実用プロマネ

わかる現場の基礎知識

はじめに

本書を手に取っていただき有難うございます。

本書は、ソフトウェア開発プロジェクトにおける新人プロジェクトマネージャ（以降プロマネと表記）[1]ないしはこれからプロマネになろうとする人、およびプロジェクトマネジメント[2]に興味のある方々を対象に書かれたもので、プロマネの仕事の基本的な骨格をできるだけ分かりやすく解説しようと試みたプロマネの入門書です。

対象とする読者は、社会人・学生の別、および発注者側・受託者側であるかどうかも問いません。しかしながら実務者としてプロマネの仕事を実際に遂行するためには、ソフトウェア設計やプログラミング等のソフトウェア開発者としての経験が必要となります。ただし設計やプログラミングに関する実務経験がない場合でも、自学自習によってその内容を理解している人であればプロマネ業務の遂行が不可能というわけではありません。 本書はまた、中堅以上のプロマネの方々および開発リーダー職の方々においても、初心を振り返るためにもお役に立てるものと思います。

また本書が対象とするソフトウェア開発プロジェクトの規模は、開発費が１００人月以下の中小規模のプロジェクト[3]における派生開発[4]を想定しています。中小規模のプロジェクトにおけるプロマネの業務は、開発リーダーおよびサブリーダー等が担うことになると思われ、プロマネの実際の守備範囲はプロジェクトマネジメントの領域のみならず開発マネジメントの領域までも含んだものになります。従って本書におけるプロマネの役割はＰＭＢＯＫ等で示されている知識領域やプロセス[5]領域だけではなく、開発リーダーとしての役割までも包含したものになっています。 なお本書はＰＭＢＯＫ[6]準拠の解説書ではなく、ＰＭＢＯＫの知識を前提とはしていません。

本書のゴールは、本書をしっかり読めば、プロジェクトマネジメントの一通りの知識や勘所がなんとなく理解できるようになる所に置いており、体系的かつ本質を押さえた内容で、広範囲な領域を網羅した構成となっています。 本書の特徴は下記の三点にフォーカスされています。

- **分かりやすい**
- **実行しやすい**
- **時間がかからない**

---

[1] **プロマネ（プロジェクトマネージャ）** プロジェクトマネジメントの手法を用いてプロジェクトを運営する責任者のこと。

[2] **プロジェクトマネジメント** 獲得したリソース（人・モノ・カネ・情報・時間）の制限内で、プロジェクトのＱＣＤ目標をすべて達成し、プロジェクトを成功に導くこと。

[3] **プロジェクト** 特別な目的を達成するための期間限定の臨時的な組織のこと。

[4] **派生開発** 新規開発後の、いわゆるバージョンアップ開発のこと。

[5] **プロセス** ある目的に従った結果を得るために、入力情報を一定の論理に従って加工する処理手順のこと。

[6] **ＰＭＢＯＫ**（Project Management Body of Knowledge）プロジェクトのマネジメントに必要なガイド、手法、メソドロジー、ベストプラクティスをまとめた、国際的に標準とされているプロジェクトマネジメントの知識体系のこと。

１．分かりやすい

難解な説明を避け、本質を分かりやすい言葉で表現し、開発の一連のプロセスの流れに沿って現場における主要な問題点に焦点を当てました。

プロセスの流れは下記の６つのプロセスで構成されています。

準備

① 第２章 事前準備のプロセス（Preparation）

：顧客・ステークホルダー情報の収集

計画

② 第３章 計画のプロセス（Plan）

：要件定義・見積り・スケジュール作成・目標値の設定・開発体制の構築などの開発の超上流工程における活動

実行

③ 第４章 実行のプロセス（Do）

：リスク解消・データドリブン開発・ドキュメントベース開発・優先順位・改善活動などの開発の中核工程における活動

管理

④ 第５章 管理のプロセス（Monitoring & Controlling）

：要求仕様の変更管理・品質管理・コスト予算管理・進捗管理などの動的に変化する仕様・ＱＣＤ指標の監視・管理

振返

⑤ 第６章 振り返りのプロセス（Retrospect）

：ラップアップ・レビューを中心とした開発実行結果の検証、失敗の真因特定、再発防止策の策定などの活動

継承

⑥ 第７章 ノウハウ継承のプロセス（Transfer）

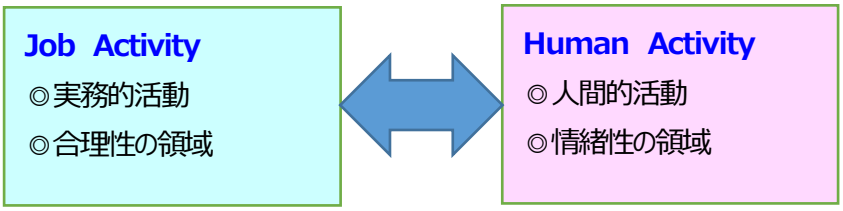
：プロジェクト活動において蓄積されたノウハウ等の成果の継承

さらに各プロセス章の中は、基本的に【Job Activity（実務的活動）[7]】と【Human Activity（人間的活動）[8]】の二つの活動領域で構成されています。【Job Activity】においては技術的な実務活動について、【Human Activity】においては人間力発揮の活動について記述されており、実務性（合理性）と人間性（情緒性）の両面からのアプローチ[9]によって、各プロセス活動の実効性を確かなものとしています。

開発者も感情を持つ人間であり、合理性に基づいた実務行動指針を与えられたとしても、情緒性における納得が得られなければそれを実行することは困難でしょう。実務性（合理性）を確かなものにするためには人間性（情緒性）における思考や行動の指針が必要となります。

なお各章におけるポイントは、プロセスマーク付きテキストボックス内において一言でまとめ、専門用語等は、ページ末の脚注において簡単な説明を加えました。

【２つの領域で構成されているプロマネ活動】（図はじめに－２）



[7] **Job Activity**（実務的活動） 実務性（合理性）を必要とする技術的な実務活動のこと。

[8] **Human Activity**（人間的活動） 人間性（情緒性）に関する人間力発揮の活動のこと。

[9] **アプローチ** 問題などへの取り組み方や方法のこと。

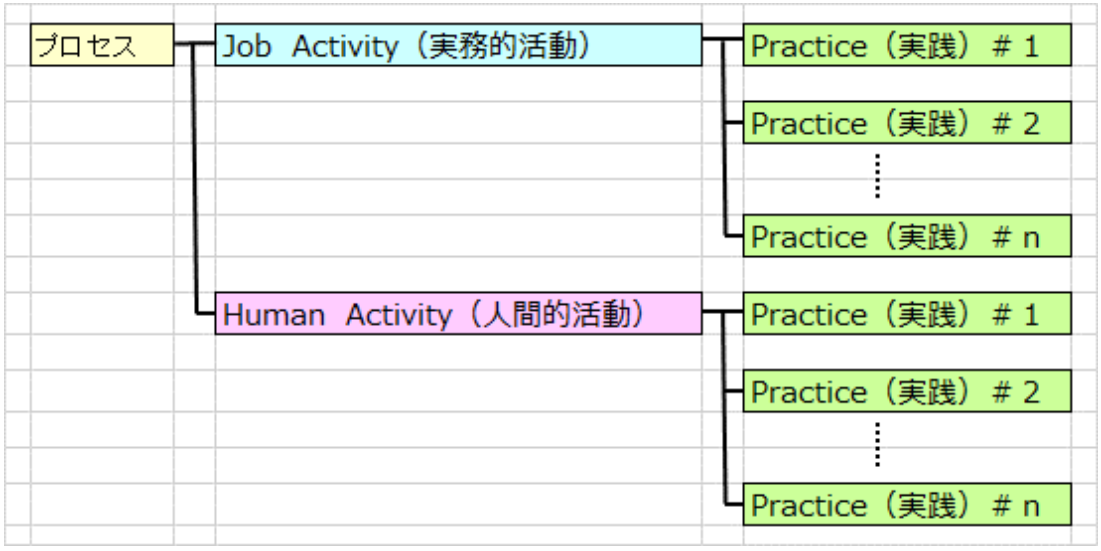
２．実行しやすい

各プロセスにおける Job Activity（実務的活動）と Human Activity（人間的活動）に属する practice（実践）は、ＰＭＢＯＫの１０の知識エリアおよび５つのプロセスを網羅し、尊重しつつも、中小規模のプロジェクトにおける現場の知見を通した実効性に価値を置き、実地に基づいた平易かつ実行しやすいものを目指しました。

従って管理帳票等は複雑かつ専門的な理解を要するものではなく、各プロジェクトにおいて自前で作成可能なＥＸＣＥＬベースのものを採用しました。

プロセス／アクティビティ[10]／プラクティス[11]の構成は次のようになっています。

【プロセス／アクティビティ／プラクティスの構成図】（図 はじめに－３）



３．時間がかからない

読者の理解を短時間で可能にするために本質を押さえたコンパクトな構成とし、豊富な図・表による視覚的な認知を容易にし、短時間で読めるようになっています。

本文は、第１章のプロジェクト・マネジメントは何かに始まり、実際の開発プロセスの時系列順に、事前準備、計画、実行、管理、振り返り、で構成されており、最後に循環するプロセスを意識したノウハウ[12]継承のプロセスで締めくくっています。

---

10 **アクティビティ** 業務遂行に必要な活動のこと。

11 **プラクティス** 業務活動（アクティビティ）を構成する具体的な実践内容のこと。

12 **ノウハウ** 業務の遂行において必要な知識や経験などの中で、とりわけ重要なものの総称。

４．チェックリスト[13]の役割

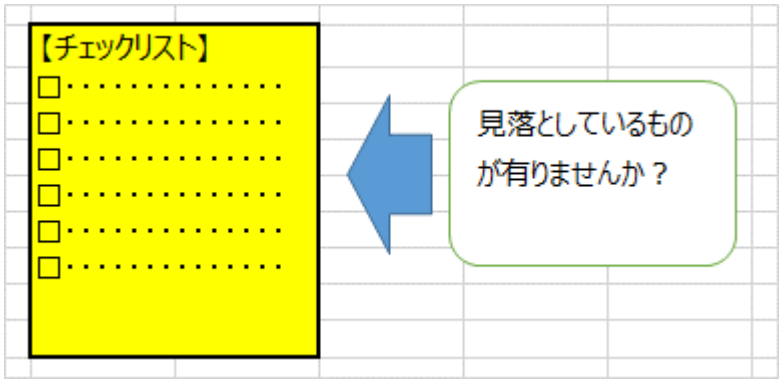
本書おいては、さまざまな局面における主なリスク[14]をチェックリスト形式で記載していますが、読者の誤解を避けるためにチェックリストの本来の役割について説明をしておきたいと思います。

チェックリストとは、過去の失敗事例をその原因ごとにパターン化してリストアップしたものです。本書において提示されたチェックリストは、致命的な手順抜けの予防を目的としたもので、問題の解決法を示したガイドブックを意図してはいません。

例えば３－１．の【早期仕様凍結のチェックポイント】（ｐ４７）における「□　早期仕様凍結のために、顧客および関連各社の参加・協力の要請を行うこと」に対する解決策は、プロジェクトを取り巻く環境によって様々であり、一つの解決策を本書で提示したとしても、余り意味を持つものにはならず、チェックリスト自体が膨大な量になりチェックリストの役割を果たせないものになってしまいます。

