めて業務を抜本的に見直すBPR（Business Process Re-engineering）により進化した。続いて、ネットワーク化された業務システムを提唱したeビジネス、2000年代に入ると、ITとインターネットを利用したWebサービス、Web 2.0の出現など、ビジネス・モデルはITの活用で急速な進化を遂げてきた。

8.1.2 マネジメント対象の「曖昧性」と「不確実性」

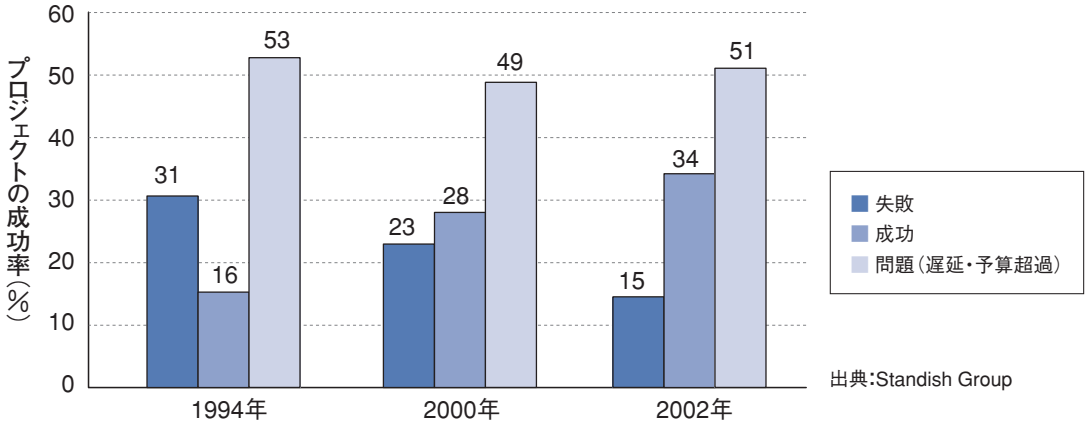
IT利用層のビジネス/サービスのイノベーションと、ITの急進性に追従する難しさから、プロジェクト・マネジメン

トには以下のような「曖昧性」と「不確実性」が存在する。

(1)ビジネス競争を背景に、猛烈な速さで変化するビジネス・モデル、パッケージ化(ブラックボックス化)されたビジネス標準モデルの「曖昧性」と「不確実性」
(2)eビジネス、Webサービス、Web 2.0などに代表されるビジネス/サービス・イノベーションが持つ「曖昧性」と「不確実性」
(3)IT個別技術の急速な進化に対して、システム・アーキテクチャ、システム統合インプリメントなどシステム系技術の遅れから発生するプロジェクト・マネジメントの「曖昧性」と「不確実性」
(4)IT利用層、IT、プロジェクト・マネジメントのいずれの層でも、その中心に存在する人間が持つ「曖昧性」と「不確実性」

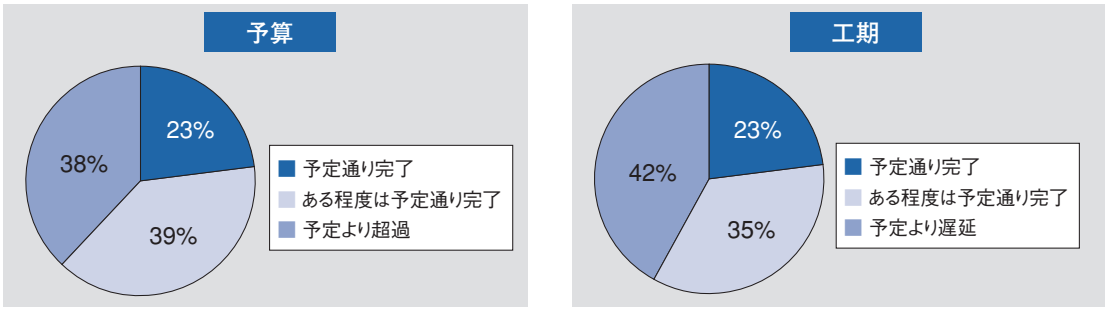
図表8-2●ITプロジェクトの成功率

プロジェクト・マネジメント導入で、10年弱で成功プロジェクトは倍増、失敗プロジェクトは半減



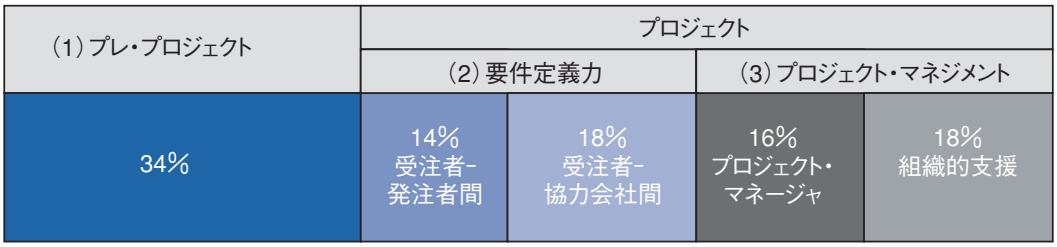
出典:Standish Group

依然、改善されない成功率。再構築システムの予算超過は38%、工期遅延は42%



出典:日本情報システム・ユーザー協会「2005年度企業IT動向調査」

図表8-3●43件のITプロジェクト危機発生要因の混入時期と概要分析



出典：拜原正人『日経ITプロフェッショナル』「プロマネ失敗学」(2006)より

8.2 プロジェクト・マネジメントの現状

「曖昧性」と「不確実性」を内包するIT環境の中で、米Standish Group Internationalのレポートによると、ITプロジェクトの成功率は1994年に16%であったが、2002年には34%とおよそ2倍となった（**図表8-2**）。しかし、依然として51%のプロジェクトが遅延、予算超過などの問題を起こし、15%のプロジェクトが中止に至っている。ちなみに、成功率倍増の要因は、プロジェクト・マネジメントの浸透にあると言われている。

一方、日本情報システム・ユーザー協会（JUAS）による2005年度の再構築システムの予算超過プロジェクトは38%、遅延プロジェクトは42%となっている。

日米いずれにおいても、60%から70%のプロジェクトが目的を達成できていないことになる。

8.3 ITプロジェクト失敗の問題の所在

このようなプロジェクトの遅延、予算超過あるいは失敗発生の要因は、どこにあるのだろうか。

1970年代から2006年にかけて国内で発生した43件の危機発生（遅延、予算超過、中止）プロジェクトを分析し、プロジェクト危機発生の混入要因（原因）を解析した結果を**図表8-3**にまとめた。リスク要因の混入時期は、プロジェクトの目的を明確に定義して発足させる「プレ・プロジェクト段階」が3分の1、システムの要件定義を明確にする「上流工程」が3分の1と、プロジェクトの成否は、ほぼ上流工程までの段階で決まっていることが分かる。ちなみに、混入したリスク要因は、若干の例外を除き、すべてシステム・テス

トの段階で顕在化している。

次に、リスク混入時期別の主要なリスク要因の概要を述べる。

8.3.1 プレ・プロジェクト段階の要因

プレ・プロジェクトの段階では、主に3つのリスク要因があった。

1つめは、猛烈な速さで変化・推移するビジネス環境で、発注側がリスク要因を内在したままに提示した、曖昧で不確実なビジネス要件に起因するものである。

2つめは、受注側の売り上げ至上主義の弊害によるものだ。1990年代に多くの日本企業がこぞって導入した成果主義の弊害ともいえる。フィージビリティを十分に検証しないまま、売り上げ数値のみの判断で受注して失敗を繰り返したケースである。これは、多くのIT企業に多大な赤字経営をもたらした。

3つめは、受注側のプロジェクト・マネージャの選定誤りだ。43例で最も多かったケースである。プロジェクト・マネージャの人材不足あるいは受注プロジェクトの性格に合わせたプロジェクト・マネージャの人選が行われていなかったことに起因する。本質的には、IT業界のプロジェクト・マネージャの育成不足に起因する人材不足、およびプロジェクトの特性とプロジェクト・マネージャのスケジューリング機能の不在によるところが大きい。

8.3.2 要件定義力の不足

プロジェクト開始後では、要件定義力が不足しているためにリスク要因が混入するケースが多い。主な要因は2つある。

第1に、発注者/受注者の要件定義の齟齬から発生したリスク要因である。発注側の組織内ステークホルダーの業務要件調整/設定能力が低いことや、受注側の対象業務知識の不足、それに起因した両者間の業務要件設定コミュニケーションの脆弱性によってもたらされることが多い。

第2に、受注側と協力会社のシステム要件構築力が不足しているために生じるリスク要因である。上流工程では業務機能要件が中心となりがちで、中流・下流工程で顕在化する非機能要件*1の検討が漏れやすい。システム全体を俯瞰する能力を要する非機能要件の検討漏れが、時間的な余裕がない下流工程で顕在化すると、プロジェクトに危機をもたらす。

8.3.3 プロジェクト・マネジメント力の不足

要件定義力とならんで、プロジェクト・マネジメント力が不足した場合も

リスク要因が混入しやすい。

まず、プロジェクト・マネージャのマネジメント能力不足が挙げられる。IT業界全般において、プロジェクト・マネージャの能力不足は大きな問題である。IT業界の持つ「曖昧性」と「不確実性」の特性からくる、複雑なプロジェクト・マネジメントの難しさに起因している。しかし、ステークホルダーのマネジメント能力、全体的視野に立ったシステム・マネジメント能力、ドミナント・アイテム(プロジェクトの成否を左右する支配的な要因)を抽出・解決する/させる能力など、基本的なプロジェクト・マネジメント能力の不足が大きい。

また、プロジェクト・マネージャへの組織的なサポートが不足している面もある。受注側でプロジェクト・マネージャの位置付けが確立されていないため、プロジェクト・マネージャに対する権限委譲、プロジェクト特性（ステークホルダーの質·量、プロジェクトの規模/要員数、開発期間の長短など）に合わせたプロジェクト・マネージャ・スタッフの付与といったサポートが欠けている。IT業界の多くのプロジェクト・マネージャは孤独である。

以上のような結果から、プロジェクトの失敗をプロジェクト・マネージャに帰している経営陣は、プロジェクト・マネージャの人材育成も含めて、自らの責任をかえりみる必要がある。また、多くのプロジェクト・マネジメント方法論が、プロジェクト発足後の開発プロセスを指向していること、上流の要件定義領域の問題指摘に関して定性論が多いこと、さらに現実の企業内で上流工程に投入されている費用割合が16%(JUAS調査）に過ぎないことなどから、上流工程に失敗回避の大きな余地があることが分かる。

8.4 上流工程の課題

上流工程のリスク関連項目から、プロジェクトの主要ステークホルダーである発注者、受注者、ベンダーの上流工程における業務要件設定能力などの不足が明らかになった。ここでは、上流工程における受注者の視点から、上流工程で抱える課題とその解決策を考える。**図表8-4**に、上流工程と超上流工

*1 非機能要件とは、業務機能を実現する機能要件に対して、その機能を発揮させる①品質要件（信頼性、拡張性、性能、セキュリティなど）、②技術要件（システム・アーキテクチャ、開発方法、開発環境など）、③運用操作要件、④移行要件などを指す

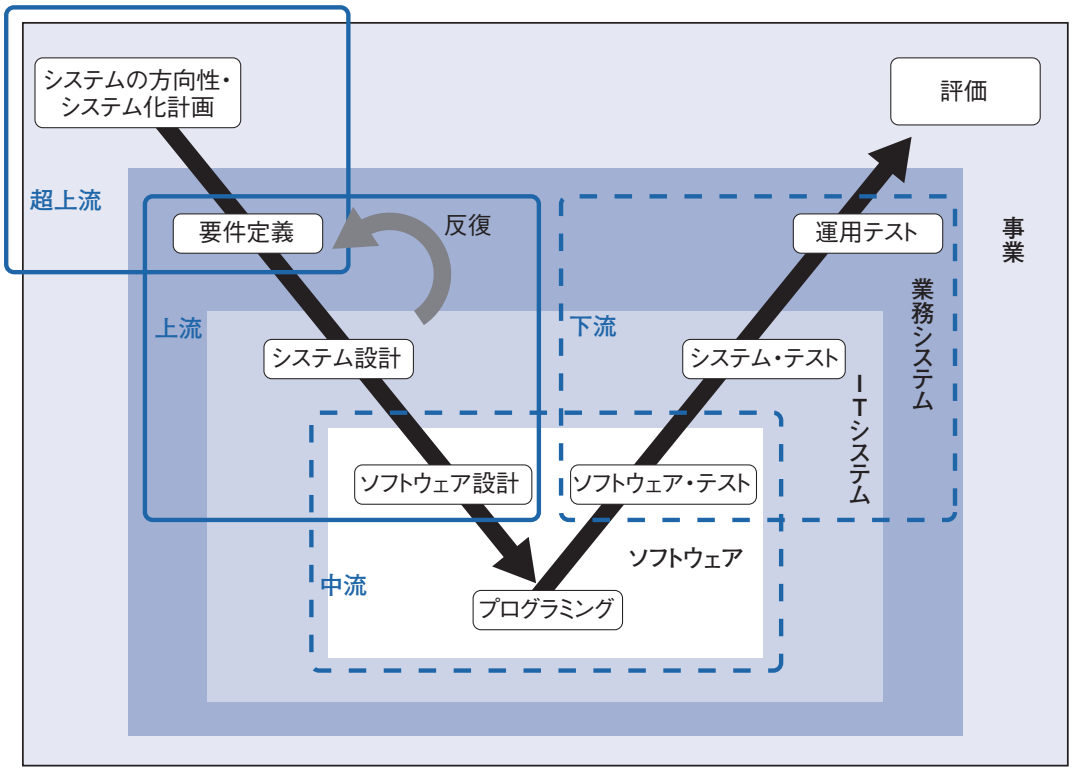
程、中流・下流工程の関係を示す。

通常、システム開発の受発注は、発注者が策定したRFP（Request for Proposal、提案要求）に基づいて、受注希望企業がプレ・プロジェクトの超上流工程でシステムの方向性・システム化計画を記述した提案書を作成し、発注者に提出する。発注者はそれらの提案書を評価して、調達先を決定し、契約を結ぶ。受注者は、発注者が策定した業務要求およびシステム要求を受けて、受注者内の超上流工程でシステムの方向性とシステム化計画を策定。さらに、プロジェクト・マネージャを決定して、プロジェクトを発足する。

プロジェクト発足後の上流工程では、超上流工程で策定したシステムの方向性、システム化計画と発注者の業務要求、システム要求を受けて、業務要件、システム化要件に落とし込む。これにより、中流・下流工程の入力要件となるソフトウェア開発機能要件および非機能要件を策定する。

ここで重要なことは、発注者の業務

図表8-4●ITプロジェクト上流工程の位置



要求、システム要求、超上流工程からのシステムの方向性/システム化計画は、すべて曖昧性と不確実性を残していることである。ウォーターフォール型開発にしろ、アジャイル開発にしろ、システム開発モデルの多くが、業務要件、システム化要件の確定を前提としている。しかし、数カ月から1年に満たない短期開発プロジェクトでさえも、競合他社の要因、企業内方針の変更、サービス環境の変化などにより、プロジェクトへの機能要件/非機能要件が変わる。それほどプロジェクトを取り巻く環境の変化は激しいのである。

この変化を受けるのは、発注者主体の業務要求/システム要求、受注者主体の業務要件/システム化要件であるが、変化がこれらすべてに平均的に影響するわけではない。パレートの法則で言われるように、20%の要因が全体の80%を支配する。重要なことは、下流工程でプロジェクトの危機をもたらす変化の要因のうち、20%の主たる要因（すなわちドミナント・アイテム）を、上流工程で早期に検出・解決することである。後工程で曖昧性、不確実性が顕在化することがプロジェクトの成功率を落とす大きな要因であり、それを「見える化」して早期に解決することが、マネジメント成功の最大の要点である。

上流工程におけるドミナント・アイテムを「見える化」するために、プロジェクトの上流から下流まで全体を俯瞰的に見通すことが重要である。俯瞰的とは、「木を見て森を見ず」の過ちに陥って、上流で解決すべきドミナント・アイテムを見過ごすことがないように、高所から全体を見ることである。全体とは、上流だけにとどまらず、中・下流からプロジェクト終了後のシステム運用・維持までを指す。

全プロジェクトの工程を視野に入れ、プロジェクトの下流工程でよく起こるような危機の火種を、上流で摘むことが重要である。そのためには、上流工程で見逃しがちなプロジェクトの非機能要件(多くが下流工程におけるシステム化要件)に十分配慮する必要がある。

中流・下流工程に対しては、超上流・上流工程で策定したシステム要件、ソフトウェア開発要求を提示する。これを受けて、中流・下流工程ではソフトウェア開発・テストのスケジュールを策定・実行。システムを完成させて発注者に提供する。これでプロジェクトは初期の目的を完遂する。

しかし、中流・下流工程のプロジェクト・マネジメントも、多数の人間によるプログラム開発、非機能要件を含めてのシステム・テストといった、本質的に分からないもの（曖昧性、不確実性）のマネジメントである。次の工程

に進むために、予測/仮説に基づいた準備作業が必須である。この段取りの巧拙が中流·下流工程のプロジェクトの成否を左右する。

通常、この準備作業は、プロジェクト・マネージャ主管で、次工程が始まる2〜3週間前から実施される。上流から引き継がれてきた「曖昧性」、「不確実性」を回避するために、あらかじめ起こり得るリスクを予測して事前検討するリスク・マネジメントとオプションの導入が効果的である。

具体的には、プロジェクトの完了時点（あるいは工程の節目）の状況からさかのぼって実施すべき作業を想定し、さらに準備作業として何を行うべきかを明確にする。このようなシミュレーションを通して、潜在する問題への気付きが可能になる。

上流工程におけるリスク・マネジメントとして、中流・下流工程に対して予測すべき事例を以下に示す。

(1)システム・テストに対する発注側との意識の違い（機能/非機能要件の確認、受注側主体/発注側主体のテスト明確化など）の顕在化

(2)パッケージ、既存システム接続など、グレーあるいはブラックボックスに

図表8-5●上流工程におけるリスク混入要因とその対策

課題		対策	
混入リスク要因	混入リスク要因の概要	曖昧性要因	
要件定義力の不足	超上流で導出された業務要求に内在するリスク要因を見落とす	●発注側ステークホルダー内の業務要求調整/設定能力向上 ●受注側の発注側業務知識の修得	
	業務要求から業務要件への変換過程でリスク要因が混入	発注側/受注側間における要件設定コミュニケーションの強化とその結果のドキュメント化による合意	
システム要件構築力の不足	システム機能/非機能要件の検討漏れ	●システム機能/非機能要件、特に非機能要件のプロジェクト危機に対する重要性の認識 ●俯瞰図の導入によるドミナント・アイテムの顕在化 ●ステークホルダー(発注側/受注側/協力会社)間の情報共有	
プロジェクト・マネジメント能力の不足	●ITプロジェクト特有のステークホルダーの「多様性」、「曖昧性」、「不確実性」を考慮したマネジメント能力が不足 ●プロジェクト・マネジメントの基本動作ができていない ●IT業界におけるプロジェクト・マネージャの育成不足	●超上流の発注側業務要求からくる曖昧性の認識とその解決 ●下流方向のシステム機能要件/非機能要件の曖昧性の認識と解決策の折り込み ●これらの工程的、質的に異なる曖昧性を統括マネジメントする機能の設置 ●プロジェクト・マネージャが対応困難な領域に対する専門スタッフの配置	

関するテスト項目・稼働積み上げ漏れの発見（テスト・データ、テスト・シナリオの漏れなど)、準備作業関連の漏れの発見（テスト環境、スケジュールの見直しが必要になる場合もある）
⑶時間不足に陥るクリティカルパスの発見（多くは並行線表化、キーパーソン不足に陥る）

8.5 上流工程が抱える課題への対策

上流工程におけるリスク要因から提起された課題とその対策をまとめるとともに、上流工程と他の工程の関係から導き出された危機回避策の「見える化」についてまとめてみる。

まず、上流工程におけるリスク要因と回避策を**図表8-5**にまとめた。上流工程におけるリスク要因とプロジェクトの本質である「曖昧性」、「不確実性」に着目したリスクの混入回避策を示している。

この章の最初に、プロジェクトの実態から、超上流および上流工程でプロジェクト危機要因の3分の2が混入していることを紹介した。また、混入したリスクは、若干例を除いて、すべて下流工程のシステム・テストで顕在化している。この事象から、以下の点を学ぶことができる。

第1に、上流から中・下流への情報の流れを清流化する必要があること。これは常に言われていることであるが、各工程の意図、特に超上流の発注者意図、目的、機能要求、制約条件など発注側の要求を、ドキュメント化して「見える化」を図ることがまず重要だ。次に、発注者要求を業務要件、システム要件にまとめてドキュメントに落し込み、上流工程から下流工程まで一貫した情報の受け渡しを行うべきである。各工程のミッションを果たすうえで必要な、情報の流れを作ることが重要になる。

	不確実性要因
	●ステークホルダー(発注/受注社)間における不確実性未解決項目の共有 ●リアル・オプション導入などによる解決スケジュールの策定・実施
	●業務要件未解決項目の共有 ●上記と同様の解決スケジュールの策定、実施
	●曖昧性も含めた非機能要件未決項目のステークホルダー内共有 ●上記と同様の解決スケジュールの策定・実施
	●超上流における発注側ビジネス/経営環境の変化からくる不確実性の認識 ●不確実性を前提とした意思決定の余地を与える、全体最適解の視点からのマネジメント ●変化に対応し得るダイナミックで柔軟な意思決定とそれを支える手法(オプションなど)の導入

第2に、これら上流工程・下流工程間のコミュニケーションを促進する環境作りが必要であることだ。工程や組織横断のPMOを設置したり、上・中・下流工程間の見える化ツールとして、俯瞰図やオプションなどを導入したりすることにより、上流・下流間のコミュニケーションを促進すべきである。

以上、IT業界のプロジェクトで現在起きているプロジェクトの実態を分析し、その本質の解明と対策についての一端を紹介した。IT業界の変化は、今もとどまることなく進んでいる。IT業界に生きる我々は、この変化に臆することなく、変化する状態を定常（願わくば変化率は一定であってほしいが）と捉えて、それをもマネジメント可能な世界へと変えていきたい。

（株式会社クロスリンク・コンサルティング
拝原正人、栗田存）

おわりに

本書では、「失敗しそうなプロジェクトを救う活動」を主眼において、プロジェクトの上流工程での「見える化」を解説した。

下流工程での見える化と同様に、「見えないものを見えるようにしたい」、「見えたものを見えたまま知らせたい」、そして「問題があれば良くしたい」という考え方のもとに「見える化」を検討した。見える化に利用できるツールとして、チェックシート、測定項目リスト、俯瞰図を作成した。また、上流工程でリスクを見出すためのリスク分類表を作成し、過去のプロジェクトのリスクとその結果の例を「見切り」事例集としてまとめた。

今後の取り組み

2005年度は下流工程の見える化の活動を行い、2006年度は上流工程の見える化のためのチェックシートなどを作成した。中期計画（**図表A**）の最終年度である2007年度では、今までの上流工程および下流工程での見える化の活動成果をまとめる。また、中流工程での見える化を加え、上流から下流工程にわたる全体の見える化の体系化を行いたい。

今後も、優れたプロジェクト・マネージャが持つ「プロジェクトを見える化するノウハウ」を形式知にし、体系化していきたい。そして、それを広く普及させることにより、システム開発における失敗プロジェクトを撲滅したい。

図表A●本部会の活動予定

	2005年度	2006年度	2007年度
研究活動の狙い	●下流工程のプロジェクトの見える化方法整備	●上流工程のプロジェクトの見える化方法整備	●見える化の体系化
アウトプット	●見える化チェックシート、症例分類表 ●対応策・手法 ●解説書	●見える化チェックシート、リスク分類表 ●対応策・手法 ●解説書	●解説書

付録

見える化のツールと関連資料

上流工程における「見える化」を実践するためのツールとして、SECは「自己評価シート」、「ヒアリングシート」、「測定項目リスト」、「事例集」、「リスク分類表」を開発した。これらを本書の付録資料として掲載するとともに、同じ内容のファイルをSECのWebサイト（http://sec.ipa.go.jp/）からダウンロードできるようにしている。

ただし、測定項目リストだけは紙面の都合により、本書の付録資料としては掲載しない。SECのWebサイトからダウンロードしてもらいたい。

自己評価シート、ヒアリングシート

プロジェクト・リスクを洗い出すために、自己評価シートとヒアリングシートの2種類を用意した。自己評価シートはプロジェクトの状況について、プロジェクト・マネージャが自らチェックするためのチェックシートである。ヒアリングシートは、プロジェクトの状況について、外部の専門家がプロジェクト・マネージャにヒアリングをするときに活用するチェックシートである。

SECのWebサイトからダウンロードできるExcel形式のチェックシートでは、各ヒアリング項目へのデータ入力を完了させると、その評価結果をレーダーチャートで表示できる。自己評価シートとヒアリングシートの使い方は、第3章と第5章を参照してもらいたい。

測定項目リスト

プロジェクト･リスクの存在を、定量的に測定するための指標をまとめた。測定項目を知識エリアごとに分類した「測定分析データ一覧表」と、測定分析データ一覧表の各項目を測定するにあたってベースとなる測定指標をまとめた「ベース尺度一覧表」がある。これら測定項目リストの使い方は、第4章と第5章を参照してもらいたい。測定項目リストは、SECのWebサイトからダウンロードできる。

事例集

上流工程で問題が発生した58プロジェクトの事例集を用意した。プロジェク

ト・マネジメントの“見切り”に着目して、プロジェクトの舵取りを学べるように構成している。事例集の使い方は、第3章、第5章を参照してもらいたい。

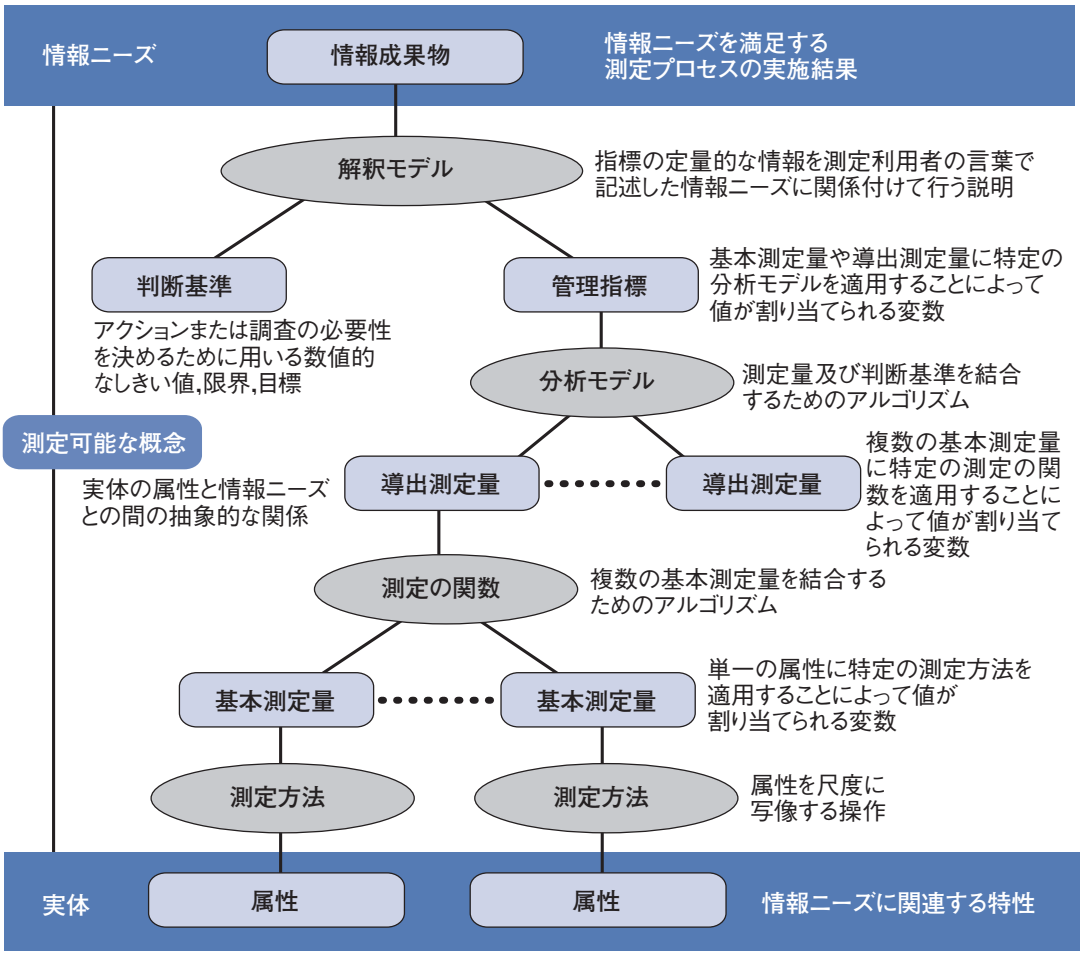
リスク分類表

プロジェクトのリスクを確認する視点（分類）を軸に、ヒアリングシート、測定分析データ一覧表、事例をそれぞれ関連付けたものである。これを用いることによって、ツールの横断的利用を可能とした。リスク分類表の利用方法は、第5章を参照してもらいたい。