そういうわけで本書において提供されたチェックリストは、リスクの見落としを防ぐ役割は果たすもののリスクそのものを解消する直接的な機能は持っていません。具体的なリスク解消策は、読者およびその組織の経験則の中にあり、具体的な対策はその時々の事情で変化するため、各読者においてはその時々の状況にマッチした解決策を考え、それをリスク管理表に記載し、リスク回避を実行していただきたいと思います。

【見落としを防ぐチェックリスト】（図　はじめに－４）



参照：ｐ２６６　チェックリスト一覧表

[13] **チェックリスト**　致命的な手順抜けの防止を目的として、失敗事例を手短にまとめたリスト。

[14] **リスク**　現時点では問題として現れてはいないが問題として現れる可能性のあるもので、プロジェクトを失敗させるすべての要因のこと。

５．本書の文章構成および使用マークの意味

本書は、読みやすさや分かりやすさを実現するために、以下のようなスタイルを採用しました。

文章の構成

- 文章の構成は、複雑さを避けるために、章および節の２階層構成としました。
- 各章の開始ページには、その章の趣旨の概要を記述し、理解の準備としました。
- 各章の最後は、その章の骨子をまとめ、理解の再確認としました。
- 頻出用語ないしは、なじみの薄い用語に関しては、各ページの下に脚注として用語説明を加え、検索の利便性を考慮し、巻末に用語集としてまとめて記載しました。
- 本文中に使用した図・表の全体像を表示したものを、巻末に付録図表として記載しました。
- 本文中に使用したチェックリストの一覧表を、巻末に記載しました。

本文中のマークの意味

プロマネ

- **ロゴマーク：**各章を明示するためのロゴを、各章のすべてのページの右肩に配置しました。

顧客情報の収集

- **テーママーク：**節の中で重要なテーマとなるものについては、テーママークにてテーマ名を枠囲み文字で表記し、テーママークの背景色は、各章特有の背景色としました。

- **トピックスマーク**　：注目すべき話題の印。

◎

- **項目マーク**　：トピックスマーク下の説明項目の表題。

☆

- **注目マーク**　：項目マーク下の注目記事。

- **ポイントマーク**　：各章内における説明のまとめ文の印。

◆

- **チェックリストマーク**　：チェックリストの印。

上記の囲み文字入りのマークは、一例を示したもので、各章において文字および背景色は異なります。

目次

第４章　実行のプロセス　ｐ１０３

【Job Activity】

第７章　ノウハウ継承のプロセス【統合マネジメント】ｐ２１５

【Job&Human Activity】【統合マネジメント】

プロマネ

第１章

プロジェクト

マネジメントとは何か

【統合マネジメント】

概要

第１章　プロジェクトマネジメントとは何か

プロジェクトを成功に導くための基本的な仕組みについての解説です。

- プロジェクトは、特別な目的達成のための臨時的な組織（１－１．）
- プロジェクトマネジメントとは、プロジェクトを成功に導くこと（１－１．）
- プロジェクトの成功要因は、ＱＣＤの三つ（１－１．）
- プロマネの基本的な業務は、見積り・早期仕様凍結など１１項目がある（１－２．）
- プロマネは、仕様の全体像の把握と実行方法の理解が必要（１－３．）
- プロジェクトマネジメントには、さまざまなリスクがある（１－３．）
- 開発プロセスは、INPUT－PROCESS－OUTPUT で構成される（１－４．）

プロジェクトマネジメントの具体的な解説に入る前に、そもそもプロジェクトとは何か、マネジメントするとはどういうことか、について筆者の経験に基づく理解を述べ、さらにプロジェクトマネジメントという仕事の全体像を把握しておくために、プロジェクトの成功要件、プロマネの役割、主な問題とリスク、およびプロセスということについての理解を読者の方々と共有しておきたいと思います。

1－1 プロジェクトについて【統合マネジメント】

プロジェクトとは何か

プロジェクトの元々の意味は「何かを考え出す」ということで、今までなかった何かの役に立つものを作り出すことを指しています。私たちのソフトウェア開発におけるプロジェクトとは、ある期間内である特別の目的を達成するために、その実行目標を設定し、それを実現するために、ソフトウェア技術を主な実現手段として使い、必要なリソース[15]（人材、資材、資金、情報など）を一時的に集めた組織のことだと言えます。

例えば製品開発組織および製造部門をもっている工場の場合、新製品を開発している部門の各チームが採用している組織形態がプロジェクトの代表的なものです。私たちのソフトウェア開発組織もプロジェクト制によって運営されており、開発と名のつく組織はすべてプロジェクト制によって運営されています。

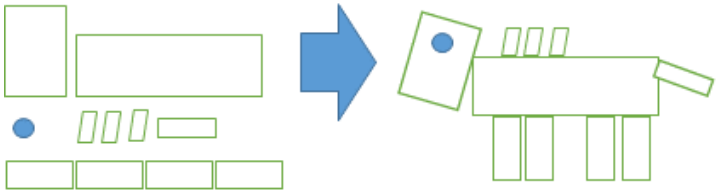
一方、新製品を世の中に商品として送り出すための開発以外の組織として、製造部・品質保証部・総務部・経理部などがありますが、これらの組織は、一定期間ではなく長期に渡って運営され、既知の通常業務（ルーチンワーク[16]）を繰り返し実行する組織です。

例えば、世の中に存在しない特殊なネジを開発することは「特別な目的」に従ったプロジェクト業務であり、JIS ネジ B0205 を量産することは「所定の目的」に従ったルーチンワーク業務だと言えます。

PM

◎プロジェクトとは、特別な目的を達成するための期間限定の臨時的な組織のこと。

【プロジェクトとは？】（図１－１－１）



今まで有ったものから、今までになかったものを創り出すこと。

15 **リソース** 開発の実行に必要な人材・資材・資金・情報・時間などの有形無形の資源のこと。

16 **ルーチンワーク** 手順や手続きが決まっている定型作業のこと。

プロジェクトマネジメントとは何か

マネジメント[17]をするということは、みんなの先頭に立って道を切り開き、障害物を取り除き、チームの盾となり、進むべき方向を指し示し、メンバーが最大限の力を発揮できるようにすることです。メンバーたちの後方に立って全員突撃と叫ぶリーダーではなく、俺に続けと先頭を駆け抜けるリーダーになりたいものです。

プロジェクトマネジメントとは、「プロジェクトを、その目的および目標を達成するために、適切に運用すること」だと言えます。 マネジメントという言葉は一般的に管理とか経営という言葉に翻訳されることが多いものですが、いずれもしっくりしませんので、本書では直訳の「マネジメント」表記のままとします。

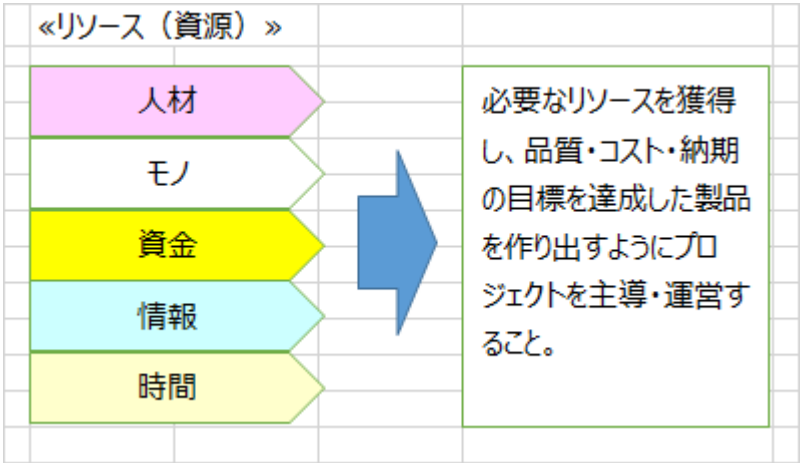
プロジェクトマネジメントとは、一言でいえば「プロジェクトを成功させること」だと言えます。

プロジェクトマネジメントを担う役職がプロマネであり、プロマネの役割は獲得したリソース（人・モノ・カネ・情報・時間）の制限内で、プロジェクトのＱＣＤ[18]目標をすべて達成し、プロジェクトを成功に導くことだと言えます。

PM

◎プロジェクトマネジメントとは、
獲得したリソース（人・モノ・カネ・情報・時間）の制限内で、プロジェクトのＱＣＤ目標をすべて達成し、プロジェクトを成功に導くこと。

【プロジェクトマネジメントとは？】（図１－１－２）



[17] **マネジメント** ものごとを適切に運用し、所定の成果が生み出されるように組織を運営すること。

[18] **ＱＣＤ** Quality 品質、Cost コスト、Deliverly 納期、の頭文字をとった略語で、プロジェクトの成功指標とされている。

プロジェクトの成功要件

一般的にソフトウェア開発プロジェクトの成功要件は、次の三つの目標の達成だと言われています。

【プロジェクトの成功要件】

1．品質目標　Q：Ｑｕａｌｉｔｙ
2．コスト目標　Ｃ：Ｃｏｓｔ　⇒利益目標
3．納期目標　Ｄ：Ｄｅｌｉｖｅｒｙ　⇒生産性[19]目標

コスト目標は同時に利益目標となり、また納期目標は同時に生産性目標となります。

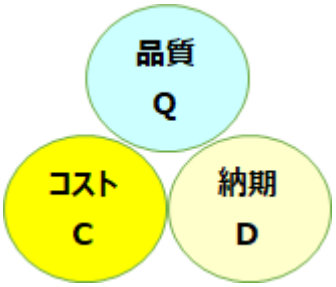
これらのＱＣＤの目標は、三つが同時に達成されて初めて成功プロジェクトだと言えます。
例えば下記の三つのケースにおける目標の達成はほとんど意味がないということが分かります。

① 品質目標は達成したがコスト・納期目標は未達成の場合
② コスト目標は達成したが品質・納期目標は未達成の場合
③ 納期目標は達成したが品質・コスト目標は未達成の場合

コスト目標は、開発組織内においては利益目標として表現されることもあります。

これらの三つの要件は顧客と開発組織との間の契約によって約束されるもので、どれか一つでも未達成ならば契約違反となり、民事的に公的な処罰の対象となることもあります。

【プロジェクト成功の３要件】（図１－１－３）



品質・コスト・納期目標の同時達成

[19] **生産性** 時間当りに生み出される成果物の量および質を表わす指標。

プロマネ

プロジェクトを成功に導くためにはプロマネは下記の実行が必要になります。

【プロジェクトを成功させるポイント】

① 妥当な見積りによる妥当な開発期間および開発工数の確保を行うこと。
② 要求仕様の早期凍結を行うこと。
③ リスクの早期解消を行うこと。
④ リーダーが開発仕様の全体像を押さえておくこと。
⑤ 開発メンバーに対して適切な仕事の配分を行い、十分に仕様の理解をさせておくこと。
⑥ 開発効率化・失敗防止のための継続的な改善活動[20]を実行すること。

効率的開発、良い品質、納期の確保、利益の創出を実現するためには上記の活動が必要になり、これらの活動を主導するのがプロマネの中心的な役割だと言えます。

PM

◎プロジェクトの成功要件は、目標ＱＣＤの三つを同時に達成すること。

[20] **改善活動** ＱＣＤに関する問題の改善を図りプロジェクトを成功に導くと同時に、人材の育成を図る活動のこと。特に失敗に学ぶ活動は有効な結果を生む。

1－2 プロマネの役割【人的資源マネジメント】

プロマネの基本的な役割

プロジェクトマネジメントは基本的にマネージャの職務であり、すなわち一定の人事権や決済権を持った人でなければ実行できない領域がありますが、マネージャの職位ではないプロマネ補佐的な仕事を担うプロマネ初心者においても、正規のプロジェクトマネージャの役割を十分理解した上で自分の役割を果たす必要があります。 プロマネ業務の一覧は次に示す通りです。

【プロマネ業務の一覧】

① 適正な見積り交渉の実行
② 早期仕様凍結[21]の実行
③ プロジェクト計画書の作成
④ プロセス管理の実行
⑤ プロジェクト進捗管理の実行
⑥ リスク管理の実行
⑦ 課題管理の実行
⑧ 損益管理の実行
⑨ プロジェクト完了報告
⑩ 日次情報共有会議の実行
⑪ プロジェクト関係他社開発チームとの情報共有および連携

(プロマネ業務一覧の詳細は「付録図表１． プロマネ業務一覧」を参照のこと)

多くのソフトウェア開発企業においては、時間的余裕が非常に少ないために、適正人材の育成やプロジェクトマネジメント業務をほとんど実行できないでいるのが実態だと思われます。前記の１１項目はリスク物件や一定規模以上の物件では必須の業務であり、どれが欠けても失敗の要因となります。

プロジェクトマネジメント力を持つリーダーの育成は、マニュアルや管理規定書によって育成されるものではなく、日々の仕事における一歩先を目指す活動、すなわち改善活動によって育成されていくものです。多くの人が参加する改善活動から、多くの有能なプロジェクトリーダー[22]が生み出されることでしょう。

PM

◎プロマネの最終的な役割は、
プロジェクトから一人たりとも犠牲者を出すことなく、関係者すべてに対して、物心共に実りの多い成果を生み出すと同時に、顧客の満足と信用を得ること。

---

21 **仕様凍結** 顧客の要求内容を仕様化し、顧客の同意を取り付けること。基本的な仕様を設計着手前に凍結することがプロジェクトの成功要因の一つとなる。

22 **プロジェクトリーダー** プロジェクトを統率する筆頭責任者のこと。プロマネや開発リーダーなどを束ねる統括マネージャのこと。筆頭プロマネの別称。

プロジェクトの規模とプロマネの役割

本書が対象とするプロジェクトの規模は、開発費が１００人月以下の中小規模の派生開発プロジェクトであり、そのプロジェクトにおけるプロマネの基本的な役割については、前記の、「プロマネの基本的な役割」で述べた通りですが、その中においてもプロジェクトの規模や難易度によっては、必要なプロマネの人数やその役割の分担も変わってきます。また元請会社と下請け会社の混成チームの場合にも、プロマネの役割分担は変わってきます。

中小規模プロジェクト体制におけるプロマネの立ち位置には、大体次のような種類があります。

１．プロマネ１名（開発リーダー兼務）

小規模で難易度が低いプロジェクトにおいてはプロマネが開発リーダーも兼務する場合が多いでしょう。

２．プロマネ１名（プロマネ専任）

小規模で難易度が高い場合および中規模で難易度が低い場合はプロマネ業務の専任者を立てる場合が多いでしょう。

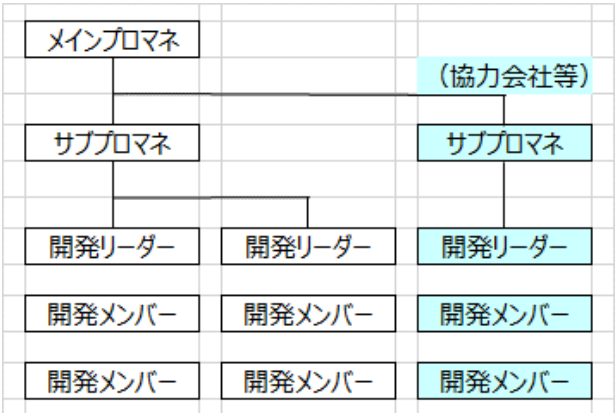
３．メインプロマネ１名＋サブプロマネ１名または２名

中規模以上のプロジェクトにおいては専任のメインプロマネにサブプロマネをつける場合が多いでしょう。またプロジェクトの難易度に応じてサブプロマネを２名体制にすることや、サブプロマネに開発リーダー兼務をさせないような施策も必要となります。

サブプロマネはメインプロマネの役割の一部を担いつつ、開発リーダーとしての職務を果たしている場合が多いでしょう。

開発体制上のプロマネの位置を図で表わすと次の図のようになります。

【プロジェクト体制におけるプロマネの位置】（図１－２－１）



サブプロマネが必要なプロジェクトにおいて、サブプロマネはメインプロマネの手が回らない進捗管理や課題管理などの、開発により近いプロマネ業務を担当することになります。

プロジェクトに下請け開発メンバーが参加する場合、下請け開発のリーダーは上図のサブプロマネの位置に配置されることが多いでしょう。

また多数の小規模プロジェクトのプロマネを担当し、開発リーダー職も兼務せざるを得ない立場のプロマネないしはサブプロマネも少なからず居ますが、自分の限界を越えそうな場合は、早めに開発リーダー職の兼務を解きプロマネ業務への専任化などを上司に相談すべきでしょう。

複数物件の平行プロジェクトマネジメントも基本は単独物件の場合と同じであり、プロジェクトマネジメント手法がきちんと確立されていれば決して難しいことではないでしょう。

その為にはプロセス管理表を初めとした他の管理表等の書式および運用も標準化されている必要があり、誰が使用しても妥当な結果を生むようになっている必要があります。

サブプロマネ（下請けプロマネ）の役割および限界

多くの場合、サブプロマネはプロジェクトの人事権や決済権を持たされていませんので、下請け開発のリーダーがサブプロマネとしてプロジェクトに参加した場合、遂行可能な役割には限界があります。

「プロマネの基本的な役割」で示した１１項目のプロマネ業務の内で、自分が担当する業務の遂行において、元請け[23]のメインプロマネの不作為によって実行が困難になりそうな局面が少なからず発生するものと思われます。

例えば、プロマネ業務の②早期仕様凍結の実行が元請け側においてなされない場合、自分の責任範囲外だからと言って放置してしまえば、自分の担当である仕様に関しても、いつまでたっても決定されない状況に陥り、自分の責任も果たすことができない結果となってしまいます。

このような事態を避けるためには、下請けのサブプロマネであったとしても元請けのプロマネに対し仕様案の提案や顧客との仕様打ち合わせへの参加などの、前向きな対策の進言を行う必要があります。

そうは言っても、元請け側においては「そんなことは余計なことだ」とか、下請け側においては「そこまではやりたくない」というような反応が出て来そうですが、これであきらめてしまえば両者ともどもプロジェクトを失敗に導くことになります。

メインであれ、サブであれプロマネはプロマネの基本的役割を十分に認識し、お互いに不足な部分を補完し合いながら、プロマネとしての役割をまっとうする必要があります。

---

23 **元請け** 顧客から仕事を最初に受注する企業のこと。いわゆる１次ベンダーのこと。

1－3 プロジェクトマネジメントにおける問題とリスク【リスクマネジメント】

プロジェクトマネジメントにおける主な問題

プロマネにおけるプロジェクトマネジメントの問題は大きく分けて次の二点に集約されます。

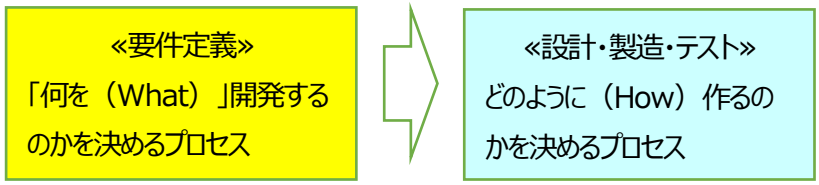
① プロジェクトマネジメントの全体像を理解していないこと。

② プロジェクトマネジメントの実行の方法を理解していないこと。

要するに何を（Ｗｈａｔ）どのような方法（Ｈｏｗ）で実行して良いのか理解できていないところに問題の真因があります。この問題は開発実務における、確定された要求仕様（Ｗｈａｔ）に基づいて設計・製造・評価（Ｈｏｗ）を行う場面に置いても同様の問題を引き起こしています。

何を実行すべきかが分かっていなければ、実行段階には進めないことは当たり前で、何を（Ｗｈａｔ）すべきかを最初に理解しておくことが最重要なことだと言えます。

【プロジェクトマネジメントのプロセス】（図１－３－１）



何をすべきか、さらにそれをどのように処理すべきかを考え実行する基本的な力は、結局、目の前の現実や事実を自分の目で直視し、それが意味するところを自分自身で解釈・判断し、それをどのような方法で解決するのかを自分で決定し、自分で行動を起こすという、自律性の力の発揮によるものと言えます。この自律性[24]の発揮こそが、全ての問題を解消する原点となっています。

PM

◎プロジェクトマネジメントにおける主要な問題点

◎プロマネが、プロジェクトマネジメントの全体像を理解していないこと。

◎プロマネが、プロジェクトマネジメントの実行の方法を理解していないこと。

[24] **自律性** 直面する問題に対して自分自身の目で確かめ、状況を把握・判断し、実行に移す能力のこと。反対語は他人依存性。

プロジェクトマネジメントにおけるリスク

プロマネがプロジェクトをマネジメントすると言うことは、下記のリスクを解消するということに他ならず、これらのリスク回避活動がプロマネの役割であり職責そのものだと言えます。

◆【プロジェクトマネジメントリスク一覧】

- □ 外注まで含めた統合的プロジェクトマネジメント（品質・進捗・コスト・人材管理）の不在
- □ マルチベンダー下における責任分担の不明確さ
- □ 開発体制の不備（能力不足の人員配置）
- □ 上位マネジメントとの意思疎通不良（孤立無援によるリスクの増大）
- □ 無理な開発費減額要求の無条件な受入れ
- □ 無理な納期短縮要求の無条件な受入れ
- □ 見積りのルール破り（承認ルーチンの無視）
- □ あいまいな開発スコープおよび要求仕様の放置
- □ 仕様変更管理ルールの不在
- □ 条件なしの規模の大きな機能追加要求の受け入れ
- □ 利益確保を優先し、品質の確保を怠る
- □ 不適切な事前着手
- □ 知識のない分野の開発請負
- □ 新言語採用にあたっての準備不足
- □ 新技術採用における準備不足
- □ 既存資産流用の可否判断ミス
- □ 新旧システムの同時並行開発
- □ 他社ブラックボックス・ソフトウェアの事前検証不足
- □ システムの劣化あるいはスパゲッティ化の放置
- □ 納期厳守で不良品を出荷する
- □ 納期遅延による必要工程の手抜きないしはスキップの黙認ないしは指示

上記以外にもマネジメントに関するリスクは多く存在しますが、いずれにしても大部分のリスクは開発の初期工程に集中しており、これらのリスクを解消することがプロマネを含めたマネジメントの仕事だとも言えます。

◎プロジェクトをマネジメントすると言うことは、
プロジェクトのリスクを解消するということ。

PM

1－4 プロセスをマネジメントする【統合マネジメント】

プロジェクトをマネジメントするということは、プロセスをマネジメントすることと言い換えることもできます。開発プロジェクトは複数の工程すなわち複数のプロセスで構成されています。各プロセスは INPUT―PROCESS―OUTPUT で構成されており、前のプロセスの OUTPUT は次のプロセスの INPUT となり全てのプロセスは互いに接続されています。プロジェクトマネジメントにおける計画のプロセスと実行のプロセスの連鎖を示すと次のようになります。