導出尺度とベース尺度に関する補足

本書における基本的な尺度の考え方は、ISO/IEC15939（以下ISO15939）の参照情報モデルに基づいている。

図表X-1●測定情報モデル

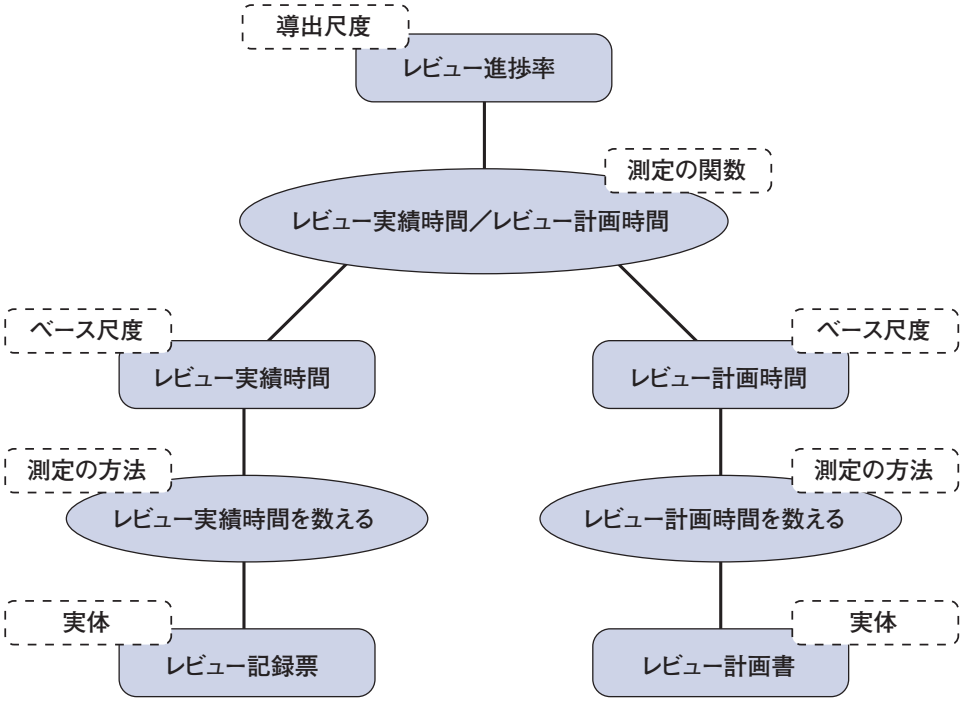


ISO15939 は，ソフトウェア測定プロセスに関する国際規格である。プロジェクト・マネジメントや品質保証などの様々な情報ニーズを満たすために、測定と分析、解釈のためのフレームワークを示している。さらに、測定と分析において扱われる情報間の関係を、**図表X-1**に示すような参照モデルとして定めている。なお、ISO15939は、国内でもJIS X0141として規格化されている。

図表X-1は、意思決定の基礎となる情報を得るために産み出される、中間情報の階層構造に関する参照モデル(測定情報モデル)である。プロジェクトで実際に測定可能なプロセスや成果物の属性（規模、工数、欠陥数など）と、意思決定のための管理指標を関連付けている。定量化された情報に基づいて、客観的な意思決定をしやすくなる。

プロジェクトに存在する様々な測定可能な属性は、定められた測定方法に従い、「基本測定量」と呼ばれる一次データに定量化される。次に、いくつかの基本測定量を関数に入力することで導出測定量が導かれる。この導出測定量を、定められた分析モデルに基づいて分析すれば管理指標が得られる。

図表X-2●レビュー進捗率の測定情報モデル



図表X-3●ベース尺度の尺度分類2

計数値	名義尺度	順序無	ゼロ点無	不等間隔	離散量	単に区別するために用いられている尺度。例えば、血液型でA型・B型・O型・AB型を、それぞれ0・1・2・3と数値に対応させたもの。数の大小には意味がない
	順序尺度	順序有	ゼロ点無	不等間隔	離散量	大小関係にのみ意味がある尺度。例えば、治療効果の判定において、悪化・普遍・改善・著効をそれぞれ−1・0・1・2と数値に対応させたもの。平均値は定義できないが中央値は定義できる
計量値	間隔尺度	順序有	ゼロ点無	等間隔	連続量	絶対量よりも数値の差に意味がある尺度。順序尺度の性質も備えている。例えば、摂氏や華氏のような想定温度など
	比例尺度	順序有	ゼロ点有	等間隔	連続量	数値の差とともに数値の比にも意味がある尺度。順序尺度・間隔尺度の性質も備えている。絶対ゼロ点を持つ。例えば、絶対温度など

プロジェクト・マネージャは、管理指標が示す情報と判断基準とを照らし合わせることで、最終的な情報成果物を得ることができる。このようにして定量的測定プロセスの実施結果を意思決定に結び付け、プロジェクトを客観的にマネジメントできるようにする。

本書では、基本測定量、導出測定量という言葉をCMMIなどで比較的幅用いられている用語で置き換え、それぞれ、「ベース尺度」、「導出尺度」と呼んでいる。

レビュー作業の進捗を測定する場合のベース尺度と導出尺度の具体的な関係を**図表X-2**に示す。

計画したレビューにかける時間に対して実際にどれだけの時間をレビューにかけたかを、レビュー進捗率として見る場合、レビュー進捗率は次の式で定義される。

レビュー進捗率＝
レビュー実績時間/レビューの計画時間

ここで、「レビュー実績時間」、「レビュー計画時間」はベース尺度であり、それぞれ「レビュー記録票」、「レビュー計画書」に記載されたレビュー時間を積算する。

また、「レビュー進捗率」は、ベース尺度である「レビュー実績時間」と「レビュー計画時間」から計算（導出）される導出尺度である。

ベース尺度一覧表の項目にある尺度分類のなかで、尺度分類2（**図表X-3**）は、統計学で一般的に用いられているデータ分類を用いている。

付録 1.自己評価シート

No.	知識エリア	チェック項目	評価基準	
S1	統合	プロジェクト計画書が作成され、レビューされているか？	●プロジェクト計画書は、プロジェクト開始から規定期限以内に作成され、レビューされていること ●プロジェクト計画書は、雛形になるような過去事例や社内標準などを基に作成していること	
S2	統合	プロジェクトの全体像を俯瞰できているか？	●今回の開発対象となるシステムを含めた、顧客側の関連システムを表わすドキュメント（システム構成俯瞰図など）を作成していること ●顧客や開発側の関係会社（協力会社など）に関係する体制図があること ●ドミナント・アイテム（プロジェクトの成否を左右する支配的要因）を把握できていること	
S3	顧客	顧客のプロジェクト目的は明確か？	●プロジェクト計画書に開発側の目的だけでなく、顧客の目的も記載していること ●顧客の目的に関する記述内容について、顧客側の資料をベースにしている、もしくは、顧客によって承認されていること	
S4	コミュニケーション	プロジェクト計画書は、全員が見ることができるようになっており、全プロジェクト・メンバーへ展開しているか？	●プロジェクト計画書の所在を全員に連絡していること ●プロジェクト・メンバーにプロジェクト計画書の内容を確認し、理解してもらうよう配慮していること	
S5	コミュニケーション	顧客、顧客側キーパーソンなどの要求仕様を十分に考慮し、必要に応じて要求仕様のレビューが行われているか？	●顧客側の要求仕様に対する理解に問題がないかをチェックしていること ●顧客からの要求仕様を整理していること ●要求仕様が有識者やキーパーソンによってレビューされていること	
S6	コミュニケーション	顧客との「仕様」「価格」「納期」に関するやりとりは文書で行っているか？	●「仕様」「価格」「納期」に関する経緯と契約が文書化され、保管されていること	
S7	組織	プロジェクト体制について、顧客も含めたステークホルダーの責任分担が明確になっており、体制上の不備不足や不安はないか？	●プロジェクト体制図が作成されており、顧客側・ベンダー側含めた全体の体制について、想定しているプロジェクトの活動目標を遂行できるメンバー・組織構成になっているかの確認ができていること ●チームごとの責任分担が明確になっていること	
S8	組織	プロジェクト内のメンバー・組織のミッションが明確で、それぞれが自分の責任を意識した行動をしているか？	●各メンバー・組織のミッションについて、各者がそれを理解し、合意していること ●プロジェクト体制図が作成できており、各メンバー・組織ごとのミッションが明確になっていること	

	マネジメントにおけるヒント	対策案
	●計画のない所には、マネジメントなし ●レビューはステークホルダー間の合意形成の場である。プロジェクト計画書のレビューを通じた合意は最初の、最も重要な合意事項である	●プロジェクト計画書が作成されていない場合は、早急に作成する。名ばかりのプロジェクト計画書にならないよう、十分な記述内容を心がける必要がある ●レビューを行った後に、スケジュール、納品物、責任分担については、レビューを通じてステークホルダーの同意を得る
	●プロジェクトの全体像を俯瞰できないと、プロジェクト計画がきちんとできていない可能性がある ●完全なものが作成できなくても、トライすることが重要である。トライの積み重ねとその後のプロジェクト・マネジメント経験の反省を通じて、俯瞰図作成レベルは着実に向上する	●俯瞰できていない場合は、まずは顧客側キーパーソンに十分にヒアリングしたり、マスタースケジュールを見直したり、関係者（たとえば接続先となる他システムの担当者）と打ち合わせを持つなどして、意見交換をすること ●各担当者へのヒアリングなどの作業を通じて、どの部分がキモになるかを把握する（ドミナント・アイテムの把握）
	●顧客との交渉を円滑に進めるために、顧客がシステム開発を通して、何を実現したいのかを把握しておくことが重要である ●顧客の目的や目標はプロジェクトのスコープに大きく影響する。そのため、顧客から示された目的や目標に対して、思い込みによって過大評価・過小評価することのないよう、プロジェクト・マネージャは顧客のプロジェクト目的について十分に確認し、明確化しておかなければならない ●プロジェクトの目的が明確になっており、顧客側の目的・目標と整合していることが、システム構築プロジェクトを成功させるうえで重要である ●プロジェクトの目的が不明確なままでプロジェクトを進めてしまうと、顧客側、ベンダー側双方でプロジェクト参画への意欲が下がり、十分な協力が得られないなどの問題が発生することがある	●プロジェクトの目的（目指すべきゴール）が明確になっていない場合、まずは顧客のキーパーソンと個別に打ち合わせをして目的・目標を聞き出しておく必要がある ●顧客側のキーパーソンが、明確なゴールを思い描けていない場合も考えられるが、この場合はプロジェクトの最初の段階で、この目的を明確化させる検討を進める必要がある。顧客側での課題認識について意見収集を行い、プロジェクトの活動方針を検討する
	●プロジェクトの概要や目的、スケジュール、スコープ、体制図など、プロジェクト計画書に記述されている内容をプロジェクト・メンバーに周知することによって、プロジェクトに参画してもらううえでの心構えを持ってもらう必要がある	●プロジェクト計画書を最新に保つようにし、特に重要な項目については必ず目を通すように徹底する ●特に、新しくプロジェクトに参加したメンバーに対しても説明することを忘れないようにする
	●顧客からの要求仕様に対して、レビュー会などの打ち合わせの場を設けるなど、直接的な会話で内容を確認するべきである ●顧客からの要求仕様は、文字にしがたい背景を含んでいることが多く、背景も含めて要求仕様をヒアリングするべきである ●顧客からの文書による依頼内容だけで要求仕様を理解した気になっていると、要求が十分に抽出されていないおそれがある	●要求内容に関して、顧客と直接打ち合わせをする ●そのためには、必要に応じて顧客から直接、説明を受けられる関係構築が重要である ●打ち合わせに出席しているメンバーに、責任者がいることを確認すること
	●口頭による約束では「言った」、「言わない」の問題になる ●文書化された内容について顧客側、関係会社（自社、協力会社含めて）それぞれのステークホルダーが合意している必要がある	●顧客とのコミュニケーションで重要な項目は文書でやり取りする ●口頭による依頼が慣習化している企業もあるので、文書で交わす手続きを慣習化させる努力が必要な場合もある
	●想定されている活動計画を遂行するにあたり、以下のような観点でプロジェクト体制を確認すること ・プロジェクト推進上必要な権限を持っている人がいるか ・ 必要な技術・スキルを持った人や組織が入っているか ・システムを利用する利用者側を取り仕切ることができるメンバーがいるか ・プロジェクトの責任者が明確になっているか ・顧客側とベンダー側とで、体制上のパワーバランス、人数バランスに偏りがないか（質・量でバランスが取れているほうが望ましい）	●分担と責任が明確になっている必要があるが、実施能力がない場合や能力が不足している場合は、外部からの人材補充などにより再構築する ●顧客側責任でプロジェクトが遅れる場合は、納期やシステム開始を遅れることについて合意しておく
	●いつまでに誰が何を行わなければならないかを確定することが重要である ●各メンバーにもミッションの内容を確認してもらい、合意してもらう必要がある。ミッションが不明確な場合、のちのち作業範囲、責任分担について問題になることがある ●各メンバーに対し、そのミッションについて担当可能かどうかを判断してもらっておく必要がある。この判断をしてもらうことによって、プロジェクト参画に対する責任感を高めてもらうことができる	●必要なスキルを明らかにして、作業内容をWBSによって明確に分割して各々の作業の分担と責任を明確にする ●作業間の責任は誰の責任作業かを明確に定義する必要がある ●スキルが不足している場合は、外部研修を含む適切な教育を行うか、または協力会社から社員代替要員を確保する

付録 1.自己評価シート

No.	知識エリア	チェック項目	評価基準	
S9	コミュニケーション	顧客、開発チーム間、協力会社との会議体が決められ実行されているか？	●ステークホルダーと協議し、必要な会議体が設定され、実行されていること	
S10	タイム	進捗状況を定期的に確認できるようになっているか？	●定期的に進捗管理が行われ、プロジェクト・マネージャやプロジェクト責任者、顧客が状況を把握できる仕組みが設定されていること	
S11	人的資源	必要なスキルを持った人材は揃っているか？	●必要な業務・知識領域と要員のスキルの比較を行ない、不足している業務・知識領域について、コンサルタントや専門家のアサインを考慮していること	
S12	タイム	要員の手配（量的）計画はできているか？	●工程の変わり目など、体制の増員が必要なタイミングに間に合うように要員手配が計画されていること ●要員の手配計画について、関係各所にあらかじめ説明をしておくこと	
S13	スコープ	顧客の要件とシステムの機能、性能が一致しているかどうかを確認しているか？	●顧客の要件と機能の対応表またはそれに相当するドキュメントがあること	
S14	スコープ	仕様として設定された要求の実現可能性に問題がないことを確認したか？	●実現可能性を新技術面、性能面から検証していること ●検証結果について、有識者によるレビューがされていること	

	マネジメントにおけるヒント	対策案
	●それぞれの会議体がどのようなことを議論し、決定する場であるのかを明確にしておく必要がある ●会議体の設定では、会議開催の頻度や参加者、目的などを事前に計画し、ステークホルダーに周知しておく必要がある。加えて、議事録を取って承認してもらうなどの手続きについても明確にしておく必要がある ●計画だけではなく、実質的にその会議体が稼働していることが重要である ●公式な場では言えないこともあるので、非公式なコミュニケーションパスにも配慮する。ただし、決定事項は会議体の中で確認する必要がある	●顧客、協力会社など、プロジェクトの状況を見て必要な会議体を設定し、定例化する
	●進捗の状況を把握するための項目を明確にし、協力会社を含めた進捗報告確認のための会議体を設定する ●進捗の状況は、プロジェクト・マネージャだけではなく、プロジェクト責任者、顧客にも報告する必要がある ●進捗状況に加えて、遅延理由、遅延対策まで検討する	●進捗会議を定例的に設定する ●進捗会議で報告すべき管理項目を明確にする ●管理項目は作業工程によって変わる (例)管理項目例 ①計画の変更数　②確定した要件定義の個数 ③実施レビュー数と種類　④日程の予実績の差異 ⑤タスク数(定義数、監視数、軽減数) ⑥課題管理表の解決した課題数、未解決課題数
	●要員のアサインには時間がかかることが常であり、場合によってはアサインできない場合もある。その場合、トレーニングなどによって要員を育てる時間が必要になるので、事前の要員計画の中で必要なスキルについても十分計画しておくことが大切である ●システム基盤に問題が発生した場合はスペシャリストのアサインが有効である ●特殊な業務に関する知識が不足している場合は、外部の専門会社などに依頼することを考慮しておく必要がある ●パッケージ利用の場合、フィット＆ギャップ分析ができる要員や、該当アプリケーションの経験者をアサインしておく必要がある	●最適なスペシャリストを投入する計画を立てる ●必要な人材確保が遅れている場合、最初のうちはそれほど高度な業務知識や専門知識がなくても問題ない作業を行い、作業が進む中で必要な人材を確保する
	●要員を出してもらうことについては、事前に関係会社に確認をしておくべきである。要員を出してもらうつもりだった関係会社から、突然協力できない旨の連絡がくる場合も想定しておくべきである ●担当者個人ごとに作業開始時期、終了時期を明確にして作業をアサインするようにする ●前工程における仕様変更などで必要な人員の増減が必要になった場合は、工程の終了を待たずに速やかに手配する必要がある	●要員遷移俯瞰図の作成を通じて、山積みではなく要員遷移分析を行い、WBSごとに、必要なスキルを持った要員の過不足を明らかにする ●前工程の終了前に、プロジェクト・マネージャがどれだけ次工程の要員手配に注力するかが重要である ●要員の手配は、早めに、過大気味に行う ●要員不足に陥ったときには、急激な破綻を起こさず、軟着陸できるようにするために、スケジュールの延長、または作業量の低減の余地を検討する
	●要件定義書と設計仕様書との対応は、仕様管理だけでなく、テスト計画にも利用できる ●顧客における常識、業界での常識などは、明文化されていない可能性があり、注意が必要である	●要件定義書がない場合には、早急に作成する(最低限、要件項目を明確化する) ●要件定義書がある場合は、現状と要件定義書を比較して特に乖離の大きい部分があればその部分の状況報告をし、顧客側と認識を合わせておく必要がある ●要件定義書には記述されていない要件がある場合、早めに洗い出しをし、クリティカルな要件になるようであれば実装を機能面・費用面から検討する ●要件定義書の作成段階で頻繁に修正があった機能については、最終的な要件について基本設計・仕様書との擦り合わせを行うなど、顧客との認識を十分に合わせておく必要がある ●要件定義書と設計仕様の対応表を作成する
	●計画または設計時に実現可能性そのものについて検討を行っているかは重要である。さらに、内容について有識者などの第三者によるレビューを行うべきである ●実現可能と判断するには、例えば、コスト、機能、納期に対してリスクを認識し、その対応策が見えている必要がある ●実現可能性について疑問視される部分については、顧客側によく確認すること。思い違いにより、ベンダー側が仕様を重く捉えている場合もある	●実現可能性を検討していない場合は早い段階で検討する ●実現可能性が見えていないものがあるかもしれないが、 それらも含めて全体を俯瞰できるよう、要求された項目の実現可能性の一覧を作成する ●実現可能性が低いと思われる機能の場合は、代替機能での実現可能性や機能削除などについて顧客とともに検討する

付録 1.自己評価シート

No.	知識エリア	チェック項目	評価基準	
S15	統合	納入物件および必要な作業成果物が明確となっているか？	●顧客との間でどういう記述方法で、どういうドキュメントを納品するかが明確になっていること	
S16	統合	すべての作業項目がスケジュール化されているか？	●WBSが書き出されており、その内容について十分検討されていること ●すべてのWBSがスケジュールに落とされていること	
S17	タイム	マスタースケジュールが作成され、ステークホルダー間で合意されているか？	●マスタースケジュールが作成されていること ●マスタースケジュールについて、ステークホルダー（顧客、関係会社、プロジェクト責任者など）の承認がされていること	
S18	タイム	スケジュールにおけるクリティカルパス（そのパス上の作業が少しでも遅れると、プロジェクト全体のスケジュールが遅れてしまうパス）は明確か？	●プロジェクトの根幹にかかわる機能や成果物に関する計画はどのパスであるかを認識し、管理していること	
S19	リスク	リスク分析により、リスク項目を明確にし、対応策まで検討しているか？	●リスク管理表などを作成して管理すること ●個々のリスクについて、リスクの対応策とその実行タイミングが明記されていること	
S20	統合	構成管理のルールが明確になっているか？	●計画書、設計書なども含めた、成果物の構成管理の手順やルールが文書化され、メンバーに周知徹底されていること ●プログラムのソースコードや仕様書、テストデータ管理、システムの設定ファイルなどの構成管理方法について計画されていること	

	マネジメントにおけるヒント	対策案
	●顧客との間で成果物の質と量が明確でないとその後の作業の割り振りや規模の見積もりなどで大きな判断ミスをする可能性がある ●事前に成果物サンプルを提示することで、納入物件に関する双方の共通認識を確立することが望ましい ●各種計画書や仕様書、運用テストの項目一覧など、具体的な成果物のサンプルを示すことで、どのような作業が必要になってくるかを事前に検討したり、要員のアサインを考えたりしやすくする	●ドキュメントによる成果物の場合は、目次、類似する成果物サンプルを用いて記述項目と記述レベルについて、顧客と合意し、議事録として残す
	●必要な作業項目を洗い出すために、たとえば各社で標準化しているWBSの雛形を利用したり、過去の他のプロジェクトを参考にするなどの工夫をすること ●システム構築のための設計書作成やプログラム開発作業だけではなく、準備作業、顧客との調整事項など、想定される作業をなるべく網羅的に洗い出すこと ●WBSを基にしてスケジュール中の各作業の前後関係の整合が取れていること	●標準WBSに基づいてテーラリングする。標準WBSがない場合であっても、過去の成功プロジェクトなどを参考にして、WBSを作成する。「このWBSで必要な作業がすべて洗い出されているか」という視点を常に持つこと ●各作業の前後関係の整合が取れていない場合、前作業の終了条件、後作業の開始条件を明確にして、各作業の前後関係が整合するように修正する ●ドミナント・アイテム関連作業のスケジュールは、計画段階から重点的にマネジメントする。例えば、ミッションクリティカル性の高い機能要件などもドミナント・アイテムである
	●マススタースケジュールの各工程での作業内容について、十分に検討していることが大切である。特に、納期を基準にして、作業工程の比率だけで、後ろからスケジュールを引いている場合、それぞれの工程がその与えられた期間で完遂可能かを十分に検証しなければならない ●マススタースケジュールの中で、すでに分かっているマイルストーンについて記載しておくこと ●マススタースケジュールの各工程について、事前準備の内容と時間なども考慮してスケジュールに記述しておくことが望ましい	●納期、総合テストの開始時期、他システムとの接続がある場合は接続テストの実施日など、明確なマイルストーンに基づいてマススタースケジュールを作成する ●マススタースケジュールの妥当性については顧客との合意だけではなく、プロジェクトの担当者、開発委託先のメンバー、有識者などに確認してもらうこと
	●「粛々とやっていくだけ」と考えてプロジェクトを進めていると、問題に気づかないことがある。計画している作業タスクの優先順位を検討して常に配慮することで、問題が発生したときにどの作業を優先させるかを冷静に判断することができる ●クリティカルパスを固定的なものと捉えないこと。状況の変化によってクリティカルになる可能性のあるパスも考慮すること ●複数のアウトプットが集まるタスク、複数のタスクのインプットにつながっているタスクはクリティカルパスの候補の一つになる	●キーパーソンを集めてクリティカルパスの抽出作業を行う。そのうえで、クリティカルパスに関連する作業は特別体制を敷く ●作業工数の見積もりが確定できず、ある程度進まないと見通しが立たない場合には、作業が少し進んだ段階で再度工数を見積もり、クリティカルパスを明確にする
	●リスクの洗い出しには、ステークホルダーが集まって討議することが有効である ●リスク洗出しの際に効果的な視点としては、次のものがある ・成果の量の増減　・成果の質の優劣 ・実施タイミングの是非　・スキルの充足度 ・セキュリティ　・変化対応の柔軟性の有無 ●プロジェクトの状況によってリスクは変化するため、リスク管理表は定期的に見直す必要がある	●チームメンバーを集めてリスクの洗い出しを行い、洗い出したリスクが発生する確率と発生した場合の影響を判定する
	●各種成果物の構成管理は、プロジェクトの初期段階でルールを決めて運用することが大切である。あとで整理しようと思ってもなかなか手を付けることが難しい ●構成管理をきちんと行うことは、成果物および活動の品質を上げることに寄与する ●構成管理ツールを新規に導入する場合には、環境設定や試験運用などの計画を明確にしておく必要がある。また、運用ルールを明確にし、入力するデータの意味などを詳細に定義しておく必要がある	●構成管理ルールがある場合には、ルールが周知徹底されているか確認する ●構成管理ルールがない場合、以下を考慮してルールを作成する ①構成管理のためのツールの導入（費用の確保、環境の整備、運用手順の明確化が必要） ②作業工程によって管理ルールが変わる（使用するツール、管理対象物、管理頻度） ●構成管理ルールが徹底されているかを確認する手段を検討する（ヘルプデスク、進捗会議でのチェックなど）

付録 1.自己評価シート

No.	知識エリア	チェック項目	評価基準	
S21	スコープ	要件定義と仕様変更要望との関連付けが明確になっているか？	●顧客側のニーズ（要求）と要件定義書の項目の関係性について、顧客側、ベンダー側双方のキーパーソンに確認を取ること	
S22	スコープ	仕様変更は一覧などにより管理され、変更履歴を残すことになっているか？	●仕様変更管理表などで管理することが明文化されていること	
S23	課題管理	課題管理表などで課題が管理されているか？	●課題（解決しなければならない既出の問題）の発生日時や課題の内容、重要度、状況、対策予定日が分かるような管理表やそれに類する資料によって課題を随時管理していること	
S24	スコープ	他のステークホルダー（顧客、関係会社など）からの提供物のスケジュールが守られているか？	●提供物のスケジュール状況を継続的にウオッチし、守られなかった場合の影響度に応じて、事前に対策を立てていること	
S25	コスト	複数視点での見積もりを実施するなどして、見積もりの精度を向上する工夫をしているか？	●複数の見積もり手法を使ったチェックや経験十分な複数人による見積もりチェックによって、妥当性を確認していること	
S26	コスト	予実績管理は実行されているか？	●収支管理計画を事前に定義していること ●週次/月次の実績把握の仕組みを確立していること	

	マネジメントにおけるヒント	対策案
	●要件定義と仕様変更の対応付けを管理することによって、顧客要件との合致性を確認する ●システムは要件定義書を基にした各種設計書によって作られている。したがって、この要件定義内容と顧客からの仕様変更要望との対応付け確認は、システムでの実装内容を把握しているキーパーソンも巻き込んで確認する必要がある ●要件定義書と、顧客側からの仕様変更の要求に論理的な矛盾があると、修正したあとのプログラムではシステムトラブルを起こすことにつながる可能性がある	●要件定義と仕様変更との関連付けを明確にするために、要件定義と仕様を対応表にして、相互に影響する要件定義変更や仕様変更を管理する
	以下の項目が明確になっているかを確認する。 ●仕様変更管理表の保管場所 ●仕様変更管理表の記入者・保管責任者 ●仕様書の変更履歴と仕様変更管理表の対応確認方法	●要件定義との関連付けが可能な仕様変更管理表（一覧）を作成する
	●課題管理表の内容が、形式的でないことが必要である ・記載されている内容は本当に課題か？ ・課題の責任者が明確か？ ・ 解決までの経過が記述されているか？（目標日変更も含めて） ・ 解決目標日と完了日に矛盾はないか？ ・目標日を過ぎている未解決項目はないか？ ・ 特定の課題に偏っていないか？ ●課題に対して顧客側、ベンダー側双方に「解決する」意志を持って取り組んでいるかを常にウオッチすること。「これは顧客側の課題だから」と放置しないこと	●顧客側、開発側のキーパーソンを集めて課題を抽出し、一覧表にまとめ、管理する。課題については、管理するだけではなく、ステークホルダー間で共有し、管理が形骸化しないようにする必要がある。具体的には、 ・議事録に課題をまとめて明記し、次回の会議で必ずフォローする ・ホワイトボードに書き出す、ポストイットを貼り付けるなど、課題を全員で共有するようにする
	●他のステークホルダーの役割と提出物のスケジュールを別途マネジメントすることにより、役割分担があいまいになっていることを原因とするプロジェクトの遅延を防止できる	●他のステークホルダーからの提供物がないことに影響を受けない作業から着手するが、作業の進捗状況とステークホルダーとの調整を並行して行い、作業が滞留しないようにする ●ステークホルダーへ情報提供の督促を行うときは、先方の担当者だけでなく、先方の上位上司にも一緒に依頼することが効果的である
	●複数の見積もりをすることによって、単一の見積もりでは気付かなかった重要な要件、実現可能性、考慮すべきリスクなどに気が付くことがある	●見積もり方法が単一の場合や経験に基づくものである場合、現実の規模や工数が想定からかけ離れている可能性がある。 当初予想していた規模と現状の規模を見比べて、大きく離れているようであれば、プロジェクトの今後の計画を見直す必要がある ●スムーズに調整するための要件は次の通り ・顧客との間で費用や実現機能についての調整を随時できる関係が築かれている ・計画工数と実績工数を常に監視し、差異が生じた場合にはその原因を分析する ・ 最終的な工数（費用）が計画値を上回りそうな場合には、その原因を明らかにして顧客と交渉する
	●期間（工程の開始/終了など）、要員の山積み、およびWBSについて予実績管理を行う ●予算を順調に消化していても、仕事が進捗しているわけではないということを分ったうえで、マネジメントすること	●予実績情報を収集・整理して、今後の計画の見直しを実施する ●すべての作業の予実績情報の収集が困難な場合は、クリティカルパスについての予実績だけでも集めて評価する ●過去に多数の類似プロジェクトの実績がある場合は、上流工程でのコストの振れや工程の遅れは抑制しやすい。ただし、「過去のプロジェクトと類似だ」という判断そのものが誤っている場合があり、その場合は後工程で工数が数倍に膨れあがることもある。類似プロジェクトかどうかは慎重に判断しなければならない

付録 1.自己評価シート

No.	知識エリア	チェック項目	評価基準	
S27	品質	品質計画があるか？	●品質計画（右の欄参照）があり、レビューされていること	
S28	統合	全体テスト方針と役割分担の検討は十分か？	●全体テスト方針（何をどのテストで誰が確認するか）とテストの役割分担を記述したドキュメントを作成し、関係者とレビューしていること	
S29	品質	前工程の作業が完了し、その品質を確認したうえで、プロジェクト内の承認を得ているか？	●前工程の結果について品質を確かめていること ●前工程に積み残された課題がある場合は、解決のメドを付けたうえで次工程に引き継いでいること ●条件付きの合格の場合や改善点を指摘された場合は、課題管理の対象としていること	
S30	調達	協力会社への発注の根拠が明確となっているか？	●業務知識、開発スキル、過去の設計・開発実績、担当者のキャリアなどの観点から、なぜその協力会社に委託することにしたのかを明確にしておくこと	
S31	調達	協力会社の作業と成果物を定期的に監視する計画がされているか？	●協力会社の作業と成果物を確認する計画になっていること（例えば定期的な進捗確認会議、工程ごとの成果物の判定会議など） ●成果物の確認は、形式的なチェックだけではなく、現物の中身も含めてきちんと状況を確認するように計画しておくこと ●契約の内容をもとに確認項目を洗い出してあること	
S32	技術	方式要件（負荷・性能・信頼性・安全性・保守/運用・移行）と方式設計結果は妥当か？	●方式要件と方式設計結果に関するレビューを実施していること ●方式設計レビューのレビュー者が十分なスキルや知識を持っていることを確認すること	

	マネジメントにおけるヒント	対策案
	●たとえば次のような点について、計画とレビューがされていることが望ましい ・レビューの種類 ・レビューの日程 ・レビュー不良率（件／ページなど） ・下流工程における品質基準（不良摘出目標など） ●そのプロジェクトで特に重要な品質は何かを検討しておくこと（信頼性なのか性能なのかなど） ●起こってはいけない障害が何かを考えて、それをクリアするためにどのような品質について向上させる必要があるかを検討しておくこと ●品質は様々な要因が総合的にからむので、レビューを通じて、問題点を早期に把握して是正する	●品質計画が作成されていない場合は、すぐに作成する ●品質レビューの検討結果に基づき、対応策を策定して、実施する
	●詳細なテスト方針はともかくとして、どのようなテストを行うかを決めておくことが重要である ●方針と役割分担を明確に決めておかないと、場当たり的なテストになり混乱するおそれがある ●上流工程でテスト工程をある程度想定しておくことで、顧客側の体制や作業場所などの準備作業、テストデータの準備など、協力も得られやすくなる。特に、全国展開するようなシステムの場合は、各事業所ごとにテストをするなど、集中的なテストができないなどの理由で、テストそのものがクリティカルパスになることもある	●単体テスト、結合テスト、運用テストのそれぞれのスケジュールと品質目標が未計画の場合は、過去事例などを基に整備し、計画する ●テスト体制も明確にする。特に、結合テスト以降は優秀な人材（各チームのキーパーソンや優秀なエンジニア）を投入し、遅れない体制を計画しておくことが重要である
	●上流工程の品質の悪さによって、下流工程でより大きな悪影響を及ぼすことがある ●前工程での積み残し課題などがあればそれらの位置付けを明確にし、どのような対応を行うかを計画に入れる	●品質に問題がある可能性がある。品質状況を確認して、問題がある場合は品質向上策を実施する
	●実稼働する担当者のスキルまで確認しておくことが望ましい ●協力会社評価票を準備しておくことが望ましい ●発注の根拠を明確にすることは、その協力会社を使ううえでのメリット/デメリットを明確化することになるので、リスクの洗い出しという観点でも重要な検討事項になる ●業務知識、開発スキル、過去の設計・開発実績、担当者のキャリアなどの観点から、なぜその協力会社に委託することにしたのかを明確にしておくこと	●発注の根拠を明確にして、リスク対策に反映させる
	●契約時に作業進捗・成果物のチェックやレビューのやり方を詰めておく ●確認時にはその報告や成果物の内容チェックまできちんと確認していることが重要。形式チェックだけでは成果物の出来を確認することはできない	●協力会社に協力してもらい、作業報告とともに成果物（特に仕様書などのドキュメント）を最新の状態にするように管理する。 特にソースやシステム環境については構成管理チームを作って徹底的に管理するようにする
	●顧客に責任を持って方式要件を決めてもらい、それに基づいて設計する。ただし、顧客だけでは方式要件を決められないこともあるので、ベンダー側も方式要件の検討を支援する必要がある。 ●有識者（実装経験がある者や、その製品・技術の中身に詳しい者）に要件と設計結果の妥当性をレビューしてもらう ●方式設計において不安項目がある場合は、プロジェクトの初期段階でプロトタイピングを行うなどの施策を講じること	●技術や製品をよく知っている技術者だけではなく、システム構築・導入の経験者や保守フェーズの担当経験者なども入れて方式検討をするべきである ●新規のWebサービスなどで入力件数と出力予定帳票や更新データ件数などが想定できない場合は、上限を設定し、それを超えたときは、オーバーフローメッセージを出すなどの、トラブル対策について検討する計画をしておくこと ●システム障害時の対応想定として、回復の優先順位付けを行う必要がある

No.	知識エリア	チェック項目	評価基準	
S33	基本動作	基本となる動作（しつけ・作法）が徹底されているか？	●「打ち合わせを行うときは議事録をとること」、「設計内容はドキュメント化すること」 など、基本的なしつけ・作法について全メンバーが問題なく認識していること ●セキュリティに対する行動規範が明文化され、徹底されていること	
S34	基本動作	部下の状況を把握しているか？	●公式、非公式のコミュニケーションを部下と行っていること	
S35	モチベーション	プロジェクト計画に対してメンバーの気持ちがしらけていないか？	●プロジェクトの日程計画や実現可能性などに関してメンバーが不信感を持っていないこと ●特にプロジェクト・メンバーや協力会社に対して、プロジェクト計画上の問題点がないか、意見を求めていること	

	マネジメントにおけるヒント	対策案
	●チーム全体が「多少手抜きをしても大丈夫」という雰囲気にならないように、決められたことはきちんと守ることを徹底して指示する ● コミュニケーションを良くすることと、「なあなあで済ます」ことは別であり、緊張感を持って仕事に取り組むことが大切である ●先輩が後輩の良いかがみにならなければならないという雰囲気を作ること	●基本動作についての教育的な指導を普段から心がけるようにする ●すべての基本動作徹底が難しい場合は、プロジェクト状況により重点的に徹底する事項を絞る ●例えば、 ・仕様がふらつき傾向→ドキュメント化の徹底 ・進捗遅延傾向→進捗報告の数値化と見込み提出の徹底
	●部下から何の報告もない場合は、報告を上げづらい問題を抱えているときもあれば、 話しづらい雰囲気などコミュニケーション上の問題があることがある ● 定例進捗会議などの会議ではないところででも、コミュニケーションを随時行っていることが大切である ● 「何か問題を抱えてはいないか？」と声かけをするなど目に見える行動をすること。心配していることを頭で思っているだけでは相手に伝わらない	●直接部下に状況をヒアリングしたり、部下とかかわりのある第三者にヒアリングして部下の状況を多面的に把握する ●権限委譲と丸投げを誤解しないようにする。部下、協力会社に作業を任せるとき、責任まで投げず、共有するようにする
	●プロジェクト責任者よりも、実作業を担うことになるプロジェクト・メンバーや協力会社のほうが、作業工程上の問題や負荷の問題などが見えていることもある。実現可能性について積極的に意見を集めるべきである ●スケジュール、コストのベースラインは、プロジェクト・メンバーが努力すれば、守ることができる水準である ●メンバーの表情や言動を見ることも大切である。作業の質が落ちていると感じたり、粗暴になったりしている場合は、モチベーションが下がっていないか、会話をするべきである	●納得していないメンバーに対して、どの部分が納得できないのかをヒアリングし、問題点を明らかにしたうえで問題認識を全メンバーで共有する ● 「しらけ」を抱えたままの場合は、チーム全体として著しくモチベーションが下がってしまうリスクがある ●クリティカルパス上の作業には比較的「しらけ」の少ないメンバーを配置する ●作業配分、リソース配置を工夫し、直近のマイルストーンをクリアし続けることで、実現可能であることを納得させていく

付録 2.ヒアリングシート

No.	知識エリア	チェック項目	個別のヒアリング要領	
H1	顧客	RFPなどに顧客からの要件として、何を実現したいかが分かりやすく記述されているか？ また、プロジェクト・メンバーがそれを把握できているか？	プロジェクトの位置付けや目的を的確に把握して、共有できることが成功の第一歩 ●顧客企業が作成したRFPの場合、プロジェクトの定義、アウトプットの定義などが明確に記載されていること ●コンサルティング会社などが作成したRFPの場合、顧客がその内容を十分理解していること	
H2	顧客	顧客がオーナーシップを持って対応できているか？	顧客自身のプロジェクトにおけるコミットメントが明確でなければ、何をすべきか判断できない ●ユーザー部門のコンセンサスを得るための活動がなされているか ●ベンダー側としては、提案や意思決定を促すための努力が行われているか ●決定事項が変更できなくなる工程上のポイントを常に明確にしながらプロジェクトを進めることにより、前提が覆ることへの対応が図られているか	
H3	顧客	顧客のキーパーソンを押さえているか？	責任ある人の意思あるいは判断がなければプロジェクトは進まない ●責任部門のキーパーソンが必ずしも決定権を握っているとは限らないので、キーパーソン間の影響関係を把握しておく必要がある ●問題が発生した場合は、連絡先、協議者を個人名で管理できているか	
H4	統合	プロジェクトの全体を俯瞰できているか？	全体像を把握することで、各部分の対応の意味も明確になる ●対象システムの構造のほか、関連するシステムには何があるのかが把握できていること ●主要なステークホルダーとして、 システムと業務との関連性を把握できている人、 システムを利用する人、 財布を握っている人、 システムを作る人、 を把握できていること。関係者の名前だけでなく、各キーパーソンを明確にして、一番効果のある働きかけを行える相手を把握することが重要 ●大日程表の中で、どの部分がクリティカルパスとして重要かが指摘できていること	
H5	統合	プロジェクトのドミナント・アイテムを把握できているか？	ドミナント・アイテムとは、プロジェクトの成否に大きな影響を及ぼす重点項目。それを早期に見つけ出せることが成功のポイント ●俯瞰図に基づいてどのようにドミナント・アイテムを見つけ出したか（その理由など）が明らかになっている ●体制上の各キーパーソンが何をドミナント・アイテムと捉えているかを自分のこれまでの経験と照らし合わせて把握できているか	
H6	統合	プロジェクトの前提条件を、目的、契約、などの内容から、分かりやすく押さえられているか？ また、前提条件に設定漏れがないか？	●プロジェクトの基本的な前提条件が合意されている、という前提がないところから出発する ●プロジェクト・マネージャがプロジェクトは失敗するリスクが大きいことを意識していることが重要である	
H7	統合	プロジェクトの内容として、どれくらいの範囲（工数）を、いつまでに、どういう体制・コストで進めるのかが、プロジェクト計画書にまとめられてあり、そのレビュー・評価結果に、問題・課題が残されていないか？	いつまでにいくらでやるかはプロジェクトの前提条件 ●該当業務や技術に知見のある有識者でレビューすることが重要である ●規模が測られていないと、テスト量も予測できない。全体として大きなスケジュール遅れになる可能性がある ●リソースの手配（予定）と整合していること	

	評価基準	エビデンス・確認方法	対策案
	以下のような記載事項を的確に把握できているか ●開発範囲　●契約条件 ●教育訓練　●検収方法 ●スケジュール	①RFP	●不足分については、内容の不足分をリストアップし、顧客に調査の申し入れを行う ●現行システムの調査作業については、別途協議して再見積もりが必要であることを顧客と合意し、明文化する
	顧客側のプロジェクト推進力が信頼できるかを確認する ●顧客のプロジェクト推進に関する意思決定力が発揮されていること ●決定事項を覆すことが頻繁に起きていないこと	①ステアリングコミッティ ②体制図	●顧客内のプロジェクト・コンセンサスを確認する場を設定するよう要請する ●顧客タスクを文書化し、実行部隊のキーパーソンに徹底するよう要請する ●開発とは別体制の顧客プロジェクト・マネージャ支援契約を提案する ●上記のような対処をしても懸案事項（顧客決断で進めなくてはならないもの）の判断が期限に遅延する場合は、解決策を検討して顧客と合意を得る必要がある
	キーパーソンであることを識別できているかを、次の観点で確認する ●顧客内のシステム部門とユーザー部門との関係を把握していること ●仕様決定権限、予算、経営層へのコミットはどの部門が握っているを把握していること ●決定権・発言力は誰が持っているのかを理解していること	①ステアリングコミッティ ②体制図	●顧客内の意思決定体制を確認し、意思決定に関連するキーパーソンは進捗会議への出席を義務付ける
	全体把握（全体俯瞰）の対象となる事項として、以下の3種類を確認する ●開発対象システム ●ステークホルダー（開発体制を含めることもある） ●スケジュール	①システム俯瞰図 ②ステークホルダー俯瞰図（体制図を含む） ③スケジュール俯瞰図	●プロジェクトのドミナント・アイテムを把握し、対応するために、俯瞰図を作成する
	システム俯瞰図、ステークホルダー俯瞰図、スケジュール俯瞰図、要員遷移俯瞰図を用いるなどして、ドミナント・アイテムを指摘でき、対策を立てていること	①システム俯瞰図 ②ステークホルダー俯瞰図（体制図を含む） ③スケジュール俯瞰図 ④要員遷移俯瞰図	●プロジェクト計画書において、ドミナント・アイテムに関するQCD計画、実施状況を重点的にマネジメントするような計画とする
	プロジェクトの前提条件が、提案書や契約に矛盾なく構成されており、顧客にそのことを十分説明して納得させていること	①RFP ②提案書 ③契約書	●顧客が十分理解し、納得するまで説明する
	以下の観点で確認を行う ●予算とスケジュールと想定規模は整合性が取れていること ●（事業計画などの）当初の想定とスケジュールに大きな乖離がないこと	①プロジェクト計画書	●定量的評価をした結果、金額と納期に問題がある場合は、費用、機能、納期について、社内や顧客と調整する ●問題があっても実施しなければならない場合は、次善策として限界値を約束することが望ましい ●限界値が分からないときは、プロジェクトから撤退するポイントを決める

付録 2.ヒアリングシート

No.	知識エリア	チェック項目	個別のヒアリング要領	
H8	コミュニケーション	開発チーム間(協力会社間)とのコミュニケーションに問題はないか?	定期会議やファイル共有などの方法を通して、コミュニケーションの促進が図られていること ●会議体が定義され、実施されているか ●情報の共有に関して、形骸化した制限などが存在しないこと	
H9	コミュニケーション	作業内容と終了基準についてのメンバーの認識が曖昧ではないか?	曖昧さがあると作業の進捗が把握できなかったり、把握したと思っていてもメンバーにより認識の内容が異なっていたりして、プロジェクトとしての統一した進捗把握ができない。品質にも問題が出ることがある。プロジェクト・マネージャの認識が重要	
H10	コミュニケーション	顧客とのレビュー議事録などがあり、承認印をもらうなどして合意(承認)を得ているか?	●顧客とのレビュー計画書は作成しているか? ●レビュー実施要領は作成できているか? ●実施要領などの中で確認方法を取り決めているか? ●議事録またはメールでも記録として残しているか?	
H11	コミュニケーション	重要な指摘が発生した場合に、指摘事項が全員に周知徹底されて、再発防止を図っているか?	上流では、一つの指摘、変更が他の部分にも影響する可能性がある ●重要な情報についてはメンバーに漏れなくタイムリーに伝達する方法が確立していること	
H12	コミュニケーション	スケジュール・作業内容について顧客との認識(作業レベルの意識)にズレがないか?	●期限のほか、作業内容、成果物の記述レベルなどについて、具体的な資料を用いて調整しているか	
H13	組織	プロジェクトのステークホルダーとして位置付けられている各組織の責任・権限は明確になっているか?	●部門の責任範囲を明確にすることで、会議体への参加意識を高め、施策のフォローを容易にしているか ●事務局やステアリングコミッティが組織されている場合は、それらが機能しているか	
H14	組織	プロジェクト・マネージャがプロジェクトの規模や性質に見合った経験を持っているか? また、経験不足時には、組織的支援策はあるか?	●管理型PMOではなく、支援型PMOになってるか ●火を噴く前に支援する文化があるか	
H15	モチベーション	各メンバーは自分の範囲について責任を果たすという意識が希薄でないか? 自分が担当する範囲以外について、無関心な対応をしていないか?	プロジェクト・メンバーの発言内容や発言回数、担当する成果物の品質、普段の様子から責任意識を持って作業をしているかどうかを評価していること	
H16	タイム	実施するべき作業が明確に定義されているか?	●作業のリストアップができているか ●リストアップはできているが、その作業で何をするのかが明確にされているか	

	評価基準	エビデンス・確認方法	対策案
	●コミュニケーション計画を立案していること ●情報共有の仕方が明確で、それにより必要な計画書、設計書を各担当者がいつでも見られるようになっていること	①会議がどれだけの頻度で行われているかを議事録から確認 ②協力会社を使っている場合は、協力会社へのヒアリング	●コミュニケーションが円滑ではないと判断した場合、適切な会議体を設置する ●担当者間の相性に問題がある場合、1対1でのコミュニケーションには問題が出やすいので、第三者(プロジェクト・マネージャなど)を交えた会議体を定例化する ●協力会社から、さらに一部の作業を別会社に発注している場合、重要事項が実際の作業者まで正しく伝達されないなど、コミュニケーションが滞りがちになる。このような場合は、階層構造をフラットにし、各社からのキーパーソンを集めた会議体を作る
	例えば、システム化計画はどこまでやったら終了か、明確にしているか ●作成すべきドキュメントの完成を宣言できる条件が分かっていること ●プロジェクト・メンバーに、終了基準について説明がなされていること ●無計画な検討スケジュールにしていないこと	①作業内容の終了基準を確認	認識が曖昧になっている場合は、どのような状態になれば終了宣言してよいかの基準を明確にし、作業者に伝える。そのためには、次のことを行う必要がある ●作業進捗状況を定量的に測定するための方法を考え、定量的な進捗報告を徹底する ●成果物の作成品質レベルを評価し、作業完了を判断できるキーパーソンを明確にする
	顧客とのやり取りは文書化して記録を残していること ●顧客との各種レビューを実施 ●議事録を作成 ●その議事録について合意を得る	①レビュー計画書 ②レビュー実施要領 ②議事録	●レビュー議事録を作っていない場合は、作成する ●報告会議などで議事録のレビューを慣例化する
	レビュー結果、障害情報などの情報共有の仕組み・やり方が決められていること	①議事録 ②周知徹底の記録(メールなど)	●顧客レビューなどによる重要な指摘事項を周知徹底させることを義務付ける
	●スケジュール・作業内容について合意を得ていること	①議事録 ②スケジュール表 ③成果物サンプル(記述フォームなど)	●スケジュールの妥当性についてまずは内部で確認・合意し、その後顧客とともにレビューする ●問題がある場合はスケジュールを見直す
	事前に定義しておくべきこと ●工程別の組織 ●各組織の責任と権限 ●会議体と目的 マネジメント・レベルの会議体も準備し、想定外の問題にも対処可能としておく	①体制図 ②会議体	●体制図の見直しと責任範囲の明確な宣言をプロジェクト計画書などで文書化する ●判断をするが責任を取らない組織構造こそプロジェクトを失敗させる最大の要因である
	プロジェクト・マネージャのスキルと実際のプロジェクト規模や難しさのバランスを確認 ●過去の同等規模プロジェクトの経験とそのプロジェクトの成否	①プロジェクト計画書	プロジェクトの状態を監視する支援組織があり、プロジェクト内部のモニタリング・制御が機能しているかを監視することが重要である ●プロジェクト・マネージャに対して的確なプロジェクト・マネージャ教育を行う ●コーチを付ける　●プロジェクト計画書を有識者でレビューする ●PMOを要請する　●プロジェクト・マネージャの補佐役を検討する
	●目標期日までに所定の作業は完了できていること ●期日までに間に合わない場合は、どうするかを相談していること ●同じチームのメンバーが何をしているか知らないような状況に陥っていないこと	①進捗会議などの議事録 ②報告書 ③担当者へのヒアリング	●各メンバーに、自分自身の作業内容について自ら進捗報告をしてもらう。また、報告を受けた課題に対して対策や具体的な指示を行う ●さらに、非公式のコミュニケーションによりメンバーの真意を探り、一つずつ対策し、意識を向上させる
	●WBSを基にスケジュール表が作成されていることが前提となる ●各タスクの作業規模見積もりがなされて、いつまでに実施すべきタスクなのかが共有できていること ●各タスクの順序だてのつじつまが合っていること	①スケジュール表 ②WBS ③見積もり ④責任分担表(体制表) ⑤議事録	●キーパーソンを集めて、やるべき作業のリストアップを行う ●リストアップされた作業について、作業内容と成果物を明確にし、 作業量の見積もりと作業スケジュールを明確にする ●担当者をアサインする ●新しい領域での最初の開発など、業務ノウハウがない場合は、開発が進むにつれてやらなければならない作業が見えてくるので、新たに明らかになった作業項目を管理対象に加えるとともに、全体を見て、作業分担の偏りがないかなどを確認する

No.	知識エリア	チェック項目	個別のヒアリング要領	
H17	人的資源	各作業に担当が明確に割り当てられているか？（WBSやスケジュール表に担当者の名前が記述されているか？）	●誰がその作業をするのかはっきりさせておらず、誰にやらせるかが、常に先送りになっていないか？ ●WBSの作業には、自社でできるもの、他社との関係で作業するものがある ●失敗プロジェクトはかけもち（兼任）で作業している場合が多い ●責任分担は明確でないといけない ●兼務している担当者のスケジュールに余裕がないといけない	
H18	タイム	担当の作業負荷や作業の重なり具合、空き具合についてチェックしているか？	●管理者が各担当ごとの作業負荷が計画と整合が取れているかチェックしているか ●適切な、進捗指標を使っていること。進捗報告が利用されていること。現物（実物）をみて管理しているか ●終了したことを証明するエビデンスが規定されており、終了の際に実物で確認しているか？ ●進捗数値の計算方法を決めているか	
H19	タイム	工程進捗のつじつまは合っているか？（スケジュールに表した各タスクのアウトプット・インプットは作成順として矛盾がないか？）	●計画変更が発生した場合、追加作業前後の作業工程、スケジュールへの影響を考慮しているか？	
H20	統合	プロジェクトの日程、責任範囲、WBSなどがステークホルダー間で合意されているか？（プロジェクトのWBSと顧客のWBSを比較した上で調整を行っているか？）	●顧客とプロジェクトとの協働作業でやらないといけない ●お互いの作業分担を明確にすることがこの項目の主旨である	
H21	タイム	マススタースケジュールとチーム別スケジュールとの整合性は十分か？	●プロジェクト管理ツールを利用するなどして、整合性を取るようにしているか？ ●複数のベンダーが出したスケジュールの一つに変更あった場合に、全体的なコントロールができ、マススタースケジュールとの整合性をとっているか？ ●それぞれのスケジュールに根拠があり、単純な人月計算でスケジュールを作っていないか？	
H22	タイム	クリティカルパスは明確か？	●クリティカルパス候補を特定できているか？ ●クリティカルパスを認識し作業管理されているか？ ●変動的なクリティカルパスを、日々モニタできているか？ ●ボトルネックとなりそうな人員を識別できているか？ ●作業バランスが取れているか？	
H23	タイム	マススタースケジュールと要員山積みとの整合性は十分か？	●当初の計画との差異を見ているか？ ●作業ピークを計算した山積みになっているか？	
H24	タイム	マイルストーンについて、顧客との間でコンセンサスがきちんと得られているか？	●マイルストーンやイベントなどについては、議事録に残すだけではなく、線表にまとめ、顧客側関係者と開発側関係者で共有して意思疎通を図っているか？	
H25	タイム	本番移行計画が明確か？	●移行計画書が作成されている、もしくは作成する予定であること（移行計画には、人的移行、システム移行、データ移行などがある） ●ファイルの移行はもちろんのこと、顧客データを移行するときには、顧客データの品質に問題があることが多いので、 顧客の保持しているデータの信憑性をよく確認して移行計画を策定することが重要である ●システム導入後の支店展開など、長期にわたっての運用移行については別契約としているか	
H26	品質	本番環境でテストを実施する計画が策定されており、本番環境での確認項目が作成されているか？	●本番環境でのテストが実施できない場合は代替案を検討しているか？	