【連鎖するプロセス】（図１－４－１）

計画のプロセス				実行のプロセス		
INPUT	PROCESS	OUTPUT	→	INPUT	PROCESS	OUTPUT
入力	処理	出力		入力	処理	出力

あるプロセスの INPUT は前のプロセスの OUTPUT に相当します。プロジェクトを成功に導くためには、各プロセスにおける OUTPUT が約束通りのもので、約束された時期に提供される必要があります。

PM

◎Input－Process－Output が適切な時期に正しく実行されれば、
プロジェクトは必ず成功する。

プロセス管理表について

これらの複数のプロセスにおける主要なイベント[25]および成果物を時系列的に並べたものがプロセス管理表と呼ばれるものです。

多くの品質問題の本質は、「段取り 7 分の後３分」と言われる通り、仕事の段取りの悪さにあります。段取りとは、すなわちプロセス管理のことです。

プロセス管理における改善すべき主な問題点として次のようなものがあります。

【プロセス管理の問題点】

- 各工程における実行手順忘れ
- 各工程における手抜き行為
- 人依存・経験依存のバラバラな開発手順
- 視覚化されていない開発手順

[25] **イベント** 出来事、行事のこと。開発工程の節目ごとにおいて行われる、成果物などの受け渡しのことを指す。

これらの問題は、通常のガントチャートなどのスケジュール進捗表やレビューなどでは発見することが難しく、以下の対策が必要です。

【プロセス管理の改善】

- 標準プロセス管理表の作成（開発業務手順のビジュアル化）
- 標準プロセス管理表に基づく開発手順の実行（開発業務品質の確保）
- 標準プロセス管理表実行のレビュー
- プロセス管理表にＩＳＯ[26]・ＣＭＭＩ項目を絡めたチェック機能の盛り込み強化

プロセス管理表[27]はＥＸＣＥＬ表の横軸に、各プロセスの主だった成果物ないしはイベントについて手順項目・作業内容・担当者名・成果物名・チェック欄・実施予定日・実施日などの記述欄を設け、縦軸は各プロセス区分の時系列の早いもの順の並びとなった表で表されています。

下記はその一部分を示したものです。

【プロセス管理表（部分）】（表１－４－１）

【プロセス管理表】（サンプル）　＊ＰＬ（ｏｒＰＭ）、ＳＰＬ（ｏｒＳＰＭ）

	NO	手 順	進捗	作 業	担当者	成果物	担当チェック	ＰＬチェック	実施予定日	実施日	レビュー結果
事前準備	1	顧客・ステークホルダーの情報収集		顧客の要求内容・予算・納期等の事前把握	開発部	・調査報告書					
	2	新ＯＳ・新言語・新システム対応の準備		新ソフトウェア・新ハードウェア等に対する技術知識のマスターおよびプロトタイプによる事前調査	開発部	・調査報告書					
計画プロセス	1	見積もり依頼 見積もり回答		仕様，スケジュール，要員検討、見積依頼	ＰＬ，ＳＰＬ	・見積回答書 ・リスク管理表					
	2	要求仕様の明確化		要求事項の明確化	ＰＬ	・要求事項の明確化チェックリスト作成					
	3	仕様検討、承認		SEとの仕様打ち合わせ	ＳＥ,ＰＬ、開発メンバー	・議事録／要求仕様書の承認					
	4	不具合歯止策の選定、決定		開発に対する有効な歯止め策を	ＰＬ、ＳＰＬ、開発メン	・歯止めチェックリスト					
	5	スケジュール		大、小日程の作成	ＰＬ、ＳＰＬ	・大日程表、小日程表					
	6	プロジェクト品質目標の作成		プロジェクト内品質目標の作成	ＰＬ、ＳＰＬ	・プロジェクト品質目標					
	7	プロジェクト計画書		プロジェクト計画書の作成 リスク管理表更新	ＰＬ，ＳＰＬ	・プロジェクト計画書					
				プロジェクト計画レビュー 開発開始会議	ＳＱＡグループメンバー、ＰＭ，ＰＬ	・レビュー記録					
	8	外注見積委託作成		外注見積委託ＤＢに登録	ＰＬ，ＳＰＬ	・見積書					
	9	開発管理		開発管理資料の作成	ＰＬ，ＳＰＬ	・プロセス管理表					
						・スケジュール表					
						・課題管理表					
						・規模進捗					
	10	デザインレビュー①		要求事項の明確化チェックリスト	ＳＥ→開発部	・要求仕様の採点確定					

全プロセスを表現したプロセス管理表のサンプルは「付録図表２．プロセス管理表」を参照のこと。

◎プロセス管理表なしではプロジェクトマネジメントはできない。

PM

---

26 **ＩＳＯ**　国際標準化機構。ISO 9001 は、品質マネジメントの国際標準規格。ISO21500 は、プロジェクトマネジメントの国際標準規格。

27 **プロセス管理表**　開発における各工程の流れに沿って、開発の主要なイベントおよび成果物を時系列順に並べた管理表で、各工程における実行手順忘れ・手抜き行為などを防止するためのもの。

ソフトウェア開発における主なプロセスサイクル

ソフトウェア開発における主なプロセスサイクル[28]を下記に示します。

【プロセスサイクルの種類】（図１－４－２）

本書におけるプロセスサイクル	事前準備 Preparation	計画 Planning	実行 Executing	管理 Monitoring& Controlling	振り返り Retrospect	継承 Transfer
ＰＭＢＯＫ	立上げ Initiating	計画 Planning	実行 Executing	監視・管理 Monitoring& Controlling	終結 Closing	
一般的なソフトウェア開発	要件定義 見積り	基本設計 詳細設計	製造	評価		

上図を見て分かるように本書におけるプロセスサイクルと他のプロセスサイクルとの違いは、最初の事前準備プロセスおよび最後の継承プロセスの有無にあります。他の形式のプロセスサイクルにおいても、この二つのプロセスの機能は何らかの形で取り込まれていると思われますが、その重要性についてフォーカス[29]されているとは思えません。

事前準備のプロセスは、そのプロジェクトの成功の枠組みを早い段階で決定づける重要な仕事であることを再認識する必要があります。また継承のプロセスは、次に続くプロジェクトおよび開発者たちに前のプロジェクトにおいて獲得されたノウハウおよびヒト・モノ・カネ・情報などの有形・無形の資産を継承するためには必須のプロセスです。単発のプロジェクトだけを成功させるだけの近視眼的な思考では、開発組織の永続的な発展を望むことはできないでしょう。

[28] **プロセスサイクル** 開発の始まりから終了までの一連の流れを、複数のプロセスに分割して表したもの。

[29] **フォーカス** 焦点を当てること、ないしは明確に表現すること。

【プロジェクト・マネジメントのまとめ】

なぜプロジェクトというものが発生し、なぜプロジェクトマネジメントというものが必要になり、プロジェクトマネージャという職種が生まれたのかということについて、次にその背景を説明します。

現在端末機器レベルにおいても、数百万行を超えるソフトウェアも珍しくありません。このような大規模な開発を行うためには、どうしてもプロジェクト体制を組み分業体制による開発が必要となります。その結果一人の開発者がコントロールできる領域は限定的にならざるを得ず、システムの全体像を把握することは全く困難な状況にあります。アプリケーション[30]の開発者においても、設計業務・プログラム作成・評価業務などが分業体制となり、個々の開発者にとっては自分が担当する業務以外が見えなくなっているのが実態です。この見えない状況を放置したままでは、開発がうまく行くわけはありません。

プロジェクトマネジメントとは、このような分離分業の間に発生する情報分断を埋め、開発プロセスを統合的にまとめあげることを意味しており、その役割を担う人がプロジェクトマネージャ（プロマネ）なのです。

プロマネという職種が生まれた時代的な背景を時系列的に整理すると次のようになります。

1．ハードウェアおよびＯＳ[31]・開発言語の進化によってソフトウェアの大規模な開発が可能になった。
2．その結果ソフトウェア開発業務の工程別ないしは機能別の分業が始まった。
3．業務の分業は、情報の分断を発生させ開発が困難になった。
4．その結果情報の分断を埋め、開発プロセスを統合的にまとめあげる職種であるプロマネが生まれた。（統合プロジェクトマネジメント[32]）

PM

◎プロジェクトマネジメントとは、プロジェクトを成功に導くために必要な
プロセス（手順）およびプラクティス（実践）を統合マネジメントすること。

---

30 **アプリケーション** 特定の企業の業務にあわせて作成されるソフトウェアのこと。応用ソフトウェアともいわれる。

31 **ＯＳ** オペレーティングシステム（Operating System）。コンピュータの基本的な動作を制御する、システムソフトウェア。

32 **統合プロジェクトマネジメント** 工程別・機能別に分離・分業化された開発業務における情報分断の隙間を埋め、開発のプロセスを統合的にまとめあげる業務のこと。

１．６つのプロセスと１０のマネジメントプラクティスについて

プロジェクトの統合マネジメントの具体的な内容については、第３章以降において説明することになりますが、それらの理解を容易にするために、各工程のプロセスとそれぞれの工程で実行されるべきプロマネのプラクティス（役割）の全体像を先に示しておくことにします。

プロジェクトマネジメントの中核的な役割は、第２章以降で述べるプロジェクトの６つのプロセスおよび各プロセスにおいて実行が必要な１０のマネジメントプラクティスを統合マネジメントすることにあります。

PM

◎６つのプロセスとは、

１．事前準備　２．計画　３．実行　４．管理　５．振り返り　６．ノウハウ継承

PM

◎１０のマネジメントプラクティスとは、

①統合マネジメント　②ステークホルダーマネジメント　③リスクマネジメント
④スコープマネジメント　⑤品質マネジメント　⑥コストマネジメント
⑦タイムマネジメント　⑧人的資源マネジメント　⑨調達マネジメント
⑩コミュニケーションマネジメント

１０のマネジメントプラクティスの役割を把握するための概要説明を次に示しますが、同時にそれぞれについての解説に関する章・節を付記しましたので、各マネジメントプラクティスの実行の流れを把握するためにご利用いただきたいと思います。

【プロセスとプラクティスのマトリクス図】（図１－まとめ－１）

		プロセス					
		事前準備	計画	実行	管理	振り返り	継承
プラクティス	統合						
	ステークホルダー						
	リスク						
	スコープ						
	品質						
	コスト						
	タイム						
	人的資源						
	調達						
	コミュニケーション						