	評価基準	エビデンス・確認方法	対策案
	●誰が、何を行うかについてプロジェクト・マネージャと各担当者が認識を共有していること	①体制図 ②スケジュール表 ③WBS ④責任分担表	●各タスクの進行状況が明確になってくるにつれて、各担当者の状況に応じて動的に割り当てる方法もある ●いずれの方法でも、特定の要員に負荷が集中している一方で、遊んでいる要員がいる場合は、作業の割り振りを見直したり、多忙要員が抱えている作業を他者にまわす努力をする
	●プロジェクト・マネージャと各担当の双方がチェックしていること	①スケジュール表 （進捗管理） ②WBS ③見積もり ④責任分担表	●進捗会議などで作業負荷状況を定期的にチェックするようにする。 具体的には、開発メンバーを少人数のチームに分割し、チームごとに作業を割り振り、作業分担や作業負荷の調整はチーム内で行う。チーム内で調整できない、影響の大きい項目だけを全体（チーム間）で調整する方法が効果的である
	●各作業項目のスケジュール表（小工程・ガントチャートなど）の前後関係に矛盾がないこと ●作業開始の前提となるものができていないのに作業が開始されてしまうことがないこと	①スケジュール表 ②WBS ③見積もり ④責任分担表	全体線表と各作業項目の進捗具合をチェックし、不具合が発生しているようであれば、作業をいったん停止、見直しをする。 ●作業の効率等を考慮して、着手できるところから着手する ●後戻りが発生しないようにするため、着手しようとする作業を規定する前作業の終了を確認する
	作成したWBSがステークホルダーが納得の上で合意されていること 例えば以下のような観点で確認を行う ●現行調査の確認方法 ●要件定義に関するユーザーとの確認方法など	①プロジェクト計画書 ②責任分担表 ③WBS	●後で「聞いていない」ということにならないように、関係者を集めて日程と責任範囲について合意を得る会議を開催する
	●マスタースケジュール作成→チーム別スケジュール作成→マスタースケジュール改訂のステップを実施して両者の整合性がとれていること	①変更管理表 ②議事録 ③マスタースケジュール ④チーム別スケジュール	●マスタースケジュール作成→チーム別スケジュール作成→マスタースケジュール改訂のステップを実施して両者の整合性をとる。 具体的には、マスタースケジュールを理想のスケジュールとし、チーム別スケジュールは現実の積み上げスケジュールと捉え、作業を進めながらチーム別スケジュールをマスタースケジュールに近付けるよう調整する
	●クリティカルパスに当る作業項目が洗い出せていること	①スケジュール表 ②WBS	●クリティカルパスに遅れがある場合はキーパーソンを集め、早急に問題点の洗い出しを行い、対策を取る。クラッシング、ファーストトラッキングなどの手法がある
	●マスタースケジュール作成時に要員山積みを作成して整合性を確認していること	①マスタースケジュール ②要員山積み表	●要員調整計画を見直す ●ピーク時を想定して要員が足りなければ、開発作業と並行して要員を調達する
	●マイルストーンの内容は議事録に記載して承認してもらうこと	①議事録 ②マイルストーン スケジュール表	●マイルストーン設定内容について顧客から合意を取る
	●顧客との責任分担と日程などの作業期限を明確にすること ●移行計画や移行に関する作業についても顧客との合意を得ておくこと	①プロジェクト計画書 ②移行計画書	●本番移行計画を立て、有識者や顧客も含めたステークホルダー間でレビューを実施する
	●テストの段階で本番環境を利用しなければならない範囲・事項が存在することをあらかじめ洗い出していること（本番系しかないネットワークでのテストなど）	①プロジェクト計画書	●本番環境でのテスト計画が未計画の場合は、計画を立てる（本番環境でのテスト実施は必須。顧客だけのWBSとなることもある） ●本番環境でのテストが実施できない場合は、代替案を検討する ●小規模の改変などの場合は、省略することもできる

付録 2.ヒアリングシート

No.	知識エリア	チェック項目	個別のヒアリング要領	
H27	タイム	プロジェクト線表の進み・遅れをどのように捉えて表現しているか？（例えば、稲妻線の記述） 捉え方の根拠は明確か？	●進捗率の計算方法をあらかじめ決めているか ●進捗率に対応した成果物ができているか	
H28	人的資源	求められる業務知識のキーパーソンを獲得できているか？	●漠然と「業務分野の経験者」と捉えずに、求められる経験・知識レベルを明確に把握する必要がある	
H29	人的資源	求められる技術スキルのキーパーソンを獲得できているか？	●対象プロジェクトのキーとなる技術は何かを理解している（求められる技術スキルはプロジェクトによって異なる） ●該当技術のキーパーソンの経歴・レベルを確認する（質問する人の技術レベルにも依存するので注意。自分の技術レベルが低いと他人を「高い」という傾向がある）	
H30	人的資源	品質保証部門や他のプロジェクトのステークホルダーからの指摘、意見を受け止めて対応（必要ならば是正）を行う責任者が明確か？	●隠ぺいしようとしている部署はたちが悪い ●会社によって品質保証部門の責任の持ち方が異なる。品質の責任を持つのはプロジェクト・マネージャであるが、プロジェクトが大規模でプロジェクト・マネージャが一人でできない場合は専任の品質保証担当者を置く必要がある	
H31	タイム	（H16～H30に関するヒアリングを踏まえて）これまでのスケジュールに関する個別確認を通して、作成しているスケジュールに根拠（考え方のベースとなるもの）があると判断できるか？	●根拠を考える範囲が明確か ●協力会社のスケジュール根拠を判断しているか ●能力とスキルを考慮しているか ●発注元からスケジュールの根拠が得られているか	

	評価基準	エビデンス・確認方法	対策案
	●進捗率の値が感覚的なものではなく、きちんとした根拠に基づくものであること	①スケジュール表 ②WBS	●進捗の評価方法を形式化／文書化し、進捗報告者に徹底させる ●成果物の作成状況や現場の雰囲気なども把握して、進捗率の報告は感覚的なものになっていないかを確認する ●残りの作業項目を明らかにしたい場合は、進捗率に加えて、残作業工数での管理を追加するとよい
	●求められる業務分野の経験者で、キーパーソンとなれる要員を押さえること		●求められる経験・知識レベルを具体的に把握する ●次に、業務知識のキーパーソンを確保できているかを確認する ●社内に人材がいるときは、相談やチェックだけでも参加してもらう交渉をする ●社内に人材がいないときは、 ・協力会社から調達する ・理解力のある人物を起用または採用する ・プロジェクトから降りる（失敗コストと罰金/信用失墜との見合い） ●ただし、業務をよく知っていても引っ込み思案やリーダーシップが欠如していれば、キーパーソンにはなれない
	●プロジェクトにとって新しい技術を採用する場合には、当該領域の有識者を獲得できていること ●パッケージ利用の場合は、フィット&ギャップ分析ができる要員やプロジェクトで使用するソフト（DB・ミドル・ツールなど）の経験者がキーパーソンとなっていること		●求められる技術レベルを具体的に把握する ●次に、技術のキーパーソンを確保できているかを確認する ●社内に人材がいるときは、相談やチェックだけでも参加してもらう交渉をする ●社内に人材がいないときは、 ・協力会社から調達する ・理解力のある人物を起用または採用する ・プロジェクトから降りる（失敗コストと罰金/信用失墜との見合い） ●ただし、技術をよく知っていても引っ込み思案やリーダーシップが欠如していれば、キーパーソンにはなれない
	●課題や問題に対する対応策が計画されていること ●議論による問題解決や課題・障害の解決に向けて、ステークホルダーを説得できていること	①課題管理表	是正処置が行われていない原因を特定し、指摘事項の重要度と影響度を考えて対応策を実施する。 例えば、 ●指摘や意見に対し、是正処置を実施したいが負荷が高すぎて実施できない ●スキル・要員が不足していることが原因 ●顧客と品質レベルとしては了解を得ている ●不適切なところは、後日見直すスケジュールにしているなど
	●作業量と体制との関連付けが明確になっていること	①スケジュール表を書いた根拠	なぜ根拠のないスケジュールを作らざるを得なかったかという背景を探る ●顧客・プライムコンストラクタ・上司からの根拠のないプレッシャー ●プロジェクト・マネージャの知識／経験不足 ●メンバーの言いなり ●プロジェクト・マネージャの意思が弱い 背景を把握した上で、再スケジューリングをする。 ポイントは次の通り。 ●作業要員、個人別作業能力と、集中度（過負荷にならない）を勘案する（予算との見合いで要員増強・期間延長など） ●リスクに耐えられるスケジュールにする。必ず途中でチェック、レビュー、修正期間を入れる。“急がば回れ” ●初めに納期ありきの場合は、リスクとなる可能性が非常に高い（各フェーズごとに、それ以上は短縮できない最低作業期間がある。その期間をクリアするようにする）。この場合には、作業の進行状況に応じて、ステークホルダーと調整しながら、適宜スケジュールを見直す

No.	知識エリア	チェック項目	個別のヒアリング要領	
H32	スコープ	システム化の方向性を定義するような機能範囲と作業範囲の変更について、変更管理手順が明確になっているか？	業務要件、システム機能について、変更管理のベースが存在すること ●納期、コストとの関係を考慮してコントロールしているか ●成り行きでシステムのスコープが変わらないようにしているか ●各ステークホルダーに変更管理手続きを徹底させているか	
H33	スコープ	想定外のスコープ増（機能範囲と作業範囲の増加）が発生しているか？ 発生している場合、それは許容範囲内か？	●スコープ増加時の対応策（機能削減、費用増、スコープ変更）を顧客とあらかじめ合意しておく ●スコープ増の場合は、テストの増大やクリティカルな機能に対する影響度を確認する	
H34	スコープ	前工程担当者から引継ぐ時、引継ぎ資料の読込み、説明会の実施などによって、充分な理解を得ているか？ （引継ぎに関するQAシートに未回答や意味不明な個所がないこと）	●引き継ぎ資料が不足なくあることや、その内容についてもきちんと理解しているか ●前工程のアウトプットを十分に咀嚼しているか ●理解度が不足している場合は、引き継ぎ資料のレビューや上流工程担当者からの再説明を受けているか ●コンサル会社が入っている場合、成果物の引き継ぎがきちんとされているか	
H35	統合	保守・運用計画および作業手順・ルールを明確にし、レビューを実施しているか？ あるいはその計画があるか？	●プロジェクトのクロージングで問題になるケースがある ●統合テストより前の段階で確定しているか	
H36	リスク	リスクの内容について、その内容が明確に記述されているか？	●リスク管理表により、想定されるリスクの洗い出しが行われているか ●想定されるリスクの洗出しに関して、プロジェクトに関連する有識者からの意見が取り入れられているか ●たとえば顧客との打ち合わせに出席している者でないと意味が分からないような記述になっていないか	
H37	リスク	リスクを減少させるために、リスクの分析と影響度のプライオリティ付け、対策について十分検討しているか？	●想定リスクの重大度と発生可能性の評価結果により、対応策（人を増やす、納期を調整する、機能を落とすなど）の優先順を定めているか ●リスクの発生時の対応には、コスト予備、スケジュール予備の確保、要員の代替や追加投入が必要になることをプロジェクト・マネージャが認識できているか ●次工程におけるリスク評価のタイミングが設定されているか	
H38	リスク	顧客などのステークホルダーとリスクを共有しているか？	●プロジェクト・リスクの洗い出しや、対応策検討について顧客と対策協議を行っているか ●顧客との協議がなされていない場合、リスクが問題となって顕在化したときに、責任分担で問題になることがある	
H39	リスク	セキュリティ管理方式に対する考慮がなされているか？ 同時に、個人情報保護法の観点からレビューを行っているか？	●ユーザーが既に有している管理方式を踏襲することで、設計、運用に際して違和感を与えずにシステム構築を進められる。ただし、セキュリティ管理方式の強化が求められるプロジェクトにおいては、新しいルールの設定と徹底が必要 ●個人情報保護法に沿ってシステムが実現できているか、既存部分との関連を含めて有識者のレビューを受けておく必要がある	

	評価基準	エビデンス・確認方法	対策案
	システム対応のスコープそのものが記述されているドキュメントを特定できること。 そのようなドキュメントが存在する上で、下記のような視点で管理レベルを確認する ●窓口、取りまとめ者が明確になっていること ●解決期限が明確になっていること ●実際の実施に関して問題がないこと ●変更管理手順が明確になっていること	①変更管理表 ②変更管理ルール	●変更管理ルールがない場合は、変更管理についての手続き手順を明確にする ●変更管理の実施に問題があるかを関係者にヒアリングし、 問題がある場合は、どのような問題点があるかを聞き出し、改善する
	●スコープ増を定量的に押さえていること ●許容範囲かどうかをコスト、スケジュールの観点から評価していること	①変更管理表 ②議事録	●許容範囲でない場合、どの程度の増があるのかを明確にした上で、顧客との打ち合わせを実施する。ただし、大幅な機能範囲や作業範囲の変更がステークホルダーの合意の下に行われており、変更に伴うコスト増、納期遅れについても顧客と合意できている場合を除く
	●何が引き継ぎ資料なのか定義されていること ●検討が十分でないところが明確にされていること	①引き継ぎ資料	●理解度に不足がある場合は、コンサルティング部門に再度支援を依頼する。起こりがちなことなので、あらかじめ引き継ぎ資料を理解するための工数、期間を想定したスケジュールを立てることが望ましい
	●体制作り、保守・運用計画に問題がないかをチェックすること ●あるいは、保守・運用計画を作成する作業が予定されていること	①保守・運用計画書 ②レビュー記録票	●保守・運用に関する体制の計画を立て、関係者と合意しておく ●作業手順・ルールがない場合は、過去の事例などを参考に整備する
	リスク管理表に以下の内容が記載されていること ●想定するリスク ●リスク検知のための確認項目 ●該当のリスクの影響度 ●リスク発生防止のための対応策 ●リスク発生時の対応策とその実施タイミング	①リスク管理表	●リスク管理表上、曖昧な記述や誤解を生むような記述がないかをチェックする。そのような記述があれば、記入者を特定して、真意をヒアリングし、分かりやすく明確な記述にする ●その後、再度レビューを実施し、リスク内容をステークホルダー間で共有する
	●個々のリスクについて、その発生頻度と影響度を分析し、対策を検討していること	①リスク管理表	●リスク管理を行っていない場合は、リスク管理表を作成して、回避策について検討を実施する。手馴れた仕事で、リスクがなければ除外する
	●リスク管理表をステークホルダー間でレビューしていること	①議事録 ②リスク管理表	●リスクに対する意識を持たせるために、ステークホルダーとの打ち合わせでリスク管理表をレビューする ●プロジェクトの状況悪化を検知できるようにして、リスクを検知した場合は、逐次対処する
	以下のようなシステムの管理項目がルール化されていること ●ユーザーアカウント管理の実現方法 ●データ保存媒体管理 ●開発・テストにおけるシステム利用基準 ●テストデータ活用基準 個人情報保護法と関連性があるかを見極めたうえで、セキュリティ管理レベルについてレビューがなされていること	①リスク管理表	●顧客が有する情報セキュリティ標準を入手しておくことが前提 ●設計対象システムが顧客側のセキュリティ標準の考え方と整合しているかをレビューを通じて検証する ●システム提供者が守るべきセキュリティ管理関連のルールを明確に文面化したうえで、すべての担当者がそのルールを理解、順守するように徹底する

付録 2.ヒアリングシート

No.	知識エリア	チェック項目	個別のヒアリング要領	
H40	統合	構成管理のルールや手順を策定する予定になっているか？	●例えば上流工程においては、検討範囲をまたがる共通検討事項について構成管理のルールに基づいて、情報共有ができていないと、統一の仕様が守れないことが想定される(ドキュメントの先祖返りが起きないようにする) ●策定方針があらかじめ設定されることにより、次工程移行の管理負荷を軽減することができる	
H41	統合	上流工程における構成管理の対象(ドキュメント類)が明確になっているか？ また、プロジェクト全体にわたって、構成管理の対象(ソースプログラムなど)が明確になっているか？	●マルチベンダー実施体制の場合に1社が遅れると、順当に進まない場合がある ●ベンダーが虚偽の報告をしている場合、構成管理そのものがきちんとされていない ●不良が出たときにどうするかなど構成管理のルールが必要である ・バグ票が回ったときに追跡できるようになっているか ・問題点の追跡管理ができるようになっているか ・課題管理は優先順位を付けて行うようになっているか	
H42	スコープ	仕様変更対応の(契約上の)取り扱いについて顧客と合意し、別精算などの対応を行っているか？	特に仕様変更をどう取り扱うかについて契約がきちんとしてないと、トラブルが発生しやすい ●たとえば、承認ルート、精算方法、リスク・インパクトなどを考慮して顧客と共通認識とする ●スコープについて契約で規定されている必要がある(2次請けも同様)	
H43	課題管理	課題管理を実施しており、それらの課題について、顧客との合意が取れているか？	●課題管理について、顧客との合意ができているか ●顧客側で課題と認識しているのに、開発サイドでは認識していない場合は、将来、大きな問題に発展することがある ●課題として共通認識になっていても、重要度や緊急度の認識のずれも問題になる場合がある	
H44	課題管理	管理されている課題の対策予定日が過ぎたものに対して、対処の強化を実施しているか？	●対策予定日が大幅に過ぎているものはないか。対策予定日が大幅に過ぎているものは、以下のようなプロジェクトの推進に関して問題を含んでいる可能性がある ・担当者がアサインされていない ・担当者の認識がない ・課題が大きすぎて解決策が具体的なところまで落とし込めないまま放置されている ●課題管理の対象にリスクを混入させないこと ●課題の平均解決日数などを活用すると、課題の大きさや意志決定の速さを評価できる	
H45	課題管理	複数の課題解決策に対し、明確な判断基準(メリット／デメリット)のもとに優先順位をつけたうえで、解決策を選択しているか？	●経験や勘だけに頼った判断をしていないか？ ●経験や勘だけに頼った判断は大きな判断ミスにつながる。判断には根拠や基準が必要である ●判断の材料となる選択肢についても、十分に検討し、よりよい判断であるかを常に意識しているかをチェックする	
H46	コスト	システム化の規模見積もりを行っているか？ その際の見積もり根拠を記録してあるか？	●実施した見積もり結果の位置付けがはっきりしていること(予算取り、ベンダー調達の参考情報、顧客との契約) ●見積もり方法、見積もり基準(生産性の想定値)など、算定方法を明確に残していること ●開発規模、体制、生産性、開発期間が整合するか第三者チェックを受けていること ●再見積もりの際に、前回との見積金額差異が、説明できるようになっていること	

	評価基準	エビデンス・確認方法	対策案
	●構成管理のルールや手順について、策定方針が明確になっていること ●第1版がどの時期に承認を得たかが分かるルールになっていること	①ルール・手順書	●構成管理の対象物を明確にし、管理ルールや手順を明文化する
	成果物の管理がきちんとなされていることを確認する。例えば、以下のような観点で確認する ●ドキュメントの構成管理方法が明確になっていること ●プログラムソースの管理方法が明確になっていること ●プログラム設計レベルの修正は、どの時点の仕様書まで遡って更新されるかが決まっていること	①構成管理計画書	●構成対象が不明確、あるいは管理されていない場合は、必要な構成対象をリストアップし、それらの管理開始日を決める
	●顧客と文書で合意が取れていること	①プロジェクト計画書 ②体制図 ③責任分担表	●契約上の取り扱いについて合意が取れていない場合、開発を一度止めて、早急に合意を取る ●合意が取れるまでの経緯も、必ず文書に記録しておく
	●顧客と課題管理表のレビューを実施し、その内容および改訂事項について合意を得ていること	①課題管理表	●是正処置を実施していない原因を突き止める ●特に課題内容が大きすぎて、誰が着手するべきか判断ができない場合は、キーパーソンを集めて対応策を検討していく必要がある ●大きな課題か、なかなか合意が取れない場合は、エスカレーションして対策を打つ
	●課題管理表に完了予定日を記入していること ●期限が過ぎた課題に対して、優先順位を上げて対応していること	①課題管理表	●判断の根拠を明確にしたうえで判断する。判断の根拠、優先順位を課題管理表に記録を残す ●しかし、明確な判断ができない場合には、スキルを持ったメンバーやプロジェクト外の技術者などとレビューを実施して、解決策を選択する
	●解決策を複数考え、論理的な比較を行ってより適切な解決策を選択するようにしていること ●解決策が単なる思い込みになっていないこと	①課題管理表	●判断の根拠を明確にしたうえで判断する。判断の根拠、優先順位を課題管理表に記録を残す。 ●しかし、明確な判断ができない場合には、スキルを持ったメンバーやプロジェクト外の技術者などとレビューを実施して、解決策を選択する
	プロジェクトのシステム化対象範囲について規模見積もりを行っていること。また、その見積もり根拠が明確になっていること。見積もり根拠の例としては、現行ステップ数、過去の類似プロジェクト、機能数やファイル数からの係数見積もり、複数有識者の意見集約（デルファイ法）などがある	①システム化規模見積もり	●工程の進行に伴い、より精度の高い見積もりを行っていくことを計画に盛り込む ●見積もり結果について、前回との差異がどこから発生しているかを、顧客に説明して合意（納得）を得る

No.	知識エリア	チェック項目	個別のヒアリング要領	
H47	コスト	顧客が提示するシステム化費用において、当初の計画からの「振れ」を調整しうる部分は明確になっているか？（仕様変更の増分に対して、削減できる部分の候補を持つことなど）	●当初の計画との差異を、費用、工数、機能のそれぞれについて管理しているか ●工数山積みがコスト制約に見合っているか ●基本契約は結ばれている場合には、調整の余地が乏しい	
H48	コスト	プロジェクトの進展に伴って発生するコストが予測できているか？	●上流（企画）工程におけるスケジュール延長について、どの程度のコスト増になるか認識できている ●以降の工程についても、プロジェクトへの参画人数や業務内容に関連してどのタイミングでどの程度のコストが消費されるのか把握できていること	
H49	品質	要件定義に記された内容から、テスト段階での検証項目が導き出せるようになっているか？	●要件定義を記述する際に、統合テストの実施を先読みして、検証項目として漬し込みがしやすいようにしているか	
H50	品質	性能要件を満たしていることを確認するためのテスト計画が明確か？	●本番環境でないと実施できない性能要件テストの場合以外について、性能要件テストを計画しているか ●性能要件テストをすることで、課題を認識したり、 課題を解決するため対処策を検討し、対策を打つことができる	
H51	品質	障害対策時の運用面でのテストの計画が明確か？	●実運用面での操作・作業ができるかどうかを事前に確認しているか	
H52	統合	作業環境（作業場所、支援システム、ツール）についての見通し、およびテスト環境利用計画が検討されているか？	●計画段階でこのようなことを検討して見通しをつけ、それに基づいて対策を実施することが重要である ●環境準備の段取りがWBSとなっている ●パッケージ利用やオープンソースソフトウェアの組み合わせなどでは、設計段階で見えてなかったものが統合段階で突然出てくることがあるので注意が必要である ●方式面で初物（未経験技術や今までにないソフトウェアの組合せ）やパッケージ利用をする場合は、注意する。特に、パッケージはカタログスペックと実際のスペックが異なる場合がある	
H53	コミュニケーション	顧客への報告は随時行っているか？	●検討状況についての中間報告などを行い、随時情報を提供する必要がある	
H54	技術	システム実現の基本事項として、方式設計が行われているか？	●方式設計の位置付けを認識できていること ●方式設計の内容について、実務経験のある人が設計を行うか、またはレビューワーとして参加してもらう必要がある	

	評価基準	エビデンス・確認方法	対策案
	コスト制約に見合った機能削減や請負金額の増額、スコープ変更などの対応実施策を顧客と決めておくこと ●費用はどこまで増額が可能なのか ●削減可能な機能はどこまでか	①プロジェクト計画書 ②リスク管理表	コスト制約に見合った機能削減を実施することを顧客と決める。 スムーズに調整するための要件は次の通り。 ●顧客との間で費用や実現機能についての調整を随時できる関係が築かれている ●ある程度作業が進み、計画と実績の差異が生じた段階で、機能ごとの工数、費用を算出し、機能の削減について顧客と調整を行う
	順調に進行した場合の発生コストや不調に陥った場合に追加発生するコストの規模について把握できていること		人員の調達を中心に、プロジェクトの予実績管理について、プロジェクト・マネージャから見えるようにする
	●要件定義の内容から、テストシナリオの設定のめどが立つこと ●各要件からシステムの利用ケースになぞらえて、テストケースに分解できるようになっていること	①全体テスト計画	●テスト計画に問題点がある場合は、問題点およびリスクの洗い出しを行い、解決方法およびリスク回避方法を検討する。特に、テスト作業のクリティカルパスを見極め、全体に影響のある部分を重点的に計画する
	●運用テストや実際の運用に入ってから性能問題が発生しないように、事前に性能要件テストが計画されていること	①プロジェクト計画書	●性能テストの計画が未計画の場合は、計画を立てる ●性能目標値が明確でない場合は、顧客との打ち合わせを実施して明確にし、合意する ●顧客との打ち合わせでも性能目標が明確に決まらない場合は、SIベンダー側で妥当と思われる目標を設定して、システム性能を設計する
	●業務面での影響を考慮のうえ、障害発生時の復旧猶予時間が検討されていること ●復旧猶予時間内にシステム復旧のシナリオが明確になっていること ●代替運用のテスト計画が立てられていること	①テスト計画書 ②障害運用書	●障害対策、運用テストの計画が未計画の場合は、計画を立てる ●顧客打合せで障害・運用についてなかなか決まらずテスト計画が作成できない場合は、顧客と重要性を共有することによって、決める必要がある
	●プロジェクト計画書に明文化されてること ●開発の実施方針として、計画書に以下の観点で記述されていること ・開発環境の構築 ・開発場所の確保 ●テストの実施方針として、計画書に以下の観点で記述されていること ・テスト環境の構築 ・テストデータの準備 ・テストの実施タイミングなど	①プロジェクト計画書 ②テスト環境利用計画 ③WBS	●今後の作業計画と照らし合わせて、作業場所、支援システム、ツールなどの環境面で不都合がないかを確認する ●テスト環境利用計画を立てる
	●顧客との会議体を決めて、実際に開催していること ●緊急報告体制を構築していること		●課題やリスク、プロジェクト状況などを報告するための会議体を設定する ●緊急時は、即時連絡する
	方式要件と方式設計についてレビューが行われていること ●ハードウェア：非機能要件がインフラ（設備）設計に反映していること ●アプリケーション：外部接続のプロトコルや共通ログの出力といった要件が共通設計に反映されていること ●ユーザーインターフェース：エラーメッセージやファンクションキーなどが統一仕様としてルール化されていること		●方式設計の実施タイミングを全体計画の中で確保する

No.	知識エリア	チェック項目	個別のヒアリング要領	
H55	技術	システム的な移行切替方式（他社からの移行などを含む）については、移行対象物、切替ツールなど十分に検討しているか？	●可能な限り単純な移行切替方式を検討しているか	
H56	技術	レスポンスの確保や機器障害時の対策など信頼性に関する方式設計については十分に実施しているか？	●アプリケーションの機能設計ばかりに注目しすぎず、非機能要件（レスポンス確保や無応答状態の回避など）について、 ソフトウェア／ハードウェアの両方の観点から検討されているかどうか	
H57	技術	トラフィック量とトラフィックパターンについての想定の検討は十分に行われているか？ どのような対応策を採るか明確になっているか？	●顧客と話し、ピーク時のトラフィック量について対策を検討しているか ●限界量以上のデータが入ってきたときにサービスの停止を考慮しているか ●顧客との間で責任範囲を明確にしているか	
H58	基本動作	議事録のほか、各種ドキュメントが正しい日本語で書けているか？	●文面化した内容をもって、ステークホルダー間にそごを生まないような記述ができているか ●正しい日本語での記述は、プロジェクトの状況を把握する上での前提事項となる。特に議事録を読む相手のことを考えて、正しい日本語で記述できているかどうかが重要である	
H59	基本動作	プロジェクト内で使われる用語の意味について、顧客およびプロジェクト・メンバーの認識のずれがないようにしているか？	使われている用語の意味の取り違えにより設計ミス、作業ミスを起こさない対策を採っているか	
H60	基本動作	プロジェクトのメンバー内に話しづらい雰囲気があって報告が上がってこないという事象はないか？	●課題管理や進捗管理など、プロジェクト・マネジメント上の問題が報告されていないことはないか	
H61	基本動作	非公式の場で上下間のコミュニケーションが図られているか？	会議の場以外で情報共有を行うことで、通常では表面化しない問題に対応できることがある。特に、メンバーのマインドの問題には有効	
H62	モチベーション	プロジェクト・メンバーに、このプロジェクトでの個人成長目標を持たせているか？	●プロジェクトを通して、個人の成長が期待できるということが、話し合われていること。各メンバーが成長するための目標を持っているかどうかは、モチベーションに大きく影響する	
H63	モチベーション	プロジェクトの問題点や作業環境について、メンバー内に不満、要望がでてきていないか？	より良い作業環境を望む前向きな意見か、プロジェクトに対する不信感かを区別する。メンバーの要望がプロジェクトの運営に反映できていることが重要	
H64	モチベーション	社員の労務状況（労務時間、作業環境）は悪くないか？	●遅刻が多い、さっさと帰る、仕事がないのに残っているなどの問題がある場合に、原因分析を行い、適切な対応策（増員や人材配置の見直しなど）を実施しているか	

	評価基準	エビデンス・確認方法	対策案
	●有識者のレビューを受けて、移行方式の実現性などをチェックすること	①プロジェクト計画書 ②リスク管理表 ③移行計画書	●移行方式に問題がある場合は、リスクを洗い出したうえで顧客と打ち合わせを行う ●不測の事態に陥ったときのリカバリ方式を検討して、移行失敗に対するリスクを低減する策を練る 特に、移行する対象が明確になっているかが最も重要（お得意様先の住所や金額など顧客経営に直結するような項目のデータ品質を問われるような場合には全面的な顧客責任を前提とする）
	●例えば、大量アクセスがあった場合のスレッド枯渇や障害発生時のクラスタによる二重化などの施策は考慮されていること		●方式検討がされていない場合は、方式に問題がないかを再度チェックする ●システム基盤技術の知識や運用に長けた外部ベンダーやコンサルタントに協力を要請するなどの対策を打つ
	●クライアント数やアプリケーションのトラフィック量、ミドルウェアが利用するトラフィック量など、システム全体としての要求負荷や、ピークを想定しておくこと	①要件定義書	システムで処理できるトラフィックを定義する際、トラフィックに応じたシステムの振舞いを明確に定義しておく必要がある。 トラフィックの定義方法は、次の通り。 ●顧客から現状の運用に関する情報を集めて、想定されるトラフィック量とパターンを導き出す ●全くの新規プロジェクトの場合で運用実績がない場合はトラフィック量とパターンを見極めるのは困難なので、測定のためのプロトタイプを用意するか、本番移行前に、仮リリースしてトラフィックを測定する ●それも困難である場合は、前提条件を明確にしてトラフィック量・パターンを決めておき、それについて顧客と合意しておく
	●各種ドキュメントの内容を確認していること	①議事録	●議事録や障害票の日本語に問題がある場合は、教育的指導を実施する。また、悪い事例の修正例を公開・情報共有して、注意を促す
	●用語集などを作成し、関係者間で共有できていること	①用語集	●用語集一覧を作って顧客およびプロジェクト・メンバーとレビューを実施する ●顧客と長い付き合いがある場合は、必ずしも用語集が必要でないが、顧客側用語を使うかベンダー側用語を使うか決めておくことは必要である
	●進捗、課題管理の定例などでのメンバーの発言量が少なくないこと ●メンバーが他のメンバーの発言に対して、活発に意見を出し合っていること		●プロジェクト内の雰囲気作りを推進する ●上下の人間関係に問題がある場合には、メンバー変更など体制の問題として取り上げる
	●会議以外の場でも報告、相談が行われており、共有できていること		●プロジェクト内の雰囲気作りを推進する
	●プロジェクト・メンバーと個人成長目標について話し合いができていること	①要員育成目標	●プロジェクトとして各メンバーの育成目標を定めて、各メンバーとその内容について合意する
	●プロジェクト・メンバーとプロジェクトの問題点や作業環境についてなどコミュニケーションができていること ●メンバーの不満、要望に、適切に対処していること	①メール ②ヒアリング	●プロジェクト・マネージャを補佐するスタッフを付けて管理を強化しつつ、不信感を表している人それぞれとコミュニケーションをとる
	●労務時間などを基に判断するとともに、現場に残っているかどうかを目視で確認すること	①勤務管理表 ②職場見学	●労務状況をよくするために、増員、人材配置の見直し、作業環境の改善などを実施する ●労務状況の改善が難しい場合、コミュニケーションを頻繁に取りメンバとの一体感を作るようにする

No.	知識エリア	チェック項目	個別のヒアリング要領	
H65	モチベーション	人員の離脱が多くないか？（心身における体調不良、一身上の都合など）	●離脱が多い場合、その原因がプロジェクトへの不信感・危機感ではないか	
H66	組織	協力会社を使う場合、開発の下請けが多階層になって、責任範囲が曖昧になっていないか？ 多階層による問題（責任範囲、コミュニケーション）はないか？	多階層になっている場合、責任の所在がどこにあるかが不明確になりがち。多階層構造にするか、フラット構造にするかを評価の上、組織を作っていることが重要	
H67	調達	協力会社を使う場合、作業の内製、外部委託の判定を行い、調達作業を決定しているか？	●外部調達とした理由、範囲、期間が明確になっているか ●協力会社の力量を十分つかんでいるか ●協力会社の概況を把握しているか ●力量評価に基づいて、状況把握のための定例会議を開催しているか（営業を同席させる、頻度を増やすなど） ●発注する際に約束事項を明確にしているか（例えば、不良が予定以上出た場合に品質向上策を実施する、瑕疵責任をとらせるなど）	
H68	調達	協力会社を使う場合、協力会社の進捗・品質に関する中間フォロー実施計画が明確になっているか？	●請負契約は成果物さえ納入すればよいと考えていないか ●請負業者の品質方針・品質管理体制などの具体的な品質管理手順について確認を行ったか ●中間時点のチェックを約束・実施しているか	
H69	人的資源	他部署と連携して作業を行う場合、作業範囲と役割分担、作業を推進する責任者がステークホルダー間で明確か？	●自社と他部署とのミッションがはっきりしているか	
H70	顧客	業務改革を含む場合、計画時に実現可能性について検討したか？	業務改革の内容が実現可能性上問題がないことを検討し、その内容について顧客との合意が取れていること ●業務、技術の有識者の参加を伴うウォークスルーによる検証がなされているか ●成約事項に見落としがないかをチェックした資料を残しているか ●性能面（バッチ所要時間、オンライン・レスポンスなど）、技術面での不整合に見落としがないか、検証していること	
H71	スコープ	他社の開発範囲とのシステム上の接続がある場合、責任分担は明確か？	●仕様決定に関する調整方法についてのルールなどが明確か ●仕様取りまとめの責任者が明確になっているか ●テスト方針の設定の段取りが明確になっているか	

	評価基準	エビデンス・確認方法	対策案
	●休みや離職の傾向を確認していること	①体制表	●離脱原因がプロジェクトへの不信感・危機感である場合は、プロジェクトの区切り（終わり）の時期について明言してあげることで改善したり、プロジェクト推進に関してプロジェクト一体感や仲間意識を醸成することで、離脱を抑えることができる場合がある ●また、プロジェクト・マネージャを補佐するスタッフを付けて体制を強化する方法も有効である ●さらに、定常的に付き合う協力会社などを作り、離脱が食い止められない場合の要員補充を容易にできるようにすると同時に、キーパーソンの状況を重点的に確認し離脱されないようにしておく
	●役割分担、責任範囲を明確にできていること	①体制図 ②役割分担表 ③ヒアリング	●各社のキーパーソンを集めた会議体を設置する。その中で各社の責任範囲を確認・見直しを実施し、定期的にコミュニケーションを行う
	●実施責任者（プロジェクト・マネージャ、プロジェクト・リーダーなど）および有識者を交え、協力会社に依頼する作業が適切な規模、難易度であることを検討していること ●協力会社の要員数、技術・スキルのレベル、これまでの実績、業務ノウハウの保有について判断し、適切な協力会社を選択すること ●作業の外部調達の決定に際して、社内稟議を行っていること ●外部流出してはいけないシステム化範囲を委託していないこと	①協力会社の実績資料 ②協力会社との契約書（受託明細） ③協力会社マネージャとの面談結果	●想定されるリスク対応策を立てる ●作業の内外判定をしないで、従来からの協力会社に発注する場合は、今回のSOWに関する能力を探り、サポートするか、マネジメントをしっかりすれば対応できる場合は、その点に留意してマネジメントする
	●進捗会議の定例化や品質基準の明確化についての計画がされており、ステークホルダー間で合意されていること	①進捗管理表 ②委託契約書（提出すべき中間成果物などが記述されていることがある）	●協力会社へのフォローに関して問題がなければよいが、フォローを行っていない状態である場合は、フォローの計画を立てて、実施する ●協力会社との信頼関係が築けている場合は、必要ない
	●例えば進捗報告や会議体への参画責任者が明確になっていること	①進捗報告書	●取りまとめは誰が責任を持って行うのか、ステークホルダーのそれぞれの位置を明確にする ●上流の検討段階においては、作業を分担するというよりも、一体型の共同検討の形で進める方が効率的な場合もある
	●業務改革におけるシステムの位置付けと期待される役割が明確になっていることを確認する ●理想論に偏っていて、実際の論理が破綻しているようなことがないか確認する	①プロジェクト計画書 ②リスク管理表	●未検討の場合、業務改革についての実現可能性の検討を行って、問題がありそうであればリスクとして管理する ●現場レベルの調整が必要になる可能性があるので、顧客との認識あわせをする
	他社接続に関する責任分担表とテスト・移行スケジュールを関係会社と合意しておくこと。 例えば、以下のような観点で確認を行う ●責任分解点を定義する方法、資料は特定できているか ●責任分解点が変更になる要素があるか ●ある場合は、どの時点で確定できるのかが明確になっているか	①プロジェクト計画書 ②移行計画書	●責任分担と接続に関するスケジュールを明確にすることから始め、次に、顧客、他社、自社の3者での打ち合わせを実施する

No.	知識エリア	チェック項目	個別のヒアリング要領	
H72	顧客	現行機能保証が必要な場合、現行ドキュメントの整備状況は十分であるか？	●「現行機能を保証すること」という要求の場合、範囲が明確になっているか。さらに、現行機能に関する仕様書があるか ●現行機能の保証を読み取る範囲が顧客と確認できていることが重要	
H73	調達	パッケージを活用する場合、製品・ベンダーを含めた品質を判定しているか？	●評価基準に基づく、選定会議が実施されているか ●選定会議において、採点評価がされているか ●システムの中でのパッケージの利用想定と、パッケージ製品の提供機能が整合していないのに（フィット＆ギャップ分析の結果、ギャップが大きいのに）使おうとしていないかを確認しているか	
H74	技術	新技術/未経験技術がある場合、その技術に対する対応策は十分か？	●新技術／未経験技術の導入効果に対して、過度の期待や役に立たないという思い込みなど、きわめて主観的な評価をしていないこと ●評価者自体が新技術を理解していない可能性はないか ●社内に評価・支援組織があるか ●プロジェクト・メンバーにとって、新しいシステム基盤や技術要素がある場合には、早めのプロトタイプの導入と検証の工数が確保されていること	

	評価基準	エビデンス・確認方法	対策案
	●顧客から文書の形式で仕様書を受領し、その中身について十分に確認していること ●保証しなければならない現行機能について、要求仕様を記述して、顧客と合意していること	①プロジェクト計画書 ②リスク管理表	●現行の調査作業については別途協議として、再見積もりを行う必要があることを顧客と合意し、明文化する ●現行ドキュメントがなく、ソースファイルだけという場合は、現行機能のリバースエンジニアリングが必要で、そのための工数が必要になる
	●使用するパッケージ製品の導入事例の数や、導入先の企業特性などを顧客と比較して、パッケージの品質を適切に判定して選定していること ●パッケージ・ベンダーのサポート品質レベルと、顧客の要求する品質レベルが整合しているかを考慮して選定していること ●技術的観点から評価できる有識者を交えたレビューが実施されていること	①パッケージ活用リスクチェックリスト	●ソリューションとしての的確性、品質、推進体制に関するリスク・マネジメントを事前に行い、リスク対応策を立てる ●実現すべき機能の内、パッケージによる割合が低く、パッケージに問題があっても、新規開発により納期内に完了させることができる場合は、リスクを受け入れてもよい
	●机上評価、社内外の実績を評価し、開発着手前に実機評価をするようスケジュール化していること	①プロジェクト計画書	●新技術や未経験技術に対して、開発チーム側として対応に問題があるようであれば、有識者をアサインするなどの対策を打つ ●新技術・未経験技術であることをスケジュールに反映させておく

付録 3.プロジェクトにおける問題事象と対策の事例集

1 顧客側担当者が異動になり約束事が反故に

顧客側の担当課長レベルと費用および作業内容について内々に話をつけ仕事をしたが、顧客側の職制が変更になった（前任者は退職）。後任者は前任者からこのことについて全く引き継ぎがなかったため今までの話がご破算になった

事例における見切りの内容

決められた社内ルールを無視して顧客との契約をせずに開発に着手

本来の見切りの考え方

- 費用と作業内容については最低限、文書で取り交わしておく。作業を実施するに至った経緯についても記録しておき、たとえ後付けでも作業の妥当性を顧客側の新担当者に説得できるようにしておく必要がある
- その非公式な約束事が反故になるケースを想定し、そのリカバリ方法が確保できていて、かつリカバリ・コストへの対策を考えておく

捉えるべき兆候

- 顧客側の人事異動情報

対処例

- 契約する前であれば、機能を削減し、予算判明以降に発生する赤字を減額する
- 契約内容を確認し、プロジェクト単体が最小限の損害で契約を満たす方法を検討する
- システム化対象範囲を分割して複数年度の契約にするなど、戦略的に採算が取れるような交渉を進める

2 本当のユーザーを見誤ってしまう

情報システム部門からの要請によるシステム構築であり、情報システム部門と契約し、開発を進めていた。そのため、顧客内部のユーザー部門と情報システム部門に対立があったようだが、発注者である情報システム部門主導でのシステム仕様作成を進めていた。いざ実装してユーザー部門での利用が始まると、操作性や仕様に関しての問題が相次ぎ、仕様変更対応に追われ、予定通りの本番稼働が出来なくなった

事例における見切りの内容

ユーザー部門からの旧システムに対する問題提示や新システムへの要求などが上がっていたが、あまり議題に取り上げることなく情報システム部門主管で仕様決めを行った。本来はユーザー部門側からも仕様検討者がいるべきではあるが、顧客企業の情報システム部門ということで、特に仕様に関しては問題ないだろうと思っていた

本来の見切りの考え方

- よい考え方とは言えないが、受注側のプロジェクト・マネージャーは、発注側との検討の末に定義した仕様について仕様凍結であることを合意できていれば、その後の仕様変更に対して別途契約を結ぶなどの対処を取ることができる。そのために仕様凍結の議事録を残すなどの処置は、プロジェクト遂行上、最低限必要である
- 情報システム部門だけで業務をきちんとサポートできるシステム仕様を決定できるかどうかは、その担当者の業務経験量や知識量に依るところが大きい。システム部門の担当者が業務部門から信頼が厚いことが望ましい
- 一般的にエンドユーザーであるユーザー部門から「これでは業務が回らない」と新システムに対する文句が出ると情報システム部門は対応せざるを得ないことが多い。したがって上流工程においては、仕様の決定に際し、情報システム部門だけではなくユーザー部門側の承認ももらうようにする必要がある

捉えるべき兆候

- システム機能の話ばかりで業務レベルの検討を実施しようとしない
- 先方の体制にユーザー部門が入っていない
- 情報システム部門がユーザー部門との打ち合わせの場を持ちたがらない
- ユーザー部門への質問を情報システム部門で制限をかけようとする

対処例

- はっきりと仕様変更だと認めてもらった上で、仕様変更の手続きに従い、計画の修正／変更を行う。まだ上流工程であれば手戻り量は少ないことが多いが、仕様を見直すとスコープが想定以上に広がることもあるので、場当たり的な対処はなるべく避ける
- 仕様凍結が公式に認められていない場合、更に要件定義／基本設計から請けており、発注側が「仕様検討時の問題である」と一歩も譲らない場合など、要望を随時聞いて対応するのではなく、ユーザー部門に対する要求仕様の洗い出しから整理しなおす。場当たり的な対応では、仕様矛盾が埋め込まれ、その後の工程で問題が顕在化することが多い

3 新サービスと非合理的な納期

金融機関で創立○○年の記念日に新サービスを開始することが至上課題だった。新サービスについてはマスコミへの発表もされていたため納期順守が絶対であった。ところが、新サービスということで実現方式も二転三転し、手戻りが発生。カットオーバー前に突貫作業でとりあえず動くレベルの物を納入。しかし、稼働後は突貫作業の影響で品質の問題が多発した

事例における見切りの内容

目新しい新サービスということで、実現方式がなかなか決まらなかった。そのため、非機能要件やセキュリティポリシーについてもなかなか決まらず、方式設計が曖昧なままであった。納期まで時間がないこともあって、開発に進めそうなところから着手していかざるを得なかった

本来の見切りの考え方

- プロジェクト進行上何か問題が発生した場合は、QCDのいずれかを犠牲にするしかない。つまりQ(品質)、C(コスト)、D(納期)のどれを優先させるかを明確に意識するべきである。この事例ではD(納期)を優先課題としているため、Dを犠牲にはできない。したがって、Q(品質)かC(コスト)を犠牲にする必要がある
- 上流工程の段階で不調が現れてきた場合は、最悪のシナリオを考えて、実現しなければならない最低限の機能と品質について早めに合意を取ること
- 中・下流工程であれば、予算を追加し、要員を増やしたりすることは可能であるが、上流工程では要員の増員は功を奏しないことが多い。したがって、Q(品質)をどこまで妥協できるかを顧客側と合意が取れていれば、プロジェクト側も対策を打ちやすい

捉えるべき兆候

- マスコミ発表など対外的なセレモニーを優先させるキーパーソン
- 顧客任せの仕様決め

対処例

- 顧客側でなかなか仕様が決まらない場合は、納期を基点にして、バックフォワードで線表を引き、これ以上の遅れはプロジェクトの破たんを招きかねないという危機感をもってもらうようにする。そのためには類似する他プロジェクトを参考にするなどして、なるべく詳細なWBSを準備する
- 顧客側の上層部に対しては、マスコミ発表のキャンセルについて手続きを検討しておいてもらう(これにより現実的な危機感を持ってもらうことができる)
- 方式がある程度限定可能な場合はそれを満たすマルチスキルな人材を確保し、次工程に向けた要員確保と準備を進める

4 仕切る気がないとりまとめ役

大きな開発物件で、複数ベンダーが一緒に開発を受注した。開発範囲の中でどこまでをどのベンダーが構築するかという分担がなかなか決まらずに時間だけが経過。責任分担が不明確な機能の重複や、機能自体が無い物が発生してしまった

事例における見切りの内容

会社間での調整が不十分であることが分かっていたが、開発が進むうちに開発分担が決まっていくだろうと思い、とりあえずプロジェクトは開始された

本来の見切りの考え方

- 開発の境界線が曖昧なプロジェクトは破綻する危険が高い。特に複数ベンダー間での調整の場合、なるべくうまみのあるところで境界を引きたがるため、とりまとめ役(この事例では発注者)の仕切りの技量が問われる
- とりまとめ役がどの程度プロジェクトを仕切る気があるのか、どれだけのオーナーシップをもってプロジェクトに参画しているかを見定めて、十分な仕切り能力があるのであれば、プロジェクトの進行とともに開発範囲も明確になる可能性があると見てよい
- 発注側が仕切り役であるという役割分担を認識していないようであれば、プロジェクト体制や役割分担について顧客と合意を得た上で、想定する境界線を自ら宣言してプロジェクトを進める必要がある

捉えるべき兆候

- 主張しない発注者(ベンダー任せ)
- 会議体を設定するだけの発注者
- 検討項目や課題項目だけがリストアップされて解決担当社(者)が明確にならない課題管理

対処例

- プロジェクト全体の俯瞰図を作成し、境界と見逃しの有無を確認する
- 自分の作業範囲の確定と、自分に関係する作業範囲を誰が実施するのかを明確にする
- 請け負ったシステム開発の範囲について想定範囲を明示し、その開発を進めるために必要な情報の提供を積極的に打診する

5 発注側から要件が小出しにされ、マネジメントできなくなる

発注側にて要件定義を実施しており、その要件定義の結果を受けて、受注側で開発することになっていた。しかし、サービスインのスケジュールまでの時間に余裕がなく、しかも要件について小出しにされ、提示された要件の変更も多発。基本設計の手戻りも多発することになり、結局赤字プロジェクトになってしまった

事例における見切りの内容

要件定義が未完了ではあったが、要件定義が固まったと顧客から言われたところから、基本設計に着手せざるを得なかった

本来の見切りの考え方

・工程を後ろから引いて、要件定義をフィックスしなければならない日が明確になった上で、その日までに顧客から要件について提示があることを作業の前提条件とすることを合意しておく
・重要な機能とそうでない機能があるはずであり、その優先度を意識して要件定義が進められていることが望ましい

捉えるべき兆候

・発注側の要件定義の進捗状況
・キーパーソンが要件定義状況の全体を語れるかどうか（決定済と未検討項目について説明ができるか）

対処例

・まずは発注側で洗い出そうとしている要件の検討範囲や検討項目を教えてもらい、受注側ではある程度の作業ボリュームをおさえておく
・要件未確定によるリスクの洗い出しを実施し、発注側に対し合意してもらう
・要件定義をフィックスする期限を明確にし、顧客に約束して頂く。この際、すべての要件定義を何日までに、という形では具体性がないので、先に洗い出した要件の検討範囲・検討項目に照らし合わせて、各項目ごとに優先度の高い順に期限日を設けてマネジメントを行う

6 体制の事前確保が仇に

契約に向けて顧客側と折衝を続けていたが、金額面がなかなか折り合わなかった。顧客側から要求されている納期が非常に短く、サービスインスケジュールを確保するためにベンダー側で要員を確保した。要員がそろったため作業を進めていたが、いざ契約の段階で開発対象機能を削減されてしまい、それまでに進めていた基本設計作業は契約範囲外ということでなかったことにされ、契約未締結期間の要員コストはベンダー側持ち出しとなり、赤字プロジェクトになってしまった

事例における見切りの内容

プロジェクトを進める上で必要な要員をすぐに準備できるわけではない。契約を締結してから体制作りや要員調達をしていては、短納期プロジェクトは難しい。そのため、契約は未締結であったが、金額交渉に問題はないと判断し、基本設計に着手してしまった

本来の見切りの考え方

・契約未締結であるにもかかわらず作業を実施することは認められていない
・契約未締結期間の持ち出し分についてのコストを事前に見積もっておき、その分のコスト見積もりと実施内容について、契約時に上乗せさせてもらうことを合意してもらっておく
・契約締結の遅れは営業の責任と割り切れるのであれば、営業部門と交渉し、持ち出し分の費用について負担してもらうよう確約を取る。そのためにも、プロジェクトの悪化要因が契約プロセスにあることを営業や顧客に理解してもらっておくこと
・金額面で折り合いがつかない状態であることから、費用削減交渉がされており、当然機能削減や最悪の場合プロジェクト中止の場合もあることを想定しておかなければならない

捉えるべき兆候

・契約期間の遅れとその原因

対処例

・契約の締結を急ぐ
・作業着手依頼をもらう
・依頼単位で作業承認をもらう
・作業時間の記録を残す
・上記によって、仕事を依頼しているという認識を共有する
・金額が折り合わないにもかかわらずサービスインスケジュールは守らねばならない理由があるはず。見積もり前提が崩れるなら、それを補填可能な要求を出すように折衝を進める
・契約締結・金額アップは営業をフォローする
・原価低減活動（オフショア、共通部品化など）を検討しておく

7 営業がプロジェクト・マネージャの承認のないまま概算見積もりで受注契約を結んでしまった

顧客側のシステム投資予算が決まっているようであり、ベンダー側の営業部門はその予算額をある程度把握しているようだった。その予算に収まるよう、概算で予算を見積もって顧客に提示し、契約した。しかし、最終的には見積もりを大きく上回る開発工数を要し、大幅赤字に追い込まれた

事例における見切りの内容

営業が詳細見積もり未完了のまま請負金額を確定させてしまっており、プロジェクト・マネージャによる見積もりを実施しないまま開発を着手してしまった

本来の見切りの考え方

- プロジェクト・マネージャがプロジェクトを請け負うときに、自らが見積もった金額を営業に提示し、営業側見積もり予算とプロジェクト側の見積もり予算との乖離を明示する
- その時点ですでに赤字であったとしても、それをコストのベースラインとし、それ以上の赤字を増やさないという視点でプロジェクト・マネジメントを請け負う。営業側見積もりの甘さによる赤字は、プロジェクト実施側の責任ではない。このことを営業と合意できているのであれば、プロジェクトを進めてもよい

捉えるべき兆候

- 契約確定時の見積もり根拠
- 戦略受注（たとえ赤字になろうとも受注することに重きを置くという受注）かどうか

対処例

- まだ上流工程で問題が顕在化していないのであれば、プロジェクト・マネージャが見積もりをしなおして、早めに契約時の見積金額との差異を理由と共に明らかにする
- たとえ営業を問い詰めても「それを何とかするのがプロジェクト・マネージャだろ」と言われることが多い。可能なコスト低減策を説明し、それでもこれだけの低減が精一杯であるということを説明すること
- 可能な限り生産性を上げつつ、合理的な理由を掲げ少しでも回収する

8 基本設計の承認なしで詳細設計に着手

顧客がビル移転をするとともに古いシステムを廃棄することなっていたため、新ビルでの新システム稼働が絶対条件であり、サービスインスケジュールになんとしてでも間に合わせる必要があった。要件定義工程が大幅に遅れたため基本設計を顧客に提示後、承認を得ないまま、詳細設計まで進行。途中、顧客から基本設計の変更を要請され、詳細設計をやり直すという二度手間になり、遅れたスケジュールを取り戻すため、要員の追加することとなってしまった

事例における見切りの内容

要件定義に時間がかかったが、その分、システムの実装の姿が見えていたつもりだった。顧客側もだいぶ焦っており、基本設計書を顧客に提出したが、もともと承認されるまでに時間のかかる顧客だったので、承認未完了のままで詳細設計に着手した

本来の見切りの考え方

- 詳細設計に顧客の合意がないまま着手できるのは、その着手範囲の仕様変更が発生してもリカバリができるだけの範囲であることである。たとえば基幹系の重要な部分や他システムとの連携が多い部分は、仕様がひっくり返されれば影響範囲が広いため、たとえ工程が遅れても合意を得た上で着手する必要がある
- 仕様の固まり具合を見定めて、システム単位、機能単位、データ単位などのいくつかの視点で、仕様変更に柔軟なシステム設計をする工夫をする必要がある。その見極めを行い、仕様変更をして欲しくないところを説明できるようにしておくこと

捉えるべき兆候

- 基本設計書に顧客側の承認があるか

対処例

- まずは顧客から要望されている基本設計レベルの仕様変更が、すでに進行している詳細設計に対してどの程度の影響なのかを調べる必要がある。基本設計書の合意を得ずに進めている以上、受注側は文句を言える立場ではない

9 評価が終わっていない基本設計で協力会社に着手依頼

基本設計のフェーズが若干遅れてしまい、まだ品質評価（フェーズ終了判定）が終わっていなかったが、詳細設計を協力会社に着手してもらった。その後、下流工程で不適合個所が発見され、原因調査の結果、基本設計不良であるとこが分かり、大幅な手戻り作業が発生してしまった

事例における見切りの内容

詳細設計以降を委託する予定だった委託先とはすでに契約済みだった。基本設計の品質評価（完了判定）が未完了ではあったが、その委託先の着手日が到来してしまったこともあって、その基本設計を提示し、詳細設計に着手してもらった

本来の見切りの考え方

・先出しする仕様の確定度がどの程度かをよく吟味すること
・先出しする仕様に変更があった場合に、どの程度の手戻りが発生するかを把握できており、たとえその手戻りが発生してもリカバリできると考えているのであれば先出し着手でもよい

捉えるべき兆候

・基本設計の出来具合を、直接執筆者にヒアリングし、問題意識を感じているかどうか

対処例

・要件の確定度が高く、基本設計の品質の高い機能から詳細設計に着手。その際、その部分の仕様変更が発生した場合のコスト増については自社がかぶるものという認識と覚悟が必要
・協力会社との契約の見直しが可能であれば見直しを行い、着手日の延期を調整する。それにともなう費用と、仕様変更による費用との比較を行って、よりコストの低い方法を選択するという考え方もあるが、仕様変更は別のリスクを生むため、なるべくならお金を払ってでも、着手を待ってもらうべきである

10 性急な立ち上げでの体制確保のため、要員調達でコスト高に

スケジュールを守るために開始時期をずらせなかった。通常は設計工程は社員比率を高くして実施するが、社員を集めるためには社内調整に時間がかかるため、協力会社の要員で補充することにした。ただ、急を要する手配だったことと、ハイスキルの要員補充をお願いする必要があったため、コストをあまり下げることができなかった

事例における見切りの内容

プロジェクトの立上げに手間取り、設計フェーズでの要員の確保が不十分のまま、設計を開始した

本来の見切りの考え方

・上流工程は人に依存する部分があり、上流工程の途中で人を増やしても効果が出ないことが多い。したがって、社員か社外（協力会社など）の要員かにかかわらず、上流工程の立ち上げ時点からべったり参画できるかどうかが重要