２．各マネジメントプラクティスの役割

統合マネジメント[33]：プロジェクトを統合的にマネジメントする

◎統合マネジメントの役割

１．９つのマネジメントプラクティスを相互に結び付け、一つの統合されたプラクティスとして働くようにすること。

２．分割された工程間および組織間に発生する情報やコミュニケーションの断絶を防ぎ、一連の統合されたプロセスとして働くようにすること。

【解説の章・節】

◎**第１章** プロジェクトマネジメント

１－１．プロジェクトについて　ｐ２

１－２．プロマネの役割　ｐ６

１－４．プロセスをマネジメントする　ｐ１１

◎**第３章** 計画のプロセス

３－８．●統合的なプロジェクト体制の構築　ｐ９２

３－１０．プロジェクト計画書の作成　ｐ９５

３－ｈ１．プロジェクト統合管理　ｐ９７

◎**第６章** 振り返りのプロセス

６－１．プロジェクト活動の振り返り　ｐ２０２

６－２．プロジェクト完了報告書　ｐ２０７

◎**第７章** ノウハウ継承のプロセス

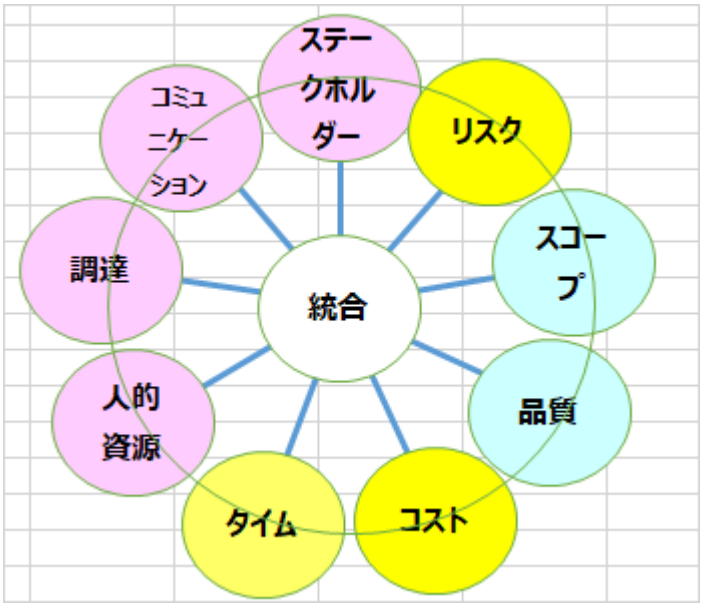
７－１．ノウハウの継承　ｐ２１６

７－２．改善活動のすすめ　ｐ２２０

---

[33] **統合マネジメント** プロジェクトマネジメントの基本的な９つのマネジメントプラクティスである、ステークホルダーマネジメント、リスクマネジメント、スコープマネジメント、品質マネジメント、コストマネジメント、タイムマネジメント、人的資源マネジメント、調達マネジメント、コミュニケーションマネジメントを相互に結び付け一つの統合されたプラクティスとして働くようにすること。

【統合マネジメント】（図１－まとめ－２）



ステークホルダー[34]マネジメント[35]

：ステークホルダーとの良好な関係を保つようにマネジメントする

◎ステークホルダーマネジメントの役割

顧客をはじめとしたステークホルダーからＱＣＤ等の情報を収集し、早期の仕様凍結・妥当な見積り・開発リスクの解消を通して、顧客の信頼を獲得すること。

【解説の章・節】

◎**第２章**　事前準備のプロセス

２－１．顧客・ステークホルダー情報の収集　ｐ３０

【ステークホルダーマネジメント】（図１―まとめ－３）

◎予算
◎納期
◎仕様

[34] **ステークホルダー**　プロジェクトを取り巻く利害関係者のこと。

[35] **ステークホルダーマネジメント**　顧客をはじめとしたステークホルダーからＱＣＤ等の情報を収集し、早期の仕様凍結・妥当な見積り・開発リスクの解消を通して、顧客の信頼を獲得すること。

リスクマネジメント[36]：プロジェクトへのリスクの影響が最小限になるようにマネジメントする

◎リスクマネジメントの役割

開発プロセスに内在するリスクを早期に発見し、解消することでプロジェクトのＱＣＤ目標を達成させること。

【解説の章・節】

◎**第１章** プロジェクトマネジメント

１－３．プロジェクトマネジメントにおける問題とリスク　ｐ９

◎**第２章** 事前準備のプロセス

２－２．未経験エリアへの対応準備　ｐ３８

２－３．事前準備工程のリスク　ｐ３９

◎**第３章** 計画のプロセス

３－１．早期の仕様凍結　ｐ４６

３－２．要求仕様に関するリスク　ｐ５７

３－５．●見積りにおけるリスク　ｐ６７

３－８．●開発体制不備による失敗リスク　ｐ９０

◎**第４章** 実行のプロセス

４－１．リスクの解消　ｐ１０４

４－３．●開発管理表の概要　ｐ１２４

【リスクマネジメント】（図１－まとめ－４）

[36] **リスクマネジメント** 開発プロセスに内在するリスクを早期に発見し、解消することでプロジェクトのＱＣＤ目標を達成させること。

スコープ[37]マネジメント[38]：プロジェクトへの要求事項を確実に達成するようにマネジメントする

◎スコープマネジメントの役割

要求仕様の内容および開発範囲を明らかにし、顧客と同意しておくこと。

スコープの明確化は見積りの妥当性確保、ＱＣＤ目標値の達成に必須の要件となる。

【解説の章・節】

◎**第３章** 計画のプロセス

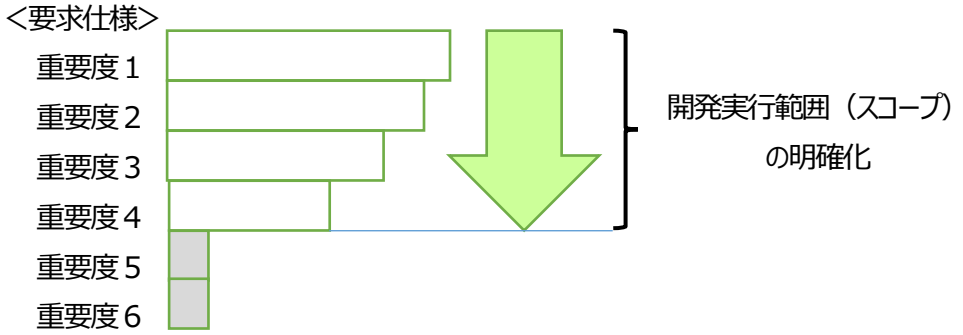
３－１．早期の仕様凍結　ｐ４６

３－３．要求仕様の構造化（ＷＢＳ）　ｐ５８

３－４．顧客要求の重要度順位の設定　ｐ６０

３－７．プロジェクト目標の設定　ｐ７３

【スコープマネジメント】（図１―まとめ－５）



[37] **スコープ** プロジェクトが開発すべき要求内容およびその範囲のこと。

[38] **スコープマネジメント** 要求仕様の内容および開発範囲を明らかにし、顧客と同意しておくこと。 スコープの明確化は見積りの妥当性確保、ＱＣＤ目標値の達成に必須の要件となる。

品質マネジメント[39]：作業と成果物の品質目標を達成するようにマネジメントする

◎品質マネジメントの役割

業務品質および製品品質を数値化し、達成すべき品質の目標値を設定し、品質目標を達成するための改善活動を伴った開発業務を遂行すること。

【解説の章・節】

◎**第３章** 計画のプロセス

３－７．●品質目標値の設定　ｐ７６

◎**第４章** 実行のプロセス

４－２．ＱＣＤ目標値の達成　ｐ１１４

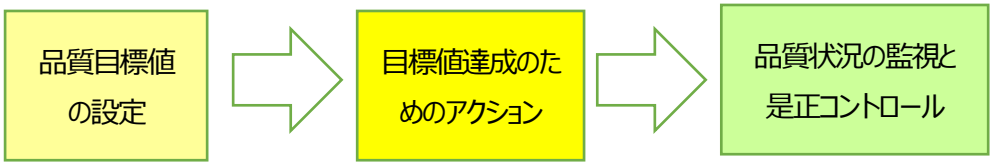
４－３．品質目標値の達成　ｐ１１７

４－６．改善活動の実行　ｐ１３５

◎**第５章** 管理のプロセス

５－２．品質管理　ｐ１５６

【品質マネジメント】（図１―まとめ－６）



[39] **品質マネジメント** 業務品質および製品品質を数値化し、達成すべき品質の目標値を設定し、品質目標を達成するための改善活動を伴った開発業務を遂行すること。

コストマネジメント[40]：コスト（利益）目標を達成するように予算とコストをマネジメントする

◎コストマネジメントの役割

達成すべき開発コストおよび利益の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。

【解説の章・節】

◎**第３章** 計画のプロセス

３－５．見積り　ｐ６２

３－７．●コスト・利益目標値の設定　ｐ８０

◎**第４章** 実行のプロセス

４－２．ＱＣＤ目標値の達成　ｐ１１４

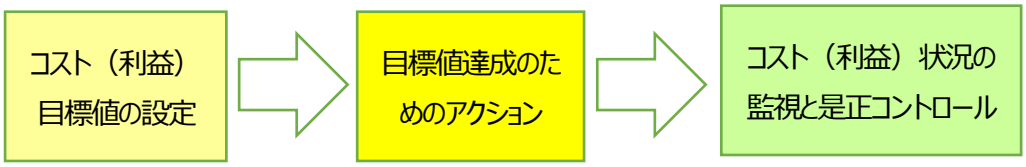
４－４．コスト・利益目標値の達成　ｐ１２６

４－６．改善活動の実行　ｐ１３５

◎**第５章** 管理のプロセス

５－３．コスト・利益管理　ｐ１６４

【コスト（利益）マネジメント】（図１―まとめ－７）



[40] **コストマネジメント** 達成すべき開発コストおよび利益の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。

タイムマネジメント[41]：スケジュール目標を達成するように進捗をマネジメントする

◎タイムマネジメントの役割

達成すべき納期および生産性の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。

【解説の章・節】

◎**第３章** 計画のプロセス

３－５．見積り　ｐ６２

３－６．スケジュールの作成　ｐ７１

３－７．●納期・生産性目標値の設定　ｐ８４

◎**第４章** 実行のプロセス

４－２．ＱＣＤ目標値の達成　ｐ１１４

４－５．納期・生産性目標値の達成　ｐｐ１３１

４－６．改善活動の実行　ｐ１３５

４－ｈ１．獲得時間の最大化と失う時間の最小化　ｐ１４１

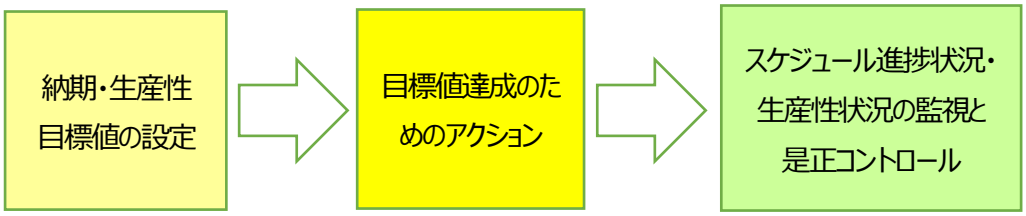
◎**第５章** 管理のプロセス

５－４．進捗管理　ｐ１６８

◎**第７章** ノウハウ継承のプロセス

７－２．改善活動のすすめ●コンセプト＃１．時間の獲得　ｐ２２４

【タイムマネジメント】（図１―まとめ－８）



[41] **タイムマネジメント**　達成すべき納期および生産性の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。

人的資源マネジメント[42]：チームのパフォーマンス[43]向上を目指して人材のマネジメントをする

◎人的資源マネジメントの役割

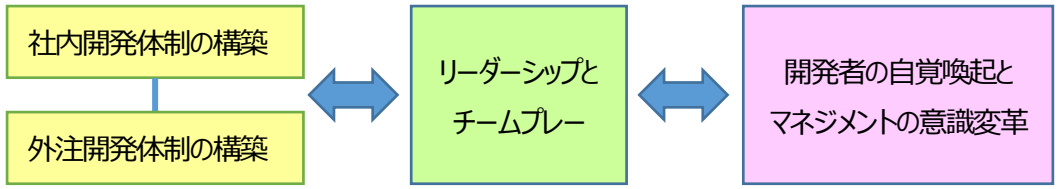
1．ＱＣＤ目標の達成に必要な人材を社内･社外に求め、プロジェクト体制を構築する。
2．開発者およびマネジメントの自覚を喚起し、モチベーション を維持する。
3．相互義務の履行、相互扶助の実行を通してチームプレーを行い、プロジェクトのパフォーマンスの最大化を図る。

【解説の章･節】

◎**第１章** プロジェクトマネジメント
　１－２．プロマネの役割　ｐ６
◎**第２章** 事前準備のプロセス
　２－ｈ１．開発者における自覚の喚起　ｐ４０
　２－ｈ２．マネジメントの意識変革　ｐ４１
◎**第３章** 計画のプロセス
　３－８．●プロジェクト体制構築の要件　ｐ８９
◎**第５章** 管理のプロセス
　５－ｈ２．チームプレーの実行　ｐ１８６
◎**第６章** 振り返りのプロセス
　６－ｈ１．失敗の原因を他に求めないこと　ｐ２１０
　６－ｈ２．やる気を起こす～モチベーションの喚起　ｐ２１２

【人的資源マネジメント】（図１―まとめ－９）

<プロジェクト体制の構築>



---

[42] **人的資源マネジメント**　１．プロジェクトのＱＣＤ目標達成に必要な人材を社内･社外に求め、プロジェクト体制を構築すること。２．開発者およびマネジメントの自覚を喚起し、モチベーション を維持すること。３．相互義務の履行、相互扶助の実行を通してチームプレーを行い、プロジェクトのパフォーマンスの最大化を図ること。

[43] **パフォーマンス**　組織における開発処理能力のこと。またはソフトウェアの処理速度のことを指す。

調達マネジメント[44]：外部からの調達を円滑にするようにマネジメントする

◎調達マネジメントの役割

1．社内人材の不足を補うために、適切な人材を外部から採用し、外注開発体制を構築する。

2．開発の遂行に必要な機器・機材を、適切な時期に適量を購入するための手配を行う。

【解説の章・節】

◎**第３章** 計画のプロセス

３－８．●外注開発体制の構築　ｐ９１

３－９．開発環境の手配　ｐ９４

【調達マネジメント】（図１―まとめ－１０）

協力会社からの人材調達 **&** 開発機材の調達

[44] **調達マネジメント**　１．社内人材における不足を補うために適切な人材を外部から採用し、外注開発体制を構築すること。
２．開発の遂行に必要な機器・機材を適切な時期に適量を購入するための手配を行うこと。

コミュニケーションマネジメント[45]

：プロジェクトのコミュニケーション[46]を円滑にするようにマネジメントする

◎コミュニケーションマネジメントの役割

1．人的な情報系の中枢を司る。
2．何をどうすべきかを明らかにし、開発者全員で共有する。
3．プロジェクトの共通の目標に向かって開発者全員のベクトルを揃える。

【解説の章・節】

◎**第３章** 計画のプロセス

３－８．●統合的なプロジェクト体制の構築　ｐ９２

３－ｈ１．●統合管理を阻害するもの　ｐ９８

◎**第５章** 管理のプロセス

５－ｈ１．コミュニケーションの活性化　ｐ１７５

【コミュニケーションマネジメント】（図１―まとめ－１１）

◎直接コミュニケーション
◎日次情報共有会議
◎顧客とのコミュニケーション
◎工程間のコミュニケーション
◎元請け・下請け間のコミュニケーション

直接コミュニケーション

---

[45] **コミュニケーションマネジメント**　１．人的な情報系の中枢を司ること。２．何をどうすべきかを明らかにし開発者全員で共有すること。３．プロジェクトの共通の目標に向かって開発者全員のベクトルを揃えること。

[46] **コミュニケーション**　聞く・話す・読む・書くなどの能力を使ってお互いの理解・合意・納得を得る双方向性の行為のこと。

３．６つのプロセスと１０のマネジメントプラクティスの関連図

プロジェクトマネジメントの全体像を、６つのプロセスおよび１０のマネジメントプラクティスのマトリクス図で表したものを次に示します。次章以降をお読みいただくにあたって各章・各節が、どのプロセスの何のマネジメントプラクティスについての解説であるのかが容易にご理解いただけるものと思います。

【プロセスとマネジメントプラクティスの関連図（部分）】（表１－まとめ－１）

		プロセス				
	マネジメントの役割	第１章 プロジェクトマネジメント	第２章 事前準備のプロセス （Preparation）	第３章 計画のプロセス （Plan）	第４章 実行のプロセス （Do）	第５章 管理のプロセス （Monitoring ＆ Controlling）
マネジメント＆プラクティス	統合マネジメント （Integration Management） プロジェクトを統合的にマネジメントする	１－１．プロジェクトについて １－４．プロセスをマネジメントする		３－８．●統合的なプロジェクト体制の構築 ３－１０．プロジェクト計画書の作成 ３－ｈ１．プロジェクト統合管理		
	ステークホルダーマネジメント （Stakeholder Management） ステークホルダーとの関係を良好に保つようにマネジメントする		２－１．顧客・ステークホルダー情報の収集			
	リスクマネジメント（Risk Management） プロジェクトへのリスクの影響が最小限になるようにマネジメントする	１－３．プロジェクトマネジメントにおける問題とリスク	２－２．未経験エリアへの対応準備 ２－３．事前準備工程のリスク	３－１．早期の仕様凍結 ３－５．●見積りにおけるリスク ３－８．●開発体制不備による失敗リスク	４－１．リスクの解消 ４－３．●開発管理表の概要	
	スコープマネジメント（Scope Management） プロジェクトへの要求事項を確実に達成するようにマネジメントする			３－１．早期の仕様凍結 ３－３．要求仕様の構造化（ＷＢＳ） ３－４．顧客要求の重要度順位の設定 ３－７．プロジェクト目標の設定		５－１．要求仕様の変更管理

全体図は「付録図表３．プロセスとマネジメントプラクティスの関連図」を参照のこと。

第２章

事前準備のプロセス

概要

第２章　事前準備のプロセス

プロジェクトを成功に導くためには、どのような事前準備が必要なのかについての解説です。

- ステークホルダー、特に顧客の情報は、プロジェクトの成功に大きな影響を与える（２－１．）
- 未経験エリアに対する事前準備は、重要なリスクヘッジ策の一つとなる（２－２．）
- 初期工程における様々なリスクの事前解消は、プロジェクト成功の確率を高める（２－３．）
- 開発者相互のコミュニケーションの活性化は、開発者の当事者意識を喚起しプロジェクトの重要な成功要因となる（２－ｈ１．）
- マネジメントには、単なる管理する人という意識からの脱却が求められる（２－ｈ２．）

プロジェクトマネジメントの最初のプロセスは事前準備です。

事前準備のプロセスは、次の計画のプロセスにおけるプロジェクト計画書を作成するために必要な情報の収集、および開発を開始するに当たって必要な未経験分野への対応準備、および開発者・マネジメントにおける心構えの準備をするためのプロセスです。

どのような仕事であれ、事前の準備を整えておけば仕事が成功する確率は高くなります。プロジェクトマネジメントは、初期における情報戦が成否を決定づけるという視点から、本章においては顧客を中心としたステークホルダーからのキー情報の入手、プロジェクトにおける未経験エリアに対する対応準備について、そのポイントを押さえると共に、プロジェクトチームにおける心理面の準備として、開発者における自覚の喚起およびマネジメントにおける意識の変革を取り上げました。

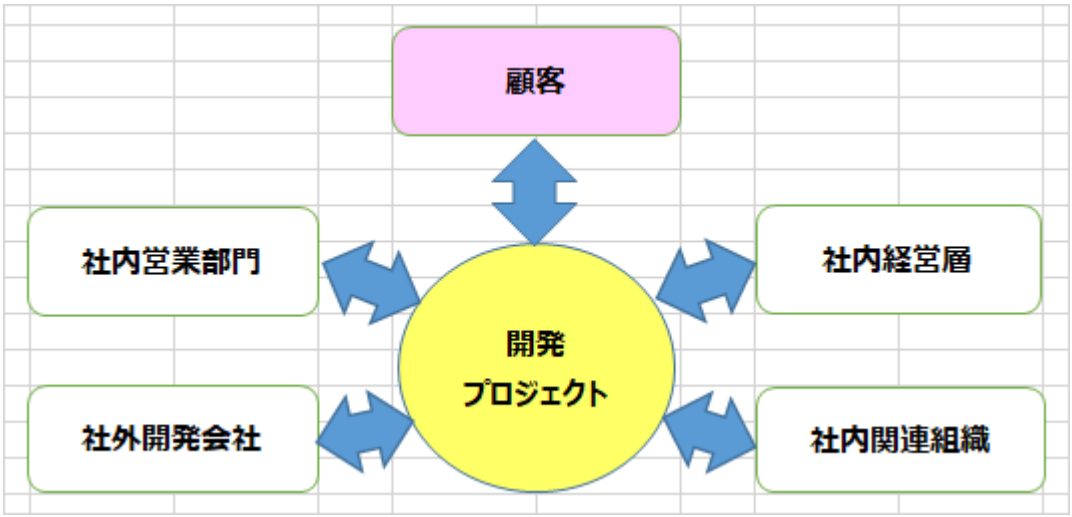
【Job Activity】

2－1 顧客・ステークホルダー情報の収集【ステークホルダーマネジメント】

ステークホルダーとは、開発プロジェクトに直接ないしは間接的な利害関係をもった組織および人々のことを指しています。代表的なステークホルダーとしては、顧客や社内の経営層・営業・フィールドサービス部門[47]・品質保証部門・ハードウェア開発部門・研究開発部門・製造部門などの関係組織および、社外の下請け等の協力会社のことだと理解しても良いでしょう。

収集すべきステークホルダーの情報は、相手が開発プロジェクトに何を期待しているのかという情報、および開発側として知りたい情報の二つになります。

【ステークホルダー】（図２－１－１）



ステークホルダーから収集すべき情報

１．ステークホルダーが、開発プロジェクトに何を期待しているのかの情報

２．開発側が、ステークホルダーから入手したい情報

[47] **フィールドサービス部門** 市場にて稼働するシステムなどの保守・整備部門。

顧客情報の収集

ＩＴ開発を情報戦だと認識している人は意外に少ないと思われます。未知のものをできるだけ正確に把握するためには、事前調査に時間をかけ、そこで得られた情報を小まめに記録し、蓄積しておくことが必要となります。

開発にとって重要である顧客情報、とりわけ顧客特有の思考・行動様式[48]や納期・要求仕様・発注予算に関する情報は、プロジェクトの生命線ともいえる情報ですが、プロマネをはじめとした開発関係者は相も変わらず待ちの姿勢が多く、これらの情報をいち早くキャッチするための、顧客との直接コミュニケーション[49]がおろそかになっている場合が多いのではないでしょうか。これらのキー情報をどれだけ早い時点でキャッチアップできるかが、プロジェクトの勝敗を決める重要なファクター[50]になります。 プロマネやプロジェクトリーダーにおいては開発開始前の段階で、自分自身が直接行動することにより、これらの情報を収集しておく必要があります。