捉えるべき兆候

・欲しい人材（社内）の業務負荷

対処例

・さらに次の工程の要員確保を早めに始め、超過分を補うべく有利な条件で契約を行いマイナス分を吸収

11 業務経験・業務知識の乏しい要員で要件定義工程を実施

要件定義に必要な要員の人数は確保できたので、プロジェクトをスタートした。担当者に業務知識がなく、要件定義は日を追うごとに遅れていき、成果物の品質もきわめて低いものとなってしまった

事例における見切りの内容

業務経験・業務知識があるかどうかを特に確認しないまま、確保した要員ばかりで、要件定義作業に着手した

本来の見切りの考え方

- 業務経験・業務知識のない要員だけでの要件定義では、十分な要件定義の品質を確保することは相当難しい。要員のそれらのスキルを確認していないことがそもそも問題
- たとえ要員の業務経験・業務知識の不足が分かっていたとしても、顧客側の担当者が業務経験・知識について深い理解を持っており、ベンダー側の業務知識不足を理解してもらっている上で仕様検討の打ち合わせをしてもらえる状況であり、かつ、その担当者によるレビューを頻繁に実施してもらえるのであれば、軟着陸させることができる

捉えるべき兆候

- 体制作りの段階での要員のスキル確認
- 要件定義工程の手戻りの量
- 顧客側からの苦情

対処例

- なるべく早期にエキスパートに参画してもらうなど、体制面での強化を急ぐ
- 要件定義のレビュー頻度を高めるなどのレビュー体制強化をする。その際のレビュアーは、顧客側の業務担当者とベンダー側社内のエキスパート
- 要件定義書の標準や作成手引きなどを作成して品質を一定におさめる

12 コーディングの標準化を実施する時間を惜しんだためにかえって非効率に

段取り不足で、標準化を実施する時間的余裕がなかった。単体テストや結合テストで十分な応答速度を得られない処理が多数発覚したうえに、プログラム修正にも必要以上に時間を要することとなった

事例における見切りの内容

コーティング規約を作成しないで、プログラムコーディングを実施した

本来の見切りの考え方

- 開発担当者のスキルとモラルが高い場合は、コーディング規約がたとえ定められていないとしても、高品質で分かりやすいコーディングをすることが多い。その場合は、必ずしもコーディング規約が必要というわけではない
- レビューを徹底して品質を確保することができれば、規約はそれほど重要ではない。ただし、コーディング規約がないことによるコードのバラツキが、レビュー効率を下げることもあるので、開発担当者のスキルのバラツキがある場合には、やはり遅れてでも作成するべきである

捉えるべき兆候

- 開発担当者のスキルが低さ（コーディング規約のなさが生産性の低下につながる）

対処例

- コーディング規約は言語ごとにインターネット上で入手することができる（組込ソフトウェア開発向けのC言語の規約についてはSECより提供されている）。自社でコーディング規約を策定する時間がない場合は、これらのコーディング規約を利用する
- プログラミングの工程にすでに入ってしまっている場合は、各自が最初に作成したプログラムに対し、全員でピアレビューを実施する。2つめに作ったプログラムも全員でピアレビューする。以降はプログラムの品質が一定になる
- 静的チェッカーを利用する

13 法務部同士で合意に至らず契約が破談

発注側と受注側の、それぞれの法務部間で契約書の内容に関する調整が滞っていたが、設計に着手した。ところが、契約書の内容について合意が得られず、破談となってしまい、それまでの設計工数が丸々無駄になった

事例における見切りの内容	本来の見切りの考え方
本来、未契約で作業を実施することはよくないことだと分かってはいたが、ただ単に事務処理が滞っているだけだと思い、設計に着手した	・契約前の作業については、その契約が破談になったときにそれまでの作業コストが自社で持ち出しになることについて認識しておくべきである ・契約前にもかかわらず作業を実施しなければならない場合は、顧客からの作業着手依頼書などの証跡を残した上で、作業を開始すること
捉えるべき兆候	**対処例**
・契約締結の遅れとその原因	・契約締結までは絶対に作業着手をしてはいけない

14 初物プログラム言語での開発で十分な性能が出ない

ユーザー側から、まだ市場に出て日が浅いプログラム言語を使っての開発を指定された。その顧客はそのベンダー側にとっては重要な顧客であったため、どうしても受注したかった。そのプログラム言語での実装を行ったが、十分な性能要件を満たすことができなかった

事例における見切りの内容	本来の見切りの考え方
どうしても受注するという営業判断もあったため、自分たちで経験していない新しいプログラム言語についての見極めができていないまま、プロジェクトをスタートさせた	・新しいプログラム言語での開発の場合、いざとなったときに外部から専門の要員を連れてくることも困難である。最悪のシナリオを考えて、慣れた言語での開発について、決定権のある顧客側キーパーソンとネゴシエーションをしておき、従来法での開発の準備をしておければよい
捉えるべき兆候	**対処例**
・いざとなったときに頼れる専門家が外部にいるかどうか	・設計中に別途チームを作って、新言語での実機能確認のためのプロトタイプを作成して確認する。同時に、この結果に問題がある場合の対処方法をユーザーと合意する

15 現行システムと新システムの並行開発

新法制度への対応のため、現行システムを、そのシステムを保守しているベンダーがリプレースするというプロジェクト。基本は現行システム機能を踏襲することになっており、新しい法制度の要件を加味したシステムにする必要があった。ところが現行システムも機能追加・変更が多く、現行システムへの機能追加・変更が、現行踏襲の新システムにも影響し、開発作業が重複するという事態になった

事例における見切りの内容	本来の見切りの考え方
新システムの開発で現行システム踏襲分の仕様を凍結し、新システムの結合テスト後に、それまでの間に現行システムに発生した機能追加分を一括開発することにした。しかしそのため、総合テスト環境と開発環境の構築が重複することになり、環境の構築、管理する要員の負荷が増え、環境ミスによる進捗遅れが生じてしまった	・結合テスト後に新システムに、現行システムで発生した機能追加分を一括開発する際、現行システムの保守メンバーを新システム構築のプロジェクト側にアサインできることを確認しておくこと
捉えるべき兆候	**対処例**
・現行システムの障害発生状況	・機能追加・変更に合わせて現行システム要員を更改システムに参加させる

16 運用テストに入ってから現場固有の業務要件が多数発覚

給与システムの更改プロジェクト。運用テストに入った段階で、実は現場でローカル仕様があることが判明。ローカル仕様の調査には、かなりの期間と工数が必要であったが、システム稼動後、ローカル処理を手作業でカバーすることにした。そのため、1年間にわたって、顧客側の運用コストが大幅にかかってしまった

事例における見切りの内容

現場でローカル仕様があることが運用テストで判明したが、他のローカル仕様の有無を調査せずに、システム更改を実施

本来の見切りの考え方

・ローカル仕様が膨大になることを想定して、新システムの移行後に、現システムに戻せるようにしておく

捉えるべき兆候

・設計工程でローカル仕様を含めた現場の業務調査を実施していない
・プロジェクト設計メンバーに顧客側業務部門が関与していない

対処例

・ローカル仕様がどれだけ存在するのかを把握し、もしその量が多い場合は、段階移行かリリースの延期をするなどの検討を顧客と実施する

17 標準仕様に先駆けてシステム構築したが、標準化への対応で品質問題が発生

ある先進的なシステムを開発していたが、他社ベンダーで同様なシステムを開発中との情報があり、それより先に提供し稼動させる必要があった。そのシステムに関連するJIS化が予定されている国際標準もあったが、独自仕様で開発し、納期に間に合わせた。ただし、その標準への完全準拠のために改造を行ったが、品質に問題が多く、顧客からのクレームがあった

事例における見切りの内容

その国際標準の国内でJIS化が進められていたが、その策定スケジュールと顧客側の納期が合わないため、独自の仕様で開発を進めた。あとでJISが策定されてから、改造すれば対応できると考えた

本来の見切りの考え方

・競合社よりも先に稼働実績を作ることに重きを置くか、他社に遅れつつもJISに徹底対応した安定システムを稼働させるか、どちらの方針なのかを明確にする
・JISと独自仕様がどのくらいのギャップがありそうかを見積もり、そのギャップを埋めるだけの改造コストと、JIS化に対応できなかった場合の業務上の影響度を天秤にかける

捉えるべき兆候

・将来予測されるシステム変更への対応方針がない

対処例

・JIS化で影響が生じる範囲を予測できれば、JIS化されても容易に対応できるようにプログラム設計する

18 仕様検討に時間を要し、十分なテストができないまま受け入れテスト（1）

短工期を求められていた上に、仕様検討に時間を要した。仕様の確定した機能から順次開発に着手したが、工期内でテストが十分できていない機能があった。 工期を守ることを優先し、その状態でリリースしたため、顧客の受け入れテストで仕様認識違いを含めて不具合が多発した

事例における見切りの内容

多少の品質の悪さは、リリース後の受け入れテスト時点での修正でなんとか対応できると考えて、工期を優先してリリースしてしまった

本来の見切りの考え方

- 品質が不十分と思われる部分について、その部分に問題があった場合に手作業や他の機能で業務を遂行でき、かつ、それに対して顧客側が納得しているのであればよい
- 顧客の合意を得ないまま、品質の低いシステムを受け入れテストにかけるのは、顧客との関係上、大きな問題に発展することもある

捉えるべき兆候

- 基本設計、詳細設計の進捗の大幅な遅れ
- 顧客の仕様決めの遅れ、仕様変更の多さ

対処例

- 上流工程であれば、実現のためのケースシナリオ（楽観、悲観、最悪）を決めておき、最悪のケースにおける、スケジュールと最小限の実現機能を顧客と合意しておく
- すでに受け入れテストに入っている場合は、受け入れテストを中断し、スケジュールを延期する。仕様再確認、修正後テストを再実施する。その後、顧客のキーマンによる仕様確認テスト（受け入れてスト前）を実施後、リリースする

19 仕様検討に時間を要し、十分なテストができないまま受け入れテスト（2）

基本設計で仕様確定に時間が掛かり、他に影響する可能性がある機能の詳細設計が遅れていた。その部分の詳細設計が完全に終わらないうちに、出来る部分からプログラミングに着手。しかしその部分のプログラミングで設計工程に起因する問題が多発し、手戻り量が多かった。その対応のために工期を守れなくなった

事例における見切りの内容

工期がまもれない場合、スケジュールの延期を顧客にお願いすれば、顧客は受け入れてくれる可能性はあると考え、できるものからプログラミングに着手した（結果的には顧客も受け入れてくれた）

本来の見切りの考え方

- スケジュール延期が許されるだろうというプロジェクトスケジュール管理の問題と、システム構築プロセスの問題を一緒にしてはいけない
- 影響範囲が広いシステム機能部分の詳細設計が遅れているときに、できたところから着手してよいと見切ることができるのは、詳細設計の後の変更が着手部分にどの程度の影響が出るかを把握しているときである。インターフェースの改修で先行着手部分が生き残るかどうかである

捉えるべき兆候

- 他に影響する可能性がある機能の詳細設計が遅れる

対処例

- 上流／下流工程にかかわらず、工期が遅れることが予想される場合は、カットオーバーのスケジュールの延長についてなるべく早く顧客と協議する
- 同様に、仕様の縮小や段階的なリリースを検討する
- 効果的であると判断される場合には、要員の追加投入を検討する（ただし闇雲な要員投入はかえって効率を悪くする場合もある）

20 顧客側メンバーの体制が弱く、仕様が詰められない

顧客側メンバーは、システム化の経験が浅く、現場部門に対する仕様決定権も弱いので、仕様確定には計画より期間がかかる可能性があったが、ベンダー側の戦略上、受注する必要があった。プロジェクトが始まると、顧客側メンバーが多忙との理由により、仕様検討の稼働や時間が十分に取れず仕様確定が遅れ始めた。顧客側の要件の提出期限をベンダー側から提示。顧客は、遅れをとりもどすためメンバーの増強を行なった。また、スケジュールを延期することとなり、当初予定していた納期は守れなかった

事例における見切りの内容

当初ベンダー側としては、顧客側メンバーのスキル不足を補う形で、ベンダー側で作成した仕様案をもとに打ち合わせすることにより、仕様確定をスムーズに行い計画通りに納めることができると考えて受注した

本来の見切りの考え方

- 顧客側の体制やスキルの不足が事前に分かっているのであれば、顧客側の体制強化を依頼する先（相談相手）を見極めておくこと
- 仕様のたたき台が作成できるほどのスキルがベンダー側にあり、現行システムのリプレース案件である場合は、現行システムのドキュメントをベースに仕様詰めする部分を絞るなどの可能性を検討しておくこと
- 契約時に顧客側起因のスケジュール遅延や予算オーバーについての免責事項を入れておくべきである

捉えるべき兆候

- 顧客側メンバーの経験が浅く、決定権も弱い

対処例

- 顧客側の責任を明確にした上で、顧客側の体制強化を依頼する
- システム検討範囲の縮小や工期延長について協議する
- 現行システムのドキュメントなどをもらって、それをベースに検討を進められるかを判断する

21 不条理な予算圧縮要請で開発に着手

システム化の対象領域の業務知識に乏しかったが、受注しないと、この先顧客との取り引きが続かないという危機感があった。請負契約をしてしまったが、大幅な工数増になり、プロジェクトの想定よりも採算が大きく悪化した

事例における見切りの内容

顧客からの「簡単にできる」という言葉にも反論できず、顧客予算に合わせた見積額の圧縮要請を受け入れて受注し、開発に着手した。赤字になっても受注するほうがよいと判断した

本来の見切りの考え方

- 予算の圧縮要請を受けても、それに見合うだけの前提条件について合意されていること（対処例参照）
- 工数増分と原因を把握しておき、都度顧客に説明し、できるかぎり費用追加を認めてもらうことは必要

捉えるべき兆候

- 対象領域の知見に乏しい
- 根拠のない費用圧縮要請

対処例

- 圧縮要請に対しては、それに見合うだけの前提条件を提示し合意してもらうこと。たとえば「現行システムのドキュメント提示」や「顧客側メンバーの業務知識の提供およびメンバーの積極的な参画」など
- （業務知識が受注側に不足している場合は難しいが）予算に応じた仕様検討対象範囲（開発範囲）の合意
- 仕様検討対象範囲（開発範囲）増に伴う再見積もりと対処策の事前確認
- 予算の増額要求

22 さほど大きな影響はないだろうと思い、既存システムのプログラムを修正

既存システムに対する追加機能の開発を受注した。追加部分の開発は、共通基盤部分にも手を入れる必要があった。ただ、契約範囲外ではあったが、現行システムのメンテナンス性の向上を図るために、自社費用で共通部分のプログラム修正を行うことにした。ところが結合テストでその修正に起因する不具合が多発し、手戻りによる工数増と工期の遅れが生じた。既開発プログラムの修正は、弊社理由によるものであり、これによるスケジュール延期を顧客に話すことができなった

事例における見切りの内容

修正による影響個所は少ないと見込んでいたので、修正部分の単体テストを十分実施しないまま、追加機能の結合テストを行った

本来の見切りの考え方

・状況から察するに、既存部分のテストのし直しが大きな工数になるため、テストのし直しをやらないで済ますことを考えていたと思われる。影響が少ないだろうという楽観的な見切りは行ってはならない。すでに稼働しているシステムに手を入れる場合は「何かが起こる」という危機感を持つこと
・最低限、いつでも現行の稼働状況に戻せるようにしておくこと

捉えるべき兆候

・そのプロジェクトで、戻し手順と戻しによる影響度について検討されていない場合は、既存分部の改修に対して甘く見ている

対処例

・既存改修部分にかかわる結合テストを優先して実行し、また追加機能部分ではない別の現行システムについてもテストを実施する
・既存改修部分に対するソースコードレビューの徹底

23 時間のかかる顧客側承認プロセスを見切って、承認前に作業着手

短納期のプロジェクトであったにもかかわらず、顧客側の意志決定プロセスがシステム部門・業務部門に関連し、複雑になっており承認までの時間がかかることが多く、顧客側の検討に多くの時間が費やされてしまった。顧客側から提示された仕様が、想定して製造していた仕様と大きく異なってしまい、テスト実施中にもかかわらず大幅に手戻りが発生した。顧客との打合せは週1回で実施され、仕様の確認プロセスに非常に多くの時間が費やされていた

事例における見切りの内容

仕様の顧客承認を得るまでに時間がかかりすぎる上に、承認を待ってから着手したのでは納期を守れないと判断したため、承認を得る前に製造に着手した

本来の見切りの考え方

・正式には顧客側承認を得てから着手するべきだが、顧客側承認を見切るには、最低限、顧客側キーパーソンの仕様に対するコミッションを得ていることである。たとえ顧客側で、その仕様に対する反対意見が出ても、声の大きいキーパーソンであれば、反対を押し切ってでも当初の仕様で押し切ってくれることが多い
・上記の見切りでは、誰がキーパーソンかということをよく把握していることが条件になる

捉えるべき兆候

・顧客側担当者と仕様案の確定度合いについてヒアリングし、もし「協議の場に出してみないとなんとも言えない」という回答である場合、その仕様がひっくり返される可能性がある(またはそれで押し切ろうという担当者の熱意がない)

対処例

・顧客側のキーパーソン(必ずしも担当者とは限らない)を見定めて、仕様に対する内定をもらっておく。できればキーパーソンと思われる人物を複数人押さえておく
・工程(マイルストーン)と先行着手のリスクについて顧客と協議し、リスク発生時の責任分担・役割分担について合意を得ておく
・顧客にプロジェクトのスピード感に応じた体制をとってもらう(線表を後ろから引くと先方にも危機感が生まれる)
・仕様(開発範囲)増に伴う再見積もりと対処策の事前確認

24 契約金が未決定ながら進めたプロジェクトだったが、契約時に予算額が大幅に減額

顧客から「契約する」との約束があったが、契約金額については未調整の状態であった。案件は短納期でかつ納期厳守のプロジェクトであった。時間もないので作業着手。しかし、契約締結時には想定していた金額よりも受注額が少なかったため、コストが超過し採算が取れなくなった

事例における見切りの内容

契約金額が確定されないまま、想定される規模に相当する人員を投入して開発作業に着手した

本来の見切りの考え方

- 契約締結前に作業を開始することがそもそもまずいが、たとえ締結前であったとしても、契約前作業をすることの覚え書きを書いてもらうようにする
- その上で、システムの開発範囲のうち、優先度の高いもの（すなわちシステムのコア部分）を先に開発するなどし、予算を低減されても開発はするだろうという部分に着手する

捉えるべき兆候

- 金額については別途調整と顧客に保留された
- 未契約状態でのプロジェクト立ち上げ

対処例

- なかなか契約金額が決まらないという状況である場合、見積有効期限を明確にして、そこまでに契約をしてもらわねば納期が守れないことを明確にする
- これから着手するという段階において、必要最低限の機能範囲に絞り込んで着手するなどのリスク低減を検討する。その際に営業担当者から顧客の予算額についての確度を確認しておく
- 作業開始後であれば、要員投入と作業／稼働実績の顧客報告を行い金額交渉を行う

25 保守フェーズにおける作業慣習で、仕様変更の見積もりとともに作業を着手してしまった

常日頃から保守SEは顧客からの依頼があったら対応するという慣習があった。ただし今回は、顧客は緊急依頼として、仕様変更を見積もってほしいと保守SEに依頼した。顧客の都合で、仕様変更は実施しないことになり、仕様変更設計や開発作業が無駄になった

事例における見切りの内容

これまでの慣習から、顧客側の承認は後から得られるものと考えて、通常通り開発に着手した

本来の見切りの考え方

- この事例は慣習にしたがって作業着手したということで、「開発することになるだろう」という実施の見込みをして作業に入っている。本来してはいけない内容
- やってしまった分は泣くしかないが、開発成果物や検討内容などの情報は保管しておき、次期システム開発などにて回収できるように調整する

捉えるべき兆候

- プロジェクト・マネージャが「誰が今何をしているのか」を管理できていない
- 保守案件の仕様変更や機能追加などのテーマ管理がされていない
- 忙しくないはずなのに、忙しくしているメンバーがいる

対処例

- 合意内容について文書化を行っておき、作業実施記録も詳細に記録しておく

26 最新技術てんこ盛りで、納期も短いシステム構築で手戻り多発

重要顧客の記念事業として、最新技術をふんだんに盛り込んだ社内システムを計画していたが、技術的に困難が予想され、しかも短納期で開発する必要があった。プロトタイピングなどあらかじめ検証する工数や工期も取れなかった。設計～製造フェーズに入り、実現できない機能が多発し、一部は要件定義に手戻りが生じた。結果、工数増大し不採算プロジェクトとなった

事例における見切りの内容	本来の見切りの考え方
実現性、生産性、性能など未知の領域を残したままだったが、顧客側の押しも強く、なんとかなるだろうと思い開発に着手した	・通常、新技術を熟知した人材を集めることは困難である。人をかき集めればなんとかなるという発想は危険。人員追加投入を繰り返し、不採算プロジェクトとなることも多い ・本来は委任契約と請負契約を使い分け、技術リスクの高い工程は委任契約とすべき。重要な顧客であるなら、なおのこと、実現性の低いプロジェクトに対して「ノー」と言うべきである
捉えるべき兆候	**対処例**
・引き合い時の顧客の過度の期待	・開発が間に合わなかった場合に備えて、既存技術での実装や他パッケージの利用も想定しておく。それに備えて、どのくらいの遅れが発生したら開発方法を切り替えるかを検討し、顧客と合意しておく

27 DBMSが詳細設計後に決まって手戻り

実績、機能、価格の異なる2つのDBMSのいずれを使うか顧客が迷っていた。また、マルチベンダーでの開発案件だったが、開発ベンダー間でもそれぞれのDBMSを押す勢力ができてしまい、お互いに譲らない形になった。そのため顧客も更に迷い、DBMS決定に手間取った。詳細設計終盤で顧客が、当プロジェクトの想定DBMSとは別のDBMSに決定。一部詳細設計が無駄になった

事例における見切りの内容	本来の見切りの考え方
プロジェクトの特性や世の中のトレンドを考え、一方のDBMSを使うことになるだろうと見切り、詳細設計に着手した	・想定外のDBMSになったときにリカバリが間に合うのであれば、先行着手することも可能 ・そのためには、システムの設計や実装のうち、どの部分がDBMSの製品に依存するかを把握できており、さらにDBMSが想定外にものになった際の回収コストと期間を見積もっておく必要がある。プロジェクトで想定する期間・予算のバッファ（保険）があると思うが、それに収まるかどうかを判断すること
捉えるべき兆候	**対処例**
・基本設計時のDBMS選定に関するすれ違い	・まだ実装着手前である場合、顧客にDBMSの製品特性が実装方式に与える影響を説明し、早期の決定を促すこと ・実現方式に依存する部分としない部分を明確にして開発を進めるように周知徹底する。もし他のDBMSになった場合を想定して、他DBMS案を推してきたチームに対して技術提供をしてもらう ・別部署へ応援手配し最低限の技術者を確保すること

28 開発効率向上のためのフレームワーク作りがボトルネックになってしまった

ソフトウェア開発フレームの標準化による開発効率性と保守性の向上を図るため、専用のフレームワークを開発し、その上に業務プログラムを乗せることにした。専用フレームワークの設計・開発にアサインした人材の能力が足りず、設計作業がずるずると遅延した。このままではすべての開発が止まるため、専用フレームワークの開発を止めた。最終的に、フレームワーク機能を縮小し設計作業を進めたが、似たような別モジュールが多く設計されてしまい、開発効率が低下し、工数が増え、工期が遅れることとなった

事例における見切りの内容	本来の見切りの考え方
専用フレームワークの設計作業に、適した人材をアサインすることができず、なんとかなるだろうと思い、別の人材をアサインした	・本プロジェクトでは専用フレームワークの構築がクリティカルパスであり、重要なマネジメントポイントである。アサインした人材が適していたか適していなかったかは結果論であって、本来は重点的なマネジメントを実施して、危険と判断したら人員投入するか、フレームワーク開発そのものを諦めて他のフレームワークを利用するなどの対応策を発動するべきであった ・クリティカルパスが何かをよく考え、そのクリティカルパスに問題が発生した場合の代替案を準備しておくことと、その代替案切り替えの判断基準を決めておくことが重要である。代替案と代替案を発動した場合の影響範囲を把握し、影響度合いがプロジェクト内で処理できるレベルであれば、見切ってもよい
捉えるべき兆候	**対処例**
・フレームワーク設計メンバーの能力不足 ・フレームワークの開発ドキュメントのできばえ	・フレームワークの開発が失敗したときのことを考えて、機能縮小の影響度合いや範囲を確認しておき、その範囲について顧客と合意しておく ・フレームワークの開発が失敗した場合の代替案を準備し、代替案発動時の影響度も考慮しておく

29 プロジェクト・マネージャに大規模プロジェクトの経験がなく、結合テストの頃から調整不足が露呈

大規模プロジェクトを受注したが、プロジェクト・マネージャを任命する責任者に大規模プロジェクトの経験がなく、そのプロジェクト・マネージャが適任かどうかを判断する能力がなかった。要件定義から設計、プログラミング、サブシステム内結合テストまでは、計画通りに順調に進んでいるように見えていた。しかし、サブシステム間テストの段階で、結合テストの進め方やテストデータの準備を誰がするのかといった調整不足が露呈し、テスト作業が停滞してしまった

事例における見切りの内容	本来の見切りの考え方
責任者が、大規模プロジェクトをマネジメントできるスキルを確認せずに、プロジェクト・マネージャを任命した	・大規模システムかどうかに限らず、プロジェクト・マネージャがそのプロジェクトを制御できるかどうかを、プロジェクト・マネージャ任命者は見極めなければならない。少なくとも、そのプロジェクト・マネージャのスキル不足による問題が発生することも考えて、プロジェクト支援のための組織（PMOなど）をバックアップに付けておく ・後進を育てる意味もあるが、そのプロジェクト・マネージャが倒れた場合もしくは機能しない場合を想定して、有能な若手をサブリーダーに付けておけば、プロジェクト・マネージャ交代時のリカバリも早い
捉えるべき兆候	**対処例**
・プロジェクト・マネージャ自身がリスクとして認識していないので、不調の予測は難しい ・上流工程の段階で、下流工程での作業内容やリスクについて想定していない（想定できない）場合は結合テストの頃からまごつく可能性あり	・組織として、高い頻度でプロジェクト実行状況のレビューをして、プロジェクト・マネージャが適任でないことを早く見つける ・一刻も早く、必要スキルのあるプロジェクト・マネージャを投入する

30 未知のパッケージを協力会社から提案してもらったが、そのパッケージに品質問題が発生

顧客に対する提案内容として、自社内では、ある中核機能に関する実現手段を持っていなかったため、協力会社に提案してもらったパッケージを利用する方法を採用した。しかし、いざその方法で構築作業に入ったところ、提案されたパッケージに重大な品質不良が発覚した。提案してきた協力会社配下にて、さらにパッケージベンダーとハードウェアベンダーの間で、原因の切り分けができず、対策に時間と費用を要した

事例における見切りの内容

中核機能の実装にあたって、協力会社がパッケージを提案してきたが、プロジェクト・マネージャにはパッケージの内容については理解不足で判断しようもなく、協力会社を信じてパッケージの品質には問題がないだろうと判断した

本来の見切りの考え方

- 機能実現にあたっての重要な機能および性能についてリストアップしておき、その部分についてのパッケージの品質を確認しておく必要がある
- そのパッケージが実は開発中のベータ版しかなく、品質確認が難しい場合は、少なくとも代替運用か、または代替パッケージを用意することによる抜け道を作っておく必要がある

捉えるべき兆候

- パッケージまたはソリューションの品質または性能に関して、ベンダー側のコメントがない。もしくは品質・性能の根拠を示さない

対処例

- 必要最小限の機能、性能でとりあえずリリースする。その後、性能向上を図る
- パッケージベンダーに機能保証をしてもらうとともに、トラブルへの迅速対応のために多階層の関係をフラットにするようリーダーを決めておく

31 経験の少ないオープン系システムで、プロジェクト・マネージャがコントロール不能に

未経験のオープン・システム基盤を用いたシステム開発案件で、顧客にとって非常に重要であり、セキュリティ要件や性能要件など、コントロールの難しい要素が多数あるプロジェクトであった。ベンダーの経験不足・知識不足もあり、安易な体制でスタートしたため、プロジェクト・マネージャは、上がってくる懸案事項や日々のタスクを処理しきれない状態になり、統括できなくなってしまった