準備

◎事前入手が必須な情報は次の四つ、

１．顧客特有の思考・行動様式	①コストカット重視 or 品質重視 ②情緒的 or 合理的 ③硬直的 or 柔軟的 ④競合他社追随的 or 独自性重視、など
２．納期情報	決定済 or 未決定、短納期 or 妥当な納期
３．要求仕様の骨子	骨子が明確 or 不明確、仕様決定力が強い or 弱い
４．発注予算情報	ある程度判明 or 不明

☆上記情報はプロジェクトの成否を分ける重要なリスク情報であり、不都合な情報であった場合は早めの対策が必要となります。

---

48 **思考・行動様式** 人や組織における、その成長過程において習得された、特有なものの考え方および行動。

49 **直接コミュニケーション** 実際に顔と顔を突き合わせた対面コミュニケーションのこと。ダイレクトコミュニケーション、フェイス トゥ フェイス コミュニケーションとも言われる。⇒参照「５－ｈ１． ●直接コミュニケーションの重要性」

50 **ファクター** ある結果を生み出す要因。

顧客が開発側に何を期待しているのかの情報

第一に入手が必要な情報は、顧客が開発側に何を期待しているのかを示す顧客特有の思考・行動様式に関する情報です。開発会社が接する顧客側の部署は、システム部とか情報企画部とか呼ばれる部署が一般的ですが、それらの部署はその会社特有の思考・行動様式を強く反映しています。プロジェクトを成功裏に終了するためには、彼らがどのような思考様式や行動様式をもっているのかを、開発が始まる前の受注活動の中で素早く理解しておく必要があります。

顧客側の思考・行動の代表的なパターンを示します。

◎身勝手な顧客の例

できるだけ安い金額と短納期でたくさんの仕様を盛り込み、良い品質のものを希望する顧客は、開発側にとっては困ったものですが、自分が車や家を購入する場合を振り返ってみると、一方的に非難することもできません。これが日本の消費者の典型的な、商品購入における思考・行動のパターンなのです。経験上このような顧客は全顧客のうちの約９割を占めていると思われます。 このような顧客に対して何の対策もなしに無防備に受注・開発を行うと以下のような問題が発生してくるでしょう。

①仕様の範囲（スコープ）がいつまでも確定されない

一旦決まった仕様ですら二転三転し、開発工程の後半になっても仕様変更や追加が止まらない。

②プロジェクトが危機に陥る

約束した開発期間や開発費は、開発半ばで消費し尽くしてしまう状況に陥る。

このような状況を回避するためには、事前準備のプロセス期間中に顧客の思考・行動に関する上記のリスクを排除しておく必要があります。下記のリスクヘッジ[51]策が有効です。

【身勝手な顧客に対する防衛策】

1．営業部署との連携による、顧客との密接なコミュニケーションを実行すること。
2．“新システム一式、仕様は打合せによる”などという基幹仕様未定な状態で受注はしないこと。
3．仕様凍結の最終期限を相互で約束しておくこと。
4．仕様凍結後の仕様変更・追加は別途見積り（開発費および開発期間）とすること。
5．短納期・低開発費を強く迫る顧客に対しては、開発済みの標準的な仕様の提案を行うこと。

身勝手な顧客に対する防衛策によって、顧客を妥当な要求に導き、開発側は妥当な開発期間・開発費を提示することで、両者ともに納得できる形で開発を進めることが可能になります。

---

51 **リスクヘッジ**（Risk Hedge）損失・失敗の防衛策のこと。

さらにこれらのリスクヘッジは開発プロジェクトにおける最大のリスクである、要求仕様問題・見積り問題および開発行為のやり直しという三つのリスクの排除に大きく貢献します。

【プロジェクトにおける三大リスク】

1．いつまでも決まらない要求仕様

2．精度が低い見積り

3．開発行為のやり直し

開発側が知りたい情報の入手

開発側が最も知りたい情報は次の三つになります。

1．納期情報

2．要求仕様の骨子

3．発注予算情報

これらの情報は、事前準備の段階での入手が必要です。

あらかじめ顧客のこれらの情報を大まかにでも入手していれば、顧客との仕様検討や見積り等を開発側主導で進めることが可能になります。 これらの情報の入手は以下の行動によって可能になるでしょう。

1．営業部署と連携した、顧客との密接なコミュニケーションの実行による信頼関係の醸成。

2．失注競合他社の提案書などの情報入手。

準備

◎顧客の情報を知れば失敗リスクの半分は消える、
残りの半分はプロジェクト自身の開発力にある。

その他のステークホルダー情報の収集

社内営業部門の情報

社内営業部門および所属ＳＥ[52]から入手が必要な主な情報は、顧客情報および発注予算情報の二つです。

◎顧客情報

顧客情報については、「２－１．●顧客情報の収集」で述べた通りですが、営業独自のルートで入手されたものについても開発側にて知る必要があります。この情報をどの程度入手できるかについては、開発側と営業側の信頼関係の深さに依存しており、日頃からの両者の密接なコミュニケーションおよび共存共栄の行動が物を言うことになります。

◎営業部門から開発プロジェクトへの発注予算

この情報は、普通は入手不可能です。

強い営業組織を持つ開発会社において、営業部門は開発部門にとって第二の顧客ともいえる組織です。

受注活動の中心を担っているこの営業部門は、顧客からの受注活動において当然のことに自部門の利益を中心として考えた予算組みを行い、その範疇内でのプロジェクトの実行を期待しています。

やはり彼らの思考・行動様式も平均的な顧客と同様に、開発部門に対して、できるだけ安い金額と短納期でたくさんの仕様を盛り込み、良い品質のものを希望するのが常です。開発側の見積り回答に対する彼らの最初の反応は常に「いくら値引きできる？」なのです。

【顧客情報収集のルート】（図２－１－２）

☆予算・納期・仕様等情報の流れ

[52] **ＳＥ** システムエンジニア（System Engineer）の略称。本書では、顧客の要件を要求仕様にまとめる職種として使用。

わずかばかりのリスク対応金額しか含まれていない見積りからそのリスク分を外し、更に相手の強い要求に負けて必要な開発費までカットしてしまえば、プロジェクトを成功させることはほとんど不可能なことでしょう。そう言うわけで、強制的なコストカットによる品質の悪化の悪循環が繰り返され、開発部は「高い・悪い・遅い」の評価が定着してしまいます。

このような悪循環を断ち切るために開発部は次のような施策の実行が必要になります。

☆根拠のない値引き要求は断ること

無理な値引き要求をしてくる人は大体開発仕様の理解に乏しく、その開発費が妥当か否かという判断ができないために、自部門の目標利益と比べた結果としての値引きを要求してくる場合が多いものです。

そのような強引な値引き要求に対抗するためには、要求者に対してこのプロジェクトに関する営業組織における損益計算が記載された稟議書[53]の提示を要求する必要があります。９割以上の値引き要求はこの方法で退けることが可能でしょう。

☆本当に営業部門としても赤字受注の物件の場合

この場合は当然のことに公式の稟議書の提示で赤字の状態が示されたもので、開発部としても応分の値引きに応じる必要があります。ただし必要な開発費・開発期間以下に切り下げるような方法ではそのプロジェクトは失敗する可能性が高くなります。そのようにならないためには開発部門において下記のような施策を常時継続的に実行する必要があります。

【正当な値引き要請に応えるための原資の確保】

1．開発組織がコストセンター[54]ならばプロフィットセンター[55]に切り替えること。
2．常時継続的に改善活動を実行することで失敗を減らし、生産性を上げていくことで利益の蓄積および開発力の向上をはかること。

準備

◎根拠のない値引き要求には稟議書の提示を求めること。
正当な値引き要請には改善活動による利益の還元を。

---

53 **稟議書** りんぎしょ プロジェクト遂行の許可申請書のこと。通常その事業の損益見込み計算書が添付されている。

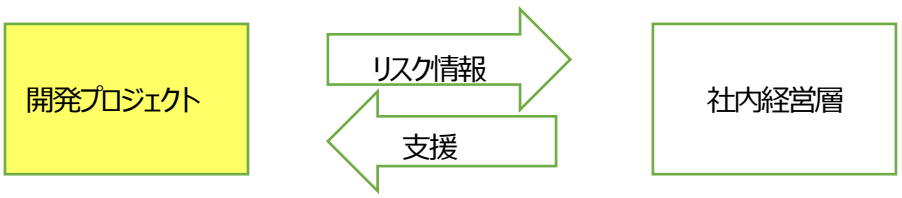
54 **コストセンター** あらかじめ用意された予算により運営される組織のこと。利益確保の直接的な責任は負わされない。
⇒参照「４－４．コスト・利益目標値の達成●プロフィットセンターとコストセンター」

55 **プロフィットセンター** 独立採算により運営される組織のこと。利益確保の責任を持つ。
⇒参照「４－４．コスト・利益目標値の達成●プロフィットセンターとコストセンター」

社内経営層の情報

プロジェクトにとって開発部門を統括する社内経営層からの支援は必要不可欠です。特にプロジェクトの事前準備期間中においては、プロジェクトが知り得るすべてのリスク情報および対応策を報告しておくことで、経営層との密接な情報共有を行うことが、プロジェクトに対するさまざまな支援を可能にします。

【社内経営層とのリスク情報の共有】（図２－１－３）

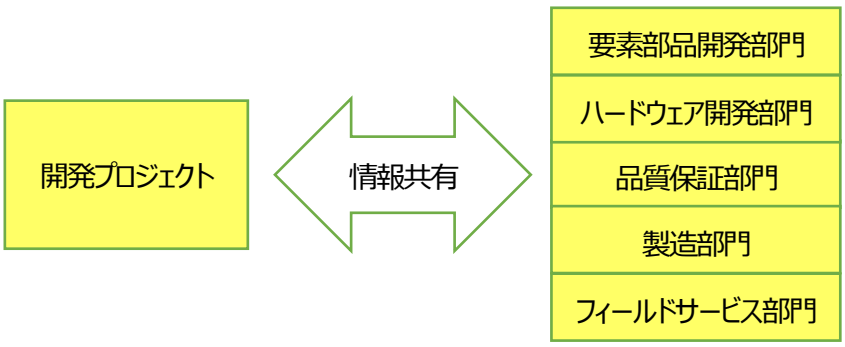


社内関連組織の情報

プロジェクトにとって必要とされる社内関連組織としては、プラットホーム[56]や要素部品[57]の開発部門、品質行政を担う品質保証部門、ハードウェア開発部門、製造部門、フィールドサービス部門等があります。

アプリケーション開発が中心のプロジェクトにとっては、これらの関連組織とのリスクも含めた情報共有や連携[58]行動はプロジェクトの成功には必要不可欠であり、定期的な統合プロジェクト会議などの実施が必要です。

【社内関連組織との連携】（図２－１－４）



---

56 **プラットフォーム** アプリケーションソフトウェアを支える、ＯＳおよびミドルウェアなどの基幹ソフトウェア。

57 **要素部品** ディスプレードライバー、プリンタードライバーなどのＩＯ系デバイスをコントロールするソフトウェアなど。

58 **連携** 連絡提携という言葉を略したもので、連絡を密に取り合って、一つの目的のために一緒に物事をすること。

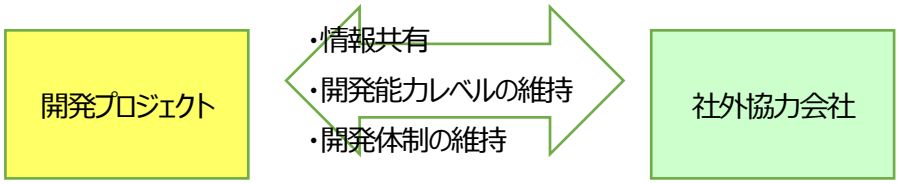
社外協力会社の情報

ある程度以上の規模の開発においては、社外の協力会社に開発工程のある部分を委託することも多くのプロジェクトで実行されています。プロジェクトにおける開発能力は、社内と外注でのレベルを一定以上の状態に保つ必要があります。どちらが低くてもプロジェクトを成功させることはできません。