事例における見切りの内容

受注前に十分調査をせず、マネジメントの問題が発生するだろうと想定することもなく、安易なプロジェクト体制で受注した

本来の見切りの考え方

- 新しい開発方法論や技術などを利用する場合、どのようなことが起こるかを想定することが難しいため、リスクも多く、問題となって顕在化する可能性も多い
- 基本設計の段階で専門家などの有識者によるレビューをするなどの手段を講じることができるのであれば、軟着陸できる可能性はある

捉えるべき兆候

- 要件の確定が遅れる（業務改善検討結果が影響している）

対処例

- 基本設計の段階からオープンシステム基盤の専門家など有識者の投入をする

32 複数会社でのプロジェクトで旗振り役が不在

ポータルサイトの実証実験に複数社が参加し、プロジェクト全体としての実験項目だけが決められていた。実証実験システムの開発プロジェクトの全体をとりまとめる体制がなく旗振り役がいなかったため、発注側が意図している実験項目が消化できないほどシステム開発が遅れた。システム開発と実証実験のスケジュール調整もできていなかった。各ベンダーでのプロジェクトコントロールがきかなくなってしまった

事例における見切りの内容

発注側は、各ベンダーで意図するとおりの役割分担になっているだろうと見切ってスタートした（ベンダーの立場から考えると自社の範囲分についてあらかじめ与えられた予算内で一定の成果さえ達成すればよいので、全体的な管理を考える必要性はなかった）

本来の見切りの考え方

- 役割分担や開発範囲についてはそこが起きるのが当然であるという意識を持つべきで、発注側はオーナーシップを発揮しなければならない
- 「うまいことやってくれるだろう」という淡い期待だけで、プロジェクトの成否にかかわるスコーピングを見切ってはいけない

捉えるべき兆候

- 要件を確定するのが誰かが曖昧である場合、プロジェクトのオーナーが実質不在であると考えてよい
- お互いが領空侵犯しないように気を遣っている、もしくは各々の事情で情報公開をあまりしない傾向がある場合は、開発範囲で抜け、漏れにつながる可能性がある

対処例

- 発注側による積極的なプロジェクト参画は必須
- 発注側と受注側の複数社それぞれからマネージャ／リーダークラスの代表者を選出してもらい、意志決定を行うための合同会議体を作る
- または、取りまとめ役のベンダーを指定して、リーダーシップを発揮してもらうようお願いする
- すでに不調に陥っている場合、現状調査を行って課題の洗い出しを行い、担当者を取り決めるなどの旗振りを発注側主導で進める

33 顧客指定のフレームワークで受注し、大幅コスト増

顧客からの提案要請時に、アプリケーションフレームワーク（FW）を利用しての開発が前提になっていた。開発を進めるうちに、FWが機能的に不十分であることが分かり、FWの上に載せるアプリケーション側で機能を補完することになった。FWの品質や性能にも問題があり、アプリケーション開発部分のテスト工程で、障害が多発したり性能がでないなどの問題もあり、ハードウェア増強など大幅なコスト増大となった

事例における見切りの内容

FWがターゲット業務に適合するかどうか、実現可能なのかを吟味せずにFWを適用することが決定された

本来の見切りの考え方

- FWの適用に関しては、想定している機能に関する性能や品質について見通しが立っているかを考える必要がある
- その見通しが立てられない場合、そのFWが全く使えないことを想定して作り直す場合や他のFWを持ってくる場合の工数を見積もっておき、線表を後ろから引いて、“顧客指定FWの品質を判断するエンドポイント”を決め、それまでにFWを評価できるかを考える
- FWの利用はもともと効率性を考えた上での戦略なので、上記のように代替案で後ろから線表を引くというのは現実的ではないことが多い（代替案のほうが期間がかかることが多い）。この場合、プロジェクトとしては顧客指定FWに問題があった場合の免責事項を契約条件の中に含めることでリスクヘッジする必要がある

捉えるべき兆候

- 顧客側にそのFWを実地に利用した経験がない
- そのFWの評判がよくない
- FWそのものが新製品である、または、導入実績が少ない
- FWについて性能評価結果が存在しない

対処例

- 上流工程においては、なるべく早期にFWの品質検証をする期間を設けることを計画する。同じFWを使った他のシステムやプロジェクトを調査するとよい
- また、FWの品質・性能に問題がありそうな場合を想定して、顧客との合意の上で、代替FWを用意する手はずも整えておく
- 体制強化として、その顧客指定FWの経験者を開発メンバーに加えたり、FW開発者と技術支援契約を結ぶ
- 下流工程に入ってしまっている場合は、事例にあるようにハードウェアを増強したり、ボトルネックを探るなどの地道な性能改善をするしかない

34 本番データに現場部門のパッチが当たっており、論理的不整合が多く移行コスト増大

長年使用してきた現行システムの再構築案件。現行システムのデータを、正確に新システムに移行することが業務上必須であった。ところがいざプロジェクトを開始したところ、売掛残(債権管理)システムについて、現行の仕様が不明確であるためデータ移行ロジックを正しく作れないことが判明。しかも移行すべきデータに、現場部門で手作業でデータ修正を加えて運用していることに気づいた。結果として、データ移行に多大な費用と期間がかかり、カットオーバー後も業務上の障害が発生しつづけた

事例における見切りの内容

提供してもらう予定の現行システムの仕様書が正確で、かつ移行するデータが論理的にも整合しているはずと考えてスケジュールと費用を算出してプロジェクト計画とした

本来の見切りの考え方

- 現行システム踏襲型のプロジェクトでは、現行システムに関する情報がどれだけ提供してもらえるかがプロジェクトの成否を握る。したがって、現行システムに関する仕様書などを見て、どのくらいの作業量があるかを見積もった上で、スケジュールと費用を算出する必要がある
- 現行データの現場部門での手作業での修正(パッチ)まではなかなか想定できない。現行システムに入っている元データに論理的な整合性がない以上はプログラムでの移行は困難。致し方ない事例
- 移行ツールを設計するにあたっての前提条件を明確にしておいて、データに問題があったり仕様書などの十分な情報がなかったりした場合の再スケジュールなど、調整余地を残すように顧客と合意しておくこと

捉えるべき兆候

- 移行データの詳細仕様が、催促を繰り返してもなかなか現システムの担当者から提示されない

対処例

- プロジェクトの前提条件に現行システムの仕様書の提供やデータ要件、運用状況などについての情報提供をしてもらうことと、それらに不備があった場合には再スケジュールや再見積もりをすることを明記する
- すでに事例のような問題になっている場合は、移行データの手作業パッチの記録や不整合パターンの洗い出しを行う
- 移行データの状況判定プログラムを作成し、移行対象データを徹底的に調査する

35 現行システムを再利用した新システムを導入しようとして例外処理に苦しむ

長年利用して保守が困難になっている複数の現行オンラインシステムがあり、利便性向上のためにフロントエンドに新しくワークフローサーバーを置いて、新しい業務を、ワークフローサーバーによるシステム間連携で実現するというのが顧客の要望だった。ところが現行システムの内容を知っている技術者は居らず(そのベンダーはすでに存在していなかった)、現行システムとの結合テストでは例外処理など予想と異なる動きが多発。テストが進まなくなった

事例における見切りの内容

プロジェクト体制に、現行システムの内容を熟知している技術者を参画させることが出来るかどうかを確認せず、プロジェクトを計画した。現行システムに関する技術者がいる、もしくは、システムの仕様が明確であろうと思って進めてしまった

本来の見切りの考え方

- 現行システムに関する仕様が明確であることが前提条件であることはこの事例の中にも表れている。この仕様が明確である、もしくは熟知した技術者をプロジェクト体制に組み込むことができることが前提条件であるので、これらの前提条件が覆されたときの対応処置案がきちんとあれば、プロジェクトを進めてもよい
- 実現可能性について懸念がある場合は、実現するレベルを抑えたり(たとえば機能を絞るとか、運用方法を利用側で注意するなど)、最悪の場合はプロジェクトを中止するなどの判断基準を顧客と取り決めておく、といった対応策を準備しておく

捉えるべき兆候

- 見積もり時から、現行オンラインシステムの仕様が十分提示されていない(特に入力エラー時などの障害処理)
- 現行システムが非常に古いシステムである(技術者や仕様を決めた者がすでにいない可能性が高い)

対処例

- まだ上流工程であれば、実現可能性について検討し、実現にあたっての課題出しから始める。そのために、現行システムの調査をしてから、新システムの実現方針を考える
- 下流工程ですでに事例のような不調に陥っている場合、実現レベル(実現機能や運用制限など)について顧客と合意した上で、スコープを決めて開発に入るか、もしくはプロジェクトを中止する
- 例外処理や処理の組み合わせパターンを把握できないなど、今後の作業量が見えないうちは、開発作業に入らないようにすること

36 顧客側からの要件出しが細切れで、なかなか要件を確定してくれない

教育機関の顧客で、他社が開発した旧システムをリプレースする案件。開発担当社側には業務ノウハウが無かった。要件定義工程では、顧客からの仕様提示が細切れで、なかなか要件が確定しない。プロトタイピングしながら進めたが顧客は否認を繰り返すばかりで大幅な納期遅延が生じた。ついには要件定義もそこそこで設計に入ったが手戻りが多発。プログラムの品質も悪く、開発要員を多数投入したが納期遅延。要員も疲弊し、デスマーチプロジェクトとなった

事例における見切りの内容

受注側は顧客側のノウハウ不足のため仕様が確定できないのだと判断し、プロトタイピングとして一部業務について仕様が確定しないまま先行開発を行なった

本来の見切りの考え方

- システムのプロトタイプを見せることによって顧客側の新業務内容検討を促進させることができる場合もあるが、それは受注側に豊富な業務知識・経験があり、それをベースに適切なプロトタイプを提供できるときである。勝手なイメージだけでプロトタイプを作っても、まったく別の業務処理が策定されて、そのプロトタイプが無に帰すことになる可能性が高い。事例のように、顧客に納得してもらえない場合も多い
- 顧客側から「こういうイメージのプロトタイプを作って欲しい」という要請があった場合を含め、プロトタイプの内容について顧客側と合意が取れていればプロトタイプを進める価値はある。いずれにしてもプロトタイプの効果を考えてから開発に進むこと

捉えるべき兆候

- 客先からの仕様提示が細切れ、曖昧である
- 客先の意思決定の責任が分散しており、キーパーソンが不明

対処例

- 顧客側に構築期間に対する危機感がない場合は、仕様凍結の時期を明確に宣言し、顧客側で早急に決めてもらう必要があることを理解してもらう
- 顧客側に要件を決める能力が不足していると思われる場合は、要件定義の舵取りをする。検討体制や意志決定プロセスの定義・確認からシステム化対象範囲の確定、検討事項の整理などを進める。そのための上流工程要員を体制に新たに組み入れるなどの対応を進める

37 既存資産が流用できると思っていたが、流用できないことが分かりコストオーバー

とある地方自治体のシステム化案件。他の自治体で、同じ業務処理をするシステム案件を実施したことがあったので、そのときの成果を流用できると判断して見積もりをし、開発を着手した。ところが、開発を進めていくうちに、業務処理が想定と大きくことなり、既存のシステムは流用できないことが判明。結果として大幅に開発規模の増大し、開発期間も長期化した

事例における見切りの内容

類似システムが流用可能と考えて、それを前提に原価を低めに見積もり、プロジェクトを計画した

本来の見切りの考え方

- 類似システムを流用できるだろうと見込んで開発に着手し、当初想定規模を大きく上回る開発規模になるということはよくあることであり、既存資産を流用することを前提とする計画立案時には注意が必要である
- 流用部分なしで新規開発する想定で見積もりを出すと、他のベンダーに価格面で負けてしまうなどの営業的な要素も絡むため、流用の適用／非適用の判断はプロジェクト・マネージャだけでできるものではない。既存資産の流用がプロジェクトの前提であるなら、既存資産の流用をすることと流用が適わなかった場合の費用・納期の変更について顧客側と同意を取ること

捉えるべき兆候

- 費用削減だけを目的とした既存資産流用
- 業務名、システム名だけで既存資産流用可能と判断する見通しの甘さ

対処例

- 流用しない場合の見積もりと流用した場合の見積もりを顧客側に提示し、流用が適わなかった場合の費用追加・納期の延期について合意を得ておく
- 流用部分についての適用可能性について評価する工程を追加し、その評価結果に応じたその後の進め方について事前に検討しておく

38 顧客との共同開発という名目で、イニシアチブが取れずに要件がなかなか確定できない

医療法人である顧客と、開発受託会社とで費用折半でシステムを共同研究・開発することになった。開発したシステムは、開発受託会社で他の医療法人へのパッケージ展開する計画で顧客とも合意していたが、その顧客にしか適用できないような個別要件が噴出し、歯止めが利かず、設計手直しなどの手戻りが多発。共同開発ということで、開発者側もイニシアチブを発揮することができなかった。結果、ローカル仕様ばかりの、他社展開できないようなシステムができあがった

事例における見切りの内容	本来の見切りの考え方
顧客側の仕様確定が遅延していたが、共同研究のメリットを重視し、仕様未確定であるにもかかわらず開発を開始した	・共同研究や共同開発の場合、顧客側と受託側とで、役割分担が曖昧なままプロジェクトが進行することが多い。業務ノウハウをなるべく引き出して他社展開できるパッケージを作りたいというシステム開発側と、たくさんの要件を入れ込んで自分たちにとって使いやすいシステムが欲しいという顧客側との思惑が一致して、要件が山のように膨れるが、金と納期が決まっているので「どこまでやるのか」という仕切りが必要になる ・全体の要件は未確定でも、確定部分と未確定部分の整理ができていれば、確定部分を先行着手することは可能
捉えるべき兆候	**対処例**
・共同研究という名目のもと、顧客と開発担当会社との役割分担や責任所在が明確になっていない ・パッケージ化対象機能かローカル対応機能かの判断基準がない、もしくは曖昧	・要件の確定部分と未確定部分の切り分けと、未確定部分による全体への影響度を調査する ・未確定部分による影響が多少あっても先行開発のほうがメリットがあると判断したら着手する。万が一のリカバリ手段も用意しておく ・パッケージ部分への機能取り込みか、ローカル開発部分への機能取り込みかの判断基準を明確にし、ローカル開発部分は追加発注であることを宣言する

39 何でも言うことを聞くという応札条件の案件

医療法人の顧客で、顧客要望を全て取り入れるということが応札条件。顧客要望が膨大に発生し、仕様取りまとめるはずの客先窓口が破綻したため、要件発散。パッケージのカスタマイズを設計したところ、提案したパッケージでは品質・納期を満たせないことが判明し、別パッケージに切り替えて開発しなおしたため、二重開発となり、費用が想定より大幅に増大してしまった

事例における見切りの内容	本来の見切りの考え方
既存パッケージソフトを流用し、一部カスタマイズすることにより、顧客要望が満たせると判断して応札した	・開発範囲と作業期間が決まらないと非定型業務であるプロジェクトを定義し、完了させることができない。したがって、「全ての要求を取り入れる」という終了条件を定義できないような前提条件については、開発範囲と作業期間の制限から顧客側にある程度の要求の押さえ込みを覚悟してもらう必要がある ・プロジェクトのスコープとして受け入れられる要件の量に限度があることを理解してもらい、開発対象範囲を絞ることと、その後の要件については追加案件ということで別途費用をもらうなどの合意を少なくとも取ること
捉えるべき兆候	**対処例**
・すべての要求を聞くというプロジェクトの完了条件が曖昧な応札仕様	・応札の見直し ・応札してしまったとしたら、納期について合意し、用意できる開発要員のリソースから開発可能な規模を算定する。それに見合うような要件までを開発するということを合意してもらう。開発範囲をなんとかして決める必要がある ・最悪のケースとして、ペナルティを払ってでもプロジェクト中止・撤退

40 パッケージの品質と機能は十分に確認しよう

パッケージを利用したシステム開発。開発途中で、パッケージの品質不良が発生し、当初予定した機能の開放が遅れ、システム改修や機能追加が多量に発生した

事例における見切りの内容

パッケージの品質・機能・性能について、大きな問題は発生しないだろうと判断した

本来の見切りの考え方

・パッケージ、特に新規に使うパッケージの場合は、品質、機能を十分に評価して、適用の可否を判断することが重要である
・リスクが高いが、戦略的な理由で、パッケージを使う必要がある場合は、パッケージ部分を先行する、代替案を考慮しておくなどのリスクヘッジが重要である

捉えるべき兆候

・利用するパッケージの品質、性能を確認せずに、また、利用する機能の提供スケジュールの担保をとらずに開発を進めた

対処例

・パッケージ側の体制強化により品質強化を図る。体制を強化し、再スケジュールする

41 仕様変更に関する扱いは、顧客と意識を合わしておくこと

全国の拠点に導入するシステム開発。ハードウェアも含めSIとして受注。ハードウェアの納入遅れにより、サービスインの遅れが発生。また仕様追加、変更についいても、顧客と交渉したが、有償対応が認められなかった。全国の拠点に導入する支援作業も、顧客との作業規模、範囲に意識違いが発覚し、大幅に増えた

事例における見切りの内容

仕様追加の場合の再見積もりなどの取り決め、支援作業の内容と範囲、ハードウェア調達遅れの取り決め、などを明確にしないまま、開発を進めた

本来の見切りの考え方

・ハードウェアの納期、仕様変更、付帯・支援作業は、納期やコストに与える影響が大きいので、あらかじめ顧客と調整しておく

捉えるべき兆候

・仕様追加、変更時の取り決め、支援作業の具体的な仕切りがないまま、開発が進んだ

対処例

・仕様変更の取り扱い、作業支援範囲の再調整
・サービスインの時期調整
・費用の扱いを明確にしておく

42 大幅な再利用を前提とした開発は要注意

他システムの再利用80%を前提にシステム開発を計画。プロジェクトの独自仕様、業務プロセスの違い、仕様の増加に伴い、再利用率が大幅に低下し、開発規模が増大した。また、仕様確定が遅れた部分に先行着手し、結果、手戻りが発生した

事例における見切りの内容

他システムと類似しているため、かなりの部分を再利用できると考え、見積もりを実施した

本来の見切りの考え方

・再利用を前提とした業務システムの開発は、パッケージ適用の場合と同様に、業務プロセス、要求仕様を十分に明確にした上で、適用可否の判断をする必要がある。もし、適用を決めた後で、適用できない部分が多いことが判明した場合は、再利用できる範囲が大幅に減少するため、納期遅延と大幅なコスト増になる

捉えるべき兆候

・再利用可能な部分を検証せず、契約した

対処例

・体制の増強、管理体制の強化、再スケジュールの実施
・パッケージの利用率が変わったときの扱いを明確にする

43 作りながら仕様を確認

ノウハウの無い業界相手にパッケージ製品を導入し、仕様の詳細は顧客にヒアリングしながら開発していく計画を立てていた。しかし、ヒアリングを進めると見た目や機能名からは想像できないような仕様が続々明確化され「無いと業務にならない」「常識だ」という言葉に踊らされて開発規模が増大。納期遅延を引き起こした

事例における見切りの内容	本来の見切りの考え方
知らないことを自分の都合の良い方に考え、「仕様は不明確だがベースとなるパッケージが必要な部分を充足してくれているだろう」と期待してしまった	・分からない部分を明確にするか、制限とすることで仕切りを作る必要がある ・システムの使用者は理想を語るので、聞いているうちに仕様が膨らんでいくことはよくある。ヒアリング結果と現実を比較する(パッケージ機能とヒアリング内容のフィット&ギャップを取る)作業と、やる、またはやらないの検討を別にする必要がある
捉えるべき兆候	**対処例**
・「よく分からないけど大丈夫」と思ってしまった。もしくはそう説明された ・ユーザーの提示する仕様の反映で仕様検討が遅延し始めた(仕様が明確にならない場合も、後で噴出する可能性がある) ・パッケージのカスタマイズ	・業務をよく知る人員の投入 ・仕様確定に向けた不明確部分の明確化と期限管理強化 ・品質確保に向けた変更管理体制の強化 ・パッケージの拡張の扱いを明確にする(戦略案件など)

44 知らないもので決まっていないものを作る

運用開始の日程が確定している状態で仕様検討から開発を開始した。仕様確定が遅延し、納期が迫っている状態であったにもかかわらず、未経験の新技術を採用して難度の高い開発を行った。その結果、開発効率が落ちた上に突貫工事で発生する不良が増加し、稼働品質を確保するために納期遅延を引き起こした

事例における見切りの内容	本来の見切りの考え方
残期間から逆算した計画を立てて対策を打つ必要があったが、新技術採用のリスクバッファを踏まえて「そうは言ってもこれくらいの期間があれば出来るだろう」と甘く見てしまった	・納期ありきで計画を立てる場合には、必ずスケジュールは逆算して検討する必要がある ・未確定な要素(新分野開拓・新技術採用など)は、悪い方で検討し、常にアンテナを高くし、コンティンジェンシープランを立てておく必要がある ・仕様検討という発注側意思が大きくかかわる部分でスケジュールに影響が出る物は、その影響を明確にして発注側の責任を明確にする必要がある(製造側にも説明責任がある)
捉えるべき兆候	**対処例**
・仕様検討で工程遅延が発生した ・工程分割(リスク軽減)した受注になっていない	・技術ノウハウを持った要員の投入 ・仕様変更と取り込むべき仕様の明確化と管理強化 ・要求を出してくるユーザーとの意識差解消と、顧客側トップへ影響把握の申し入れ

45 無理な見積もりを通すと作り手か使い手のどちらかが破綻する

顧客予算に合わせ強引に原価低減した提案を通した。パッケージ機能に可能な限り準拠した仕様とすることを前提として受注したが、曖昧な線引きのため、カスタマイズが必要となる要求が大量に発生。仕様を確定できないまま原価山積みに合わせて開発を併走させ、手戻りも多発。最後は納期遅延となりペナルティが発生してしまった

事例における見切りの内容

「パッケージ機能を前提とする」という言葉で改修量を制限できると考え、改修量や内容のしきい値を決めていなかった

本来の見切りの考え方

- 前提条件が変わる場合には、その影響を明確にして示す必要がある
- 開発範囲の明確化と意識合わせを前提とした上で仕様の調整に入る必要がある

捉えるべき兆候

- 予算に合わせて受注額を決めている
- 前提条件が破綻した
- パッケージのカスタマイズ
- 仕様確定に遅延が発生した

対処例

- 顧客承認がなければ先に進まない部分を一括範囲に含む場合は、そのリスクと影響を顧客にも理解してもらった上で話を進める
- 開発範囲の再確認と徹底
- 範囲外作業の有償化
- 顧客との合意を徹底
- 前提条件が変わったときの扱いの明確化

46 個別に検討したものを繋げてみたら、繋がらなかった

大規模なシステム開発を分割して数社に、それぞれ仕様検討から発注した。個別に開発して出来上がったシステムで業務全体を通すテストを行ったところ、システム全体で仕様の整合性が取れていない状態になっていた。仕様整合性の再検証から計画を立て直し、予算大幅超過の上、納期遅延で顧客に迷惑をかけてしまった

事例における見切りの内容

一次請けとして全体を管理しなければならなかったのだが、分配して各社に任せた時点で「個別の機能が出来上がれば全体も出来るだろう」と考えてシステム全体としての取りまとめを放棄してしまった

本来の見切りの考え方

- システムを稼働させる視点を持ってプロジェクトを見た上で、各社に開発を依頼する必要がある
- 取りまとめ者として各社間インターフェースや全体仕様など、隙間を埋めたり土台を作ったりする作業が必要である
- 顧客と各社の仕様調整については、全体影響がある内容と個別で済む内容の切り分けと調整をする必要がある

捉えるべき兆候

- システム全体を説明できる人員がいない
- 稼働責任者が明確になっていない
- 各社間コミュニケーションの場がない
- 顧客と開発側のレビューワーが機能ごとに違っている

対処例

- システム全体仕様の再確認
- 管理体制の強化
- 稼働時期調整と、それに向けたリスタート
- PMOなど、監査組織の設立

47 顧客側の体制、顧客との役割分担、作業分担を明確にすること

顧客が指定する独自ミドルウェアを利用したシステム開発で、業務やその独自ミドルウェアについて十分な知識・スキルがない体制で開始。顧客側は、現行業務の分析やミドルウェアの情報開示に協力的ではなく、上流工程で大幅に遅れが発生し、後工程でも、手戻りが多く発生して納期とコストが超過した

事例における見切りの内容

ベンダー側は、発注側がリーダーシップをもって現行業務分析の実施や独自ミドルの詳細情報開示について実施してくれるだろうと思っていた

本来の見切りの考え方

・顧客側提供のソフトウェアを利用した開発の場合は、請負ではなく委任での契約が望ましいが、請負の場合には、開示条件、ドキュメントのレベル、開発におけるサポートレベルなどの条件をあらかじめ明確にして契約すべきである。また、業務分析においても、顧客側の体制や、役割、作業分担を調整しておく必要がある

捉えるべき兆候

・独自ミドルウェアの利用に関して、守秘義務の問題から、受注後の開示とされていた。受注後に発注側ミドルウェア担当者の説明を聞いたが、仕様書の説明を淡々と実施し、あまり踏み込んだ説明をしてくれない。情報システム部門が業務をよく把握しておらず、帳票一覧も持っていない

対処例

・顧客が参画する仕様確認会議の実施
・プロジェクト要員強化
・日次会議、および週次（休日）の進捗確認会議実施でプロジェクト管理を強化。全インターフェースの再精査、主要業務評価の前倒し実施
・専任要員（PMO）による調達費用精査と削減交渉実施
・経営層への報告実施

48 オフショア開発では、ドキュメントの記述は詳細に、プロジェクト管理はしっかりと

オフショアを活用したシステム開発。オフショアに委託する段階では、最近力を付けてきているものの、まだ設計できるほどのレベルにまで落ちていなかった。そのため、中国ソフトハウス側でも勝手な思い込みで作り込み続けた。プロジェクト側もスキル不足で、上がってくる中間成果物を十分に把握できず、システム規模が増え、作業遅れが発生した

事例における見切りの内容

顧客側の業務ノウハウについて、ある程度ヒアリングを実施。システムの大まかな姿を概要設計として示し、中国ソフトハウスに委託した

本来の見切りの考え方

・国内発注の場合は、概要レベルの記述でも、スキルがある会社であれば、問題はないが、オフショアの場合は難しく、より詳細なドキュメントにする必要がある。また、途中成果物を定期的にチェックし、問題がないか確認するなど、きめ細かいプロジェクト管理が必要である

捉えるべき兆候

・中国ソフトハウスに渡したドキュメントが概要レベルにとどまっている。プロジェクト経験の浅いメンバーばかりになっている

対処例

・体制の増強、管理体制の強化、再スケジュールの実施

49 顧客の体制が弱いときは、仕様追加、変更が多発する可能性が大

業務的にシステム化範囲が広いシステムの開発を受注。発注側の仕切り能力が弱く、業務内容の定義がきちんとできないことが懸念された。それぞれのイテレーションごとに追加要求や仕様変更が頻発し、規模がどんどん膨れた。進捗も遅れだし、品質にも問題が出始めた。規模の増大に伴ってテスト規模も大幅に増え、結果的に工数が大幅超過となってしまった

事例における見切りの内容

顧客要件をきちんと抽出することで構築後に大幅な手戻りをなくすことができるだろうと考え、4回のイテレーションを回すスパイラル型で開発を進めることにした

本来の見切りの考え方

・顧客の体制が弱い上に、業務内容も定義されていない状態では、顧客要件を抽出し、仕様を固めるのは非常に難しく、後で仕様追加、変更が多発する可能性が大きい。このような状態で、開発しながら仕様を決めていく開発手法は非常にリスクが高い

捉えるべき兆候

・プロジェクトの全体像が見えていない。スケジュールの関係から、仕様が十分に決まらない状態で、設計、製造に着手する

対処例

・体制の増強、管理体制の強化、リスケジュールの実施

50 規模が大きくなくても、協力会社に丸投げは危険

中規模のシステム開発。協力会社4社へ発注。その中の1社において、設計不良から、製造に大幅な手戻りが発生。要員を大量に投入してリカバリを図るが、品質問題が頻発。結果、プロジェクト全体が遅延し始めた。体制を強化し、マネジメントの強化と品質向上を図ってリカバリしたが、費用は大幅超過してしまった

事例における見切りの内容

予算の関係上、社員1名に協力会社4社を付けて対応することにした

本来の見切りの考え方

・中小規模の開発においても、社員1人で4社の外部委託管理を行うのは、外部委託先への丸投げと等しい
・問題を早期発見し、適切なタイミングで対処できず、問題が大きくなる可能性がある

捉えるべき兆候

・1人で多数の協力会社をマネジメントしている。管理体制が弱い

対処例

・体制の増強、管理体制の強化

51 信頼関係だけでは適切な課題管理はできない

SIベンダーが受注したシステム開発に際し、要件定義をSIベンダーが実施した後、基本設計以降の作業を実績がある協力会社に全面委託した。信頼関係を根拠に、設計内容の品質を進捗報告だけでフォローした結果、協力会社だけでは解決できない重大課題への対処が遅れ、上流工程の完了が大幅に遅延することとなった

事例における見切りの内容

要件確認をSIベンダーが行ったことで当該プロジェクトにかかわる主要な要件は見切れたとの認識から、基本設計以降の作業を協力会社に全面委託した。加えて、当該協力会社の実力をこれまでの実績から評価していたこともあり、作業の進捗報告だけで管理を行い、個々の課題管理などを共有しなかった

本来の見切りの考え方

- 協力会社に設計を委託する場合、設計品質の可視化と課題の達成状況の管理がSIベンダーにとって重要なタスクとなる
- また、新規業務の開発を委託した場合などは、協力会社が抽出する課題やWBSなどの網羅性を捕捉できるSIベンダーの体制作りが、プロジェクト・マネージャの重要なタスクであることを理解しておきたい

捉えるべき兆候

- SIベンダー側で、結合テスト・統合テストの仕様書が書けない
- 計画どおりに進捗している

対処例

- SIベンダーが求める設計水準や設計範囲などを協力会社と共有する目的で、当該SIベンダー側で設計書記載のガイドラインを事前に作成し、協力会社との間で十分なレビューを行っておく
- 進捗報告以外に、設計内容の段階的なレビュー会を適宜実施することで、設計内容の網羅性や品質についてタイムリーにフォローする
- 設計作業の過程で発生する課題のエスカレーションルールを協力会社との間で確立する
- 協力会社との契約状況再検証

52 インパクトが大きい要件への対処は先送りしない

要件定義完了段階で、RFPおよび提案書に記載しないような粒度の機能を新規要件として追加した際、当該システムの開発に不可欠な基本機能との認識から、コスト増に関する顧客側の承諾が得られなかった。交渉の長期化に伴う次工程への影響を配慮し、新規要件にかかわるコストをプロジェクトの持ち出しとせざるを得ない結果を招いた

事例における見切りの内容

提案時に機能単位の見積もりを要求されなかったこともあり、要件定義の段階でも、個々の機能に関するコストのインパクトや顧客側のコスト感に関する調整、確認を顧客との間で実施しないまま、要件定義の完了時点でコストの見積もりを一括で提出してしまった

本来の見切りの考え方

- 顧客側の予算枠やコスト増への手続きなどについては、可能な範囲で、営業を通じて事前に把握しておくことで、要件定義または基本設計段階で新規要件が発生する都度、顧客担当者と事前に相談すべき案件規模の見極めやタイミングをプロジェクト・マネージャ自身が判断できるようになる
- 顧客側では見えにくいシステム機能にかかわる要件については、要件定義や基本設計の内容をフィックスする以前に、提案内容との関連やコストへのインパクトについて顧客に説明をする機会を設けることで、見積もり範囲の妥当性に関する合意形成が容易になる

捉えるべき兆候

- 要件定義の内容には関心があるが、コストについて不安を感じていそうもない雰囲気
- 業務に関する要件定義には的確に応じてくれるが、システム面の見識には乏しそうな顧客側の開発体制

対処例

- 提案時に、顧客側から特段の要求がなくても、RFPに記載されている要件から推定される実装機能の種類や、規模とコストとの関連を見積もりの添付資料として顧客側に提出する
- 見積もりコストにインパクトのある新規要件が発生する都度、コストや納期への影響にフォーカスした意識合わせを、営業や顧客との間で実施する

53 フィット&ギャップ分析をせずにパッケージ導入の決断はしない

短期開発という事情もあり、業務要件とパッケージ仕様のフィット&ギャップ分析を実施せずに上流工程を完了したが、エンドユーザーによるテストで、パッケージソフトでは対応が困難な業務が存在することが判明し、一部の機能に関し要件定義のやり直しを迫られ、大幅な遅延とコスト負担をもたらす結果となった

事例における見切りの内容

提案時に既存システムの画面や帳票レイアウトを確認した範囲では、過去に発生した法制度や規則などの変更に伴う複雑な業務仕様を想定できないまま、開発コスト低減を目的にパッケージソフトの導入を提案した。顧客側が既存ベンダーに細かい業務仕様の検討を丸投げしていたこともあり、要件確認段階でシステム部門、ユーザー部門ともに“現行保証”以上の要件提示がなかったため、フィット&ギャップ分析を実施しないまま、パッケージソフトによる開発を決定してしまった

本来の見切りの考え方

- “現行保証”という業務要件だけで行うシステム開発は、顧客側でどこまで要件が見えているのか把握しづらい。上流工程の早い段階で、業務の複雑性や特異性を把握しているキーパーソン(多くの場合、既存ベンダーである可能性が高い)を見極め、業務要件にかかわる難易度をテーマにインタビューを重ねることで、パッケージソフト適用の是非を早期に見切ることが容易となる
- 上流工程では見えにくい仕様追加が発生する可能性がある場合は、導入するパッケージソフトのカスタマイズ機能や開発体制などを事前に詳細を押さえておくことで、問題発生時の打ち手を幅広に持っておくことができる

捉えるべき兆候

- 顧客側のユーザー部門と打ち合わせをしても、過去の苦労話が一切出てこない。
- 要件確認を行う際、システム部門との関係が悪そうにみえないのに、なぜかユーザー部門の参加率が悪い(業務を担当するユーザー部門も、要件定義で話せる内容が少ない)

対処例

- 初めて扱う業務開発に関しては、過去の開発経緯などを把握している既存ベンダーを開発体制に組み入れるように、営業などを通じて早期に調整する
- 要件定義の早い段階で、受注側が把握している業務要件に関して顧客側のユーザー部門を含めたレビューを実施し、パッケージソフトを提案した妥当性について、顧客側の理解を求めておく必要がある
- パッケージソフトのサポート範囲(パッケージソフトでできること)は、上流工程の早い段階で顧客側に説明を行い、想定外の仕様追加への対応策について、コスト負担を含め、顧客側の了解を取り付けておく
- 経営、営業部門含めた課題管理の実施

54 過去データ移行の難易度はプロジェクト・コストにインパクトあり

他社リプレースに際し、過去データの量的規模を見極めたうえでデータ移行にかかわる見積もりを行った。だが、過去に幾度となく実施された制度変更を踏まえたデータの連続性が、帳票作成やデータ検索処理に求められることを把握し切れなかった。請負契約のなか、要件実現に必要な移行ツールやデータ変換機能などの開発コストを受注側で捻出する結果となった

事例における見切りの内容

過去データのなかにはオペレーションミスなどによる不正なデータが一部含まれている可能性については覚悟していたが、過去の制度変更に伴うデータ移行や業務処理などへの配慮といった特殊な要件が見えないまま、設計工程を完了してしまった

本来の見切りの考え方

- 過去データの取り扱いは、顧客側にも見えにくい要件の一つである
- また、制度変更時にビジネスロジックで対応することで、実際のデータには手を付けなかったケースも意外と多い。データ移行に関する具体的な要件を把握しづらい場合は、データ移行の特殊性を意識したうえで、既存ベンダーへのインタビューや既存のプログラム仕様書の確認などを、早い段階で実施することが望ましい

捉えるべき兆候

- 法改正や制度変更などの頻度
- 対象業務の特殊性

対処例

- 過去に発生した制度変更や業務仕様の変更がデータに与える影響調査を上流工程の段階で確実にフォローする
- 上流工程では、既存ベンダーを含む業務仕様に精通したメンバーの関与が不可欠である旨、受注の条件やプロジェクトの立上げ段階で明確に顧客側に伝える
- 経営、営業部門含めた課題管理の実施

55 中核と思われる機能以外に相応の開発規模が隠れていることも

他社パッケージ製品を採用したシステムのリプレースに際し、基本設計で入出力項目を調査したところ、パッケージを改造したり個別プログラムで補完するなど、パッケージ製品だけでは実現できない業務要件が存在することが判明した。しかしながら、商談の性格上、大幅なコスト増を顧客と折衝することもできず、相当部分を自社調達することになった

事例における見切りの内容

重点顧客に初めて自社製品を導入できる絶好の機会であったこともあり、業務要件よりもパッケージの機能比較に着目した提案を行ってしまった。短期開発であったこともあり、他社パッケージ製品に基づくスクリプトなどを解析する手間をかけるわけにもいかず、要件確認フェーズをクローズしてしまった

本来の見切りの考え方

- メジャーなパッケージ製品に限って一般に公開されている情報量も多いこともあり、パッケージ製品の機能比較に偏った提案および要件確認を行ってしまう傾向が強い
- パッケージ商談では、中核となる業務要件だけではなく、画面や帳票出力といった外部仕様へのこだわり度合いや、制度変更などに伴う処理の複雑さなどに関してもアンテナを高くしながら、上流工程全般をフォローするスタンスが重要となる

捉えるべき兆候

- 要件定義の初期段階から、顧客側が細かい画面仕様への拘りを示すようになった
- 対象業務の特殊性

対処例

- 状況が許すのであれば、上流工程の早い段階でエンドユーザーへのインタビューを行い、仕様変更の頻度や要求仕様へのこだわり度合いなどへの把握に努める
- 状況が許すのであれば、既存ベンダーへの事前ヒアリングやこれまでにかかったコスト（概要の工数でも可）の開示などを顧客側に依頼してみる
- 既存システムのパッケージに関する技術仕様に詳しいメンバーをプロジェクトに投入する

56 ドキュメントの成果レベルは事前調整が不可欠

他社リプレース開発において、上流工程で実施する要件定義や基本設計などの成果イメージを顧客と合意していなかった。このため、設計過程で記載レベルや設計範囲などの調整に時間を要した結果、上流工程の終了が遅延し、コスト増加と後続の開発スケジュールに影響を与えた

事例における見切りの内容

初めて当該顧客の開発を請け負うこととなったため、ドキュメントの品質をはじめ、成果物の出来栄えまで事前にすり合わせができるような状況ではなかった。要件定義書や設計内容に関しては、記述内容の粒度や範囲について、どの顧客もあまり大きな差がないだろうという経験的判断が働き、成果物イメージのすり合わせを事前に行わなかった。要件そのものの追加や変更が発生することは想定していたが、要件定義書に記載する内容のレベル感が顧客と合っていないとは想定していなかった

本来の見切りの考え方

- それぞれの顧客にとって、上流工程で作成するドキュメントの位置付けが様々であることを理解することが重要である。顧客によっては、開発の標準化による手続きなどの都合で、他の顧客でいう詳細設計の一部を基本設計で求めるケースも少なくない
- 特に新規顧客の場合は、顧客側で策定している開発プロセスに関する考え方や手続きなどを把握することを最初のタスクとしてもよいほどである

捉えるべき兆候

- 計画したとおりに打ち合わせが進行しない
- 受注者側では想定していなかったスケジュール遅延を顧客側から言い出し始めた

対処例

- プロジェクトのキックオフミーティングなどの機会を捉えて、顧客側が想定している開発プロセスや手続きに関するレビューを依頼する
- 顧客側に既存システムのドキュメントについて開示を要求するか、ベンダー側で成果物のイメージを共有できるサンプルの提示をもって、可能な限り、早期の意識合わせを行う
- 経営、営業部門含めた課題管理の実施

57 RFPとの差異が明確になっているか

RFPに記載されていた機能の実現方法では業務要件を達成できないと判断し、RFPとの差異を明確にしないまま提案を行った結果、要件定義フェーズで顧客側との検討が噛み合わず、上流工程の完了が大幅に遅延したばかりか、改善提案部分のコストも持ち出す結果となってしまった

事例における見切りの内容

他社競合の商談であったこともあり、自社のソリューション力をPRすべく、一部の内容についてRFPが求める実現方式と異なる提案を行った。要件定義のベースが提案書記載の内容であるという受注側の思いを先行させたまま上流工程を進めたため、設計の後半まで、顧客との意識のずれを把握し切れなかった

本来の見切りの考え方

- RFPの記載内容に対する改善提案をしたつもりでも、PowerPointによる概念的な表現が主流になっている昨今、そういった意図がストレートには伝わりにくい傾向がある。加えて、顧客側はRFPに準拠した提案であるため、潜在的な期待感もあり、相違点についてはより明確に記載するよう心掛ける必要がある。また、要件定義のベースがRFPなのか提案書なのか、といった基本的なことについても、顧客と早い段階で合意を得ておくことも大切
- 改善提案のなかには、システムの中核機能や顧客の現状に影響を与えかねない内容が含まれていることが多いので、プロジェクトの円滑な推進のため、早期にリスクとなり得る芽を摘んでおくことを心掛けておきたい

捉えるべき兆候

- 仕様検討の過程で顧客の反応が鈍ってきた
- 何度打ち合わせを重ねても顧客との合意を得られなくなってきた

対処例

- RFPと異なる要件に関しては、顧客側との意識合わせを行う主旨からも、要件定義フェーズの前半に集中討議を実施するなどして、プロジェクトのリスクとなっていないことを確認する機会を設ける

58 パッケージ開発では業務要件とのギャップが重要な見極めのポイント

パッケージ製品による開発に際し、プライマリーベンダーを支援する形態でプロジェクトに参画したが、開発体制にパッケージ製品に関するエキスパートを組み込めないまま上流工程を進めたため、パッケージ機能と業務要件とのギャップが製造途上で判明し、方式全体の見直しを余儀なくされ、設計及び製造に大幅な手戻りが発生した

事例における見切りの内容

＜プライマリーベンダーの見切り＞
顧客のビル移転との関係で稼働時期が決められていたこともあり、パッケージ製品の適用による短期開発こそが最善の開発方式であると思い込んでいた。フィット＆ギャップ分析を意識はしていたものの、顧客側の仕様決定が度々滞る事態も発生したため、パッケージ機能を活用することで、顧客の要求仕様は概ね実現できるという直感的な見切りを行うことで、上流工程を完了させてしまった
＜サブベンダーの見切り＞
採用しているパッケージ製品の製造元がプライマリーベンダーであったこともあり、プロジェクト推進過程で発生した問題解決は迅速に行えるはずという判断から、上流工程におけるエキスパート投入を重視しなかった

本来の見切りの考え方

- パッケージ製品の機能に依存したシステム開発を行う場合は、製造工程を圧縮してでも、上流工程で業務要件とのフィット＆ギャップ分析を適切に実施すべきである。分析結果という明確な証跡をもって上流工程を完了することで、例え上流工程では見えない業務要件などが潜在していたとしても、中・下流工程で方式全体の見直しを行う場面で、コストや納期を含めた顧客やプライマリーベンダーとの折衝が容易になるものである
- 自社、他社を問わず、パッケージ製品にかかわるエキスパートの投入や迅速なQ/A対応を実現できる体制作りは、パッケージ機能に依存したシステム開発の場合、避けて通ることができない重要な準備タスクである
- 自社製品とはいえ、企業の合併、吸収が頻繁なIT業界では、必ずしも製品のサポート体制が国内になるとは限らない。製品のサポートレベルについては、具体的な水準や実現時期を明確にしたうえで、プロジェクトへの参入時点で、プライマリーベンダーと体制に関する調整を済ませておくことが重要である

捉えるべき兆候

- 機能設計やプログラム設計書の記載内容に曖昧さが残る
- パッケージ機能と業務要件とのマッピングに予想以上の時間を要し始めた

対処例

- 業務要件とのギャップを埋めるノウハウを有する技術者を最優先で確保するようプライマリー・ベンダーに要請する
- 上流工程で決定した設計内容の実現の可否を評価できるエキスパートを開発体制に組み入れることで、製造工程以降の手戻りを最小化するよう、プライマリーベンダーに要請する
- プライマリー・ベンダーとの契約（分担、責任など）を明確にしておく

付録 4.リスク分類表

項番	知識エリア	リスク分類	ヒアリングシート	測定分析データー覧	事例集
1	顧客	顧客予算	H6,H42,H47		6,13,24
2		顧客特性	H2,H3		4,20,24,27,32,38,39,46,47
3		案件特性	H32,H70		23,26,27,35,36,38,39,46,47,56
4		顧客役割分担	H2	Ko1,Ko2,Ko3	2,4,5,20,32,34,46
5	スコープ	開発範囲の取り決め	H32,H34,H52		5,6,23,25,27,41,42,43,44,45,46
6		要求と要件の整合性		S1,S2,S3	3,43,45,52,53,57
7		要件とシステム化範囲の整合性	H1,xH4,H5,H6,H33,H34"	S5,S6,S7,S8,Q10,Q11,Q14,Q15,Q16	16,17,32,36,43,45,49,52,53,56,57
8		システム化範囲と体制の整合性	H3,H4,H16		2,32
9		システム化の実現可能性	H5,H6,H7,H25,H35,H72	Q14,Q15,Q16	26,31,33,53
10		変更対応	H32,H33,H42,H47	S4,S9	14,15,22,27,43,45,51,55
11	タイム	クリティカルパスの特定	H5,H22	T15	3,12,31,32,44,49
12		実行可能スケジュールの作成	H4,H16,H18,H19,H21,H23,H31	T16	3,5,10,17,18,26,32,44
13		進捗確認	H27,H68	Q10,Q11,Q14,Q15,Q16,T3,T4,T7,T8,T9,T10	49,50,51
14		作業ボリューム推移	H7,H18,H21	T1,T2,T5,T6,Q16	31,49
15		タスクの分解	H16,H19,H31	Q10,Q14	32,46,49
16	コスト	上流工程の見積り妥当性	H46	C1	7,26,43,44,45,46,56
17		基本設計以降の見積り妥当性	H7,H46,H47,H48	Q10,Q11,Q14,Q15,Q16	26,43,44,45,49
18	人的資源	業務有識者	H28	H2,H6	11,21,43,45,48,54
19		技術系専門家	H29	H1,H5,H7	11,26,28,34,35,58
20		PJ内部体制	H13,H14,H15,H17,H30,H66,H69,H70,H71	H3,H4,H8,H9,H10,Ko1,Ko2,Ko3	10,25,26,28,29,31,48
21	調達	外部調達の妥当性	H66,H67		10,30,46,48
22		パッケージ適用	H32,H73		40,43,45,53,55,58
23	組織	PJ支援体制	H14		26,29,50
24		組織の効率		O6	
25	コミュニケーション	PJ内情報共有	H1,H8,H9,H11,H20,H60	Cm2,Cm3,Cm4	25,51
26		PJ外情報共有	H2,H10,H12,H20,H24,H38,H43,H44,H53	Cm1	17,38,43,44,45,46

項番	知識エリア	リスク分類	ヒアリングシート	測定分析データ一覧	事例集
27	品質	品質計画	H26,H49,H50,H51,H68,H73	Q10,Q11,Q14,Q15,Q16,Q17,Q18,Q19	
28		品質管理	H9,H40,H41,H49,H51,H56,H68,H72,H73	T11,T12,T13,T14,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11,Q14,Q15,Q16,Q17,Q18,Q19,O1,O2,O5	30,50
29		品質保証		Q1,Q12,Q13,Te1	
30	技術	未経験技術	H74	Te2	14,26,28,31,33,48
31		標準化検討	H26,H40,H41,H52	Q13	12
32		システム実現方式検討	H25,H35,H54,H56,H57	Te1	26,31,33,35,46
33		移行・パッケージ	H55,H73,H74		30,33,34,37,39,47,54,55,58
34	基本動作	作業標準	H40,H58,H59	O1,O2,O5,O7,O8	1,8,9,24,25,43,45
35		セキュリティ順守	H39		
36	モチベーション	人材活用	H61,H62	M1	
37		当事者意識	H15,H60,H62,H63,H64,H65	M2	
38	リスク	リスク識別	H36	R1,R2	31,43,45
39		リスク分析・評価	H37	R1,R2,R3	26,27,45
40		リスク対応策	H38		27
41	課題	課題認識	H43,H44	K1,K2,K3	31
42		課題解決	H45	K1,K2	

参考文献

［1］ Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Charles V. Weber, “Capability Maturity Model for Software, Version 1.1”, Software Engineering Institute, CMU/SEI-93-TR-24, ESC-TR-93-177, February, 1993.

［2］ CMMI Product Team, “Capability Maturity Model Integration for System Engineering /Software Engineering/Integrated Product and Process Development, Version 1.1”, Software Engineering Institute, CMU/SEI-2002-TR-004, August, 2002.

［3］ Victor R. Basili, David M. Weiss, “A Methodology for Collecting Valid Software Engineering Data”, IEEE Transaction on Software Engineering, pp.728-738, November, 1984.

［4］ John McGarry, David Card, et al., “Practical Software Measurement, Joint Logistics Commanders and Office of the Undersecretary of Defense for Acquisition”, 1997.

［5］ John McGarry, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Joseph Dean, Fred Hall, “Practical Software Measurement: Objective Information for Decision Makers”, Addison-Wesley Pub., October, 2001.

［6］ ISO/IEC 15504-1:2004, “Information technology——Process assessment——Part 1: Concepts and vocabulary”, ISO, November, 2004.

［7］ ISO/IEC 15504-2:2003, “Information technology——Process assessment——Part 2:Performing an assessment”, ISO, October, 2003.

［8］ ISO/IEC 15504-3:2004, “Information technology——Process assessment ——Part 3: Guidance on performing an assessment”, ISO, January, 2004.

［9］ ISO/IEC 15504-4:2004, “Information technology——Process assessment——Part 4: Guidance on use for process improvement and process capability determination”, ISO, July, 2004.

［10］ ISO/IEC TR 15504-5:1999, “Information technology——Software Process Assessment——Part 5: An assessment model and indicator guidance”, ISO, June, 2002.

［11］ ISO/IEC TR 9126-1:2001, “Software engineering——Product quality——Part 1: Quality model”, ISO, June, 2001.

［12］ ISO/IEC TR 9126-2:2003, “Software engineering——Product quality——Part 2: External metrics”, ISO, July, 2003.

［13］ ISO/IEC TR 9126-3:2003, “Software engineering——Product quality——Part 3: Internal metrics”, ISO, July, 2003.

［14］ ISO/IEC TR 9126-4:2004, “Software engineering——Product quality——Part 4: Quality in use metrics”, ISO, March, 2004.

［15］ ISO/IEC 15939:2002, “Software engineering——Software measurement process”, 2002.

［16］ JIS X 0129-1:2003, “ソフトウェア製品の品質——第1部：品質モデル”, 日本規格協会, February, 2003.

［17］ JIS X 0141:2004, “ソフトウェア測定プロセス”, 日本規格協会, June, 2004.

［18］ 村上 弘, 飯田 元, 松本 健一, “ソフトウェア開発プロセス管理データの収集と活用の支援を目的

とした電子ガイドの提案”, 電子情報通信学会技術報告, no.SS2004-41, pp.43-48, November, 2004.

［19］Ulrike Becker-Kornstaedt, Roman Reinert, “A concept to support process model maintenance through systematic experience capture”, In Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering（Ischia, Italy, July 15 -19, 2002）. SEKE '02, vol. 27. ACM Press, New York, NY, pp.465-468.

［20］引地 一将, 飯田 元, 松本 健一, “ソフトウェア開発における定量的管理計画の立案支援”, 信学技報 Vol.105 No.491, pp.55-60, December, 2005.

［21］田中 康, 飯田 元, 松本 健一, “成果物間に関連した着目した開発プロセスモデル：PReP”, 情報処理学会論文誌, Vol.46 no.5, pp.1233-1245, May, 2005.

［22］井上 克郎，松本 健一, 飯田 元, “ソフトウェアテクノロジーシリーズ8 - ソフトウェアプロセス”，共立出版，2000.

［23］井上 克郎，松本 健一, 鶴保 征城, 鳥居 宏次, “実証的ソフトウェア工学環境への取り組み”, 情報処理7月号, vol.45, No.7, pp.722-728, 2004.

［24］飯塚 悦功編, “ソフトウェアの品質保証ISO-9003 対訳と解説”, 日本規格協会，1992.

［25］N. E. Fenton, “Software Metrics: A Rigorous　Approach”, Chanpman & Hall，1991.

［26］M. C. Paulk，et al., “The Capability Maturity Model”，Guidelines for Improving the Software Process，Addison Wesley Publishing CO.，Inc．1995.

［27］Hansen，G. A. ，“Automating Business Process Reengineering Ecglewood Cliffs”，NJ: Prentice Hall，1997.

［28］W. S. Humphrey，“Managing the Software Process”，Addion-Wesley Publishing CO.，1989.

［29］P. Oman and S. L. Pfleeger，“Applying Software Metrics，IEEE Computer Society Press”，1997.

［30］M. Lorenz and J．Kidd，“Object-Oriented Software Metrics-A Practical Guide”，PTR Prentice Hall，Inc 1994.

［31］S. D. Conte，H. E. Dunsmore and V. Y. Shen，“Software Engineering Metrics and Models”，The Benjamin/Cummings Pub.，1986.

［32］J. R. Dunham and E．Krusei，“The Measurement task area”，IEEE Computer，vol16，No.11，pp.47-54，1983.

［33］山田 茂，高橋 宗雄，“ソフトウェアマネジメントモデル入門”，共立出版，pp.20-23，1993.

［34］T. DeMarco，“Controling Software Projects : Management，Measurement & Estimation”，Yourdon Press，1982.

［35］Y. S. Sherif，Ng，E. and J. Steinbacher，“Computer software development : Quality attributes，measurements，and metrics”，Naval Reserch Logistics，Vol.35，No.3，pp. 425-436，1998.

［36］Y. S. Sherif，Ng，E. and J. Steinbacher，“Computer software quality measurements and metrics”，Microelectronics and Reliability，Vol.25，No.6，pp.1105-1150，1985.

［37］T. Glib，“Software Metrics”，Winthrop，1976.
［38］C. Jones，“Applied Software Measurement - Assuring Productivity and Quality，2nd Edition”，McGraw-Hill，1997.
［39］T. J. MaCabe，“A Complexity Measure”，IEEE trans.，Software Eng.，Vol2，No.4，pp.308-320，1976.
［40］T. K. Abbel-Hamid，S. Madnick，“Software Project Dynamics，An Integrated approach”，Prentice-Hall，1991.
［41］独立行政法人情報処理推進機構,ソフトウェア・エンジニアリング・センター，“ITプロジェクトの「見える化」下流工程編”，日経BP，2006.

執筆者（敬称略・五十音順）

プロジェクト見える化部会委員（◎：主査、○：副主査）

○	秋山 雅俊	株式会社NTTデータユニバーシティ
	飯田 元	国立大学法人奈良先端科学技術大学院大学
	大川 繁喜	日本電気株式会社
	亀田 康雄	NTTソフトウェア株式会社
	木根 秀隆	日立ソフトウェアエンジニアリング株式会社
	栗田 存	株式会社クロスリンク・コンサルティング
	桑原 秀昌	TIS株式会社
	香村 求	株式会社システムSWAT
	芝田 晃	三菱電機株式会社
	清水 友彦	株式会社東京証券取引所
○	西川 広	新日鉄ソリューションズ株式会社
	拝原 正人	株式会社クロスリンク・コンサルティング
	福地 豊	株式会社日立製作所
	水野 真澄	株式会社日立システムアンドサービス
	長岡 満夫	ソフトウェア・エンジニアリング・センター（株式会社NTTデータ）
◎	長岡 良蔵	ソフトウェア・エンジニアリング・センター
	樋口 登	ソフトウェア・エンジニアリング・センター（日本電気株式会社）
	神谷 芳樹	ソフトウェア・エンジニアリング・センター
	安田 守	株式会社野村総合研究所

研究解説執筆、編集等協力

遠藤 和博	株式会社野村総合研究所
小久保 岩生	株式会社三菱総合研究所
新田 一樹	株式会社野村総合研究所
浜石 晃	株式会社三菱総合研究所

ITプロジェクトの「見える化」 上流工程編

2007年5月1日	初版第1刷発行
著作／監修	独立行政法人 情報処理推進機構（IPA） ソフトウェア・エンジニアリング・センター（SEC）
発行人	浅見 直樹
発行	日経BP社
発売	日経BP出版センター 〒108-8646　東京都港区白金1-17-3
表紙デザイン	成田 美由喜（日経BPクリエーティブ）
制作	日経BPクリエーティブ
印刷・製本	大日本印刷

ISBN978-4-8222-6209-9

日経BP社
Nikkei Business Publications, Inc.
〒108-8646　東京都港区白金1-17-3