外注化を行っているプロジェクトにおいては、常時協力会社の要員のスキルレベル情報・コスト情報・品質情報・生産性情報を入手し、それに対する評価を行い、結果をフィードバックしておく必要があります。 このような活動を通して、プロジェクト開始前に必要な能力レベルの外注の人材を確保することが可能になります。

またプロジェクトの実行期間中においては、協力会社も含めた統合管理の実行は不可欠で、ＱＣＤ・リスク情報や協力会社からの要望事項等のすべての情報について、日次の情報共有会議や定期的なプロジェクト統合会議[59]などにおいての情報共有が必要です。

【社外協力会社との連携】（図２－１－５）



ステークホルダーと共に

それぞれのステークホルダーたちが開発プロジェクトに何をどの程度期待しているのか、ということを数値で表わされた情報として入手することが、開発プロジェクトにおけるＱＣＤ目標の設定の基盤となります。

さらに開発プロジェクトがそれぞれのステークホルダーに提供できるものと、開発組織がステークホルダーたちから得ることができるものを正確に把握し、そのバランスをとることが、これらの利害関係者たちとの永続的な信頼関係および共存共栄を保つことになります。

準備

◎ステークホルダーとの信頼関係構築に必要なことは、
相手の期待を知りそれに応えること、および
こちらが相手から入手したいものとのバランスをとること。

59 **プロジェクト統合会議** ステークホルダーとの定期的な情報共有会議。

2－2 未経験エリアへの対応準備【リスクマネジメント】

顧客を初めとしたステークホルダーから事前に得られた様々な要求情報の中でも特に重要なものとしては、当該プロジェクトにとって未経験のエリアに関する情報があります。これらの未経験エリアに関する情報の入手は早ければ早いほど良く、素早くこれらの問題に対処しておく必要があります。

例えば、未経験のＯＳ・開発言語・業種アプリケーション・ネットワーク・ハードウェアプラットフォーム・端末機器などに関する顧客要求が、プロジェクトにとって未経験ないしは経験が浅いものであった場合、プロジェクトが開始されてからこれらの知識の習得や人材・機材の用意をしていては、限られた開発期間の中でプロジェクトを無事に完遂することは不可能です。

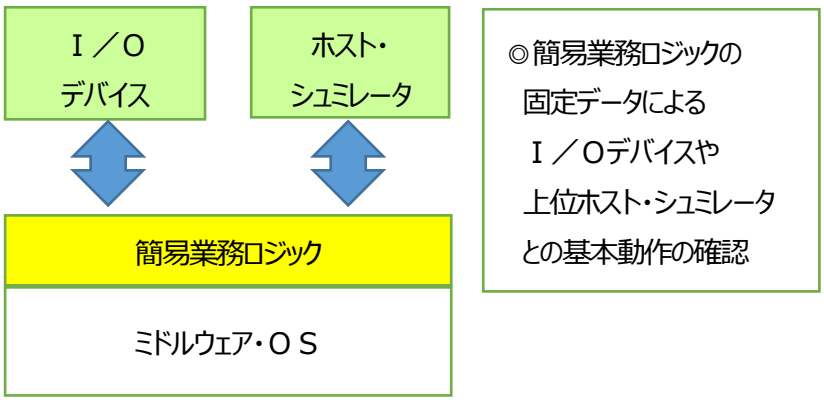
これらの未経験エリアの問題への最も効果的な対応方法は、プロトタイプ[60]の作成による事前の検証です。プロトタイプによる検証は未知のものにおける弱点・相性・性能・品質などを事前に知るためには必ず必要となるものです。予算も時間もないから、とりあえずプロジェクトを始めてから考えようというような姿勢ではプロジェクトは必ずといって良いほど失敗するでしょう。

またプロトタイプの作成および検証期間中に、未経験エリアに関する知識の習得および必要な人材・機材の手配を行っておけば、計画プロセス以降のプロセスはスムーズに進行できるでしょう。

準備

◎未知・未経験エリアに対する事前準備は、プロトタイプによる事前検証を。

【プロトタイプ・アプリケーションの例】（図２－２－１）



[60] **プロトタイプ**（prototype） ソフトウェア開発において、本番プログラムの作成に先立って新技術・新機能の検証および問題点の洗い出しのために仮に作成されたプログラムのこと。

2－3 事前準備工程のリスク【リスクマネジメント】

事前準備工程におけるリスクは次の表に示した通り、非常に多岐に渡っており、後工程に被害を及ぼす前に解消しておく必要が有ります。

【事前準備工程のリスク管理表（部分）】（表２－３－１） 拡大表は付録図表４．を参照のこと。

	プロジェクト名：				初版発行日： 更新日：		文書番号：
	リスク要因分類	QCD影響 Q	C	D	事前準備工程リスク	✓	チェック 検出日
ヒトに関するリスク	①他者依存的姿勢（自律性の放棄）	○	○	○	・リーダーシップの欠如		
					・マルチベンダー下における分担責任のあいまいさ		
					・他社パッケージの未検証採用		
					・基幹技術の選定（なりゆきまかせ）		
					・開発組織の自律性不足		
	②上位マネジメントの関与不足		◎		・顧客交渉戦略・能力不足		
					・不適切なプロマネの選任		
	③ユーザーの参加・協力度不足	◎	○	○	・顧客の参加・協力度が低い		
	④組織能力不足（未熟な組織文化、戦略の欠如）	◎	◎	◎	・不適切な事前着手（短納期）		
					・新旧システムの同時並行開発（開発量・時期の重複）		
					・顧客との名目だけの共同研究開発		
					・能力不足のプロマネの選任		
					・頻繁な更新によるシステムの劣化・スパゲッティ化		
	⑤見積り能力	◎	◎	◎	・安易な現行機能搭載の保証		
					・フィット＆ギャップ調査能力		
	⑥要件定義能力	◎	○	○	・要件定義能力不足		
	⑦リーダーのプロジェクトマネジメント能力（外部交渉、タイムマネジメント、現場主義、見える化能力など）	◎	◎	◎	・顧客交渉戦略・能力不足		
					・不適切な事前着手		
					・新旧システムの同時並行開発		
					・企画能力不足		
					・適時の支援依頼発信能力不足		
					・問題の放置		
					・未経験言語採用の準備不足		
					・開発目的の誤り（受注優先）		
					・技術者のトレーニング不足		
					・開発体制の不備（経験者・能力・知識・人数）		
					・未経験分野／業界参入の準備不足		
					・新技術採用の準備不足		
					・パッケージベース開発におけるフィット＆ギャップ調査不足		
					・タイムリミット管理能力不足		
					・技術方式選定対応における柔軟性		
	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）	◎	○	○	・オープンシステム知識の不足		
					・開発言語能力不足		
					・新技術知識不足		
					・顧客業務知識不足		
	⑨プロセス管理の有無	◎	○	○	・プロセス管理の不足		
	⑩コミュニケーション能力（阻害・ギャップ）	◎	○	○	・顧客のパートナー化失敗		
					・顧客とのコミュニケーション能力		
	⑪関連部署との連携不足	◎	○	○	・組織間の協調性・コミュニケーション		
モノ	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）	◎	○	○	・開発ガイドラインの不備（開発プロセス、設計手順書、コーディング規約、単体・総合評価テスト手順書等）		
	⑬開発のベースの有無	◎	◎	◎	・開発のベースの有無		
	⑭開発環境の不備	◎	○	○	・開発環境の不備		
カネ	⑮開発費不足	◎	◎	◎	・赤字受注		
					・開発費不足（見積りの失敗、開発の失敗）		
	⑯資源投入戦略の誤り（開発費投入時期ミス）	◎	○	◎	・資源投入戦略の誤り		
情報	⑰あいまいな開発範囲（スコープ）	◎	◎	◎	・あいまいなスコープ		
	⑱あいまいな要求仕様	◎	◎	◎	・あいまいな要件		
	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）	◎	○	○	・情報の不備・不足		
					・未経験分野／業界の業務知識・技術情報不足		
					・新技術の技術情報不足		

【Human Activity】

事前準備プロセスにおいて必要な、人間力発揮の活動には次のようなものがあります。

開発者における自覚の喚起【人的資源マネジメント】

プロジェクト活動を成功させるために最も重要な条件は、その活動を実行する人々における当事者意識[61]をはじめとした心構え、すなわち自覚がしっかりとしていることです。合理的な解決手段が生かされるか否かは、その当事者自身の心の問題にかかっています。自覚とは言葉をかえて言えば「その気があるか」ということです。その気がなければ仕事もついなおざりになり力も入りません。その気を出すためには、何のために誰のためにその仕事をするのかという明確な動機が必要です。

現在、多くの仕事が過度に専門化し細分化されてしまい、同時に組織の細分化も進み、その結果組織間のみならず個人の間でのコミュニケーションも希薄になり、組織・個人の孤立化が進んでいます。そのような職場環境において孤立状態におかれがちな開発者たちは、自分は何のために開発という仕事をしているのかという当事者意識、すなわち自覚を見失いがちになってしまいます。貧乏暇なしで日々の仕事に追いまくられていると、全く自分のことを振り返る余裕もなく、ちょっとした幸運に有頂天になったかと思えば、ちょっとした失敗に深く落胆し、気持ちの落ち着く暇もありません。このような一喜一憂を繰り返しているうちに仕事をしている意味を見失うような状態に陥ることも少なからずあります。

このような状態に陥っている開発者たちに向って、開発者としての自覚を持てとか、モチベーションを上げろとか、やるべきことをやれ、などと檄を飛ばしても逆効果しか得られないでしょう。

この問題を解決するためには不必要な組織の細分化を減らし、開発者相互の密接なコミュニケーションを復活する必要があります。

このような荒廃した環境を作るのも復興させるのも人間、とくに組織をリードする者たちです。プロマネの役割と責任を自覚する必要があります。 個人および集団に実利の獲得と人間的成長という希望をもたらす運動の一丁目一番地は、コミュニケーションの活性化であり改善活動に他ならないでしょう。

準備

◎自覚の喚起・当事者意識の回復には、
コミュニケーションの活性化と改善活動の実行を。

61 **当事者意識** 他人に寄りかかることなく自分の役割をまっとうするという心構えのこと。

事前準備

2－h 2 マネジメントの意識変革【人的資源マネジメント】

プロジェクトにおけるマネジメントの役割および責任は重大です。私見ですが、プロジェクトの失敗の８割以上はマネジメントの失敗にあると考えられます。ここで言うマネジメントに相当する職種としてはプロジェクトを主導するプロマネおよびプロジェクトリーダーを指しており、職位としてはなんらかの管理・監督の権限を持つ人たちのことです。マネジメントの人間がプロジェクトにおける自分の役割と責任を自覚していなければ、プロジェクトの成功は難しいでしょう。

準備

◎プロジェクト失敗原因の８割は、プロジェクトマネジメントの失敗による。

開発メンバーの不出来はマネジメントの失敗から

開発組織における部長や課長と呼ばれている人たちの本来の役割は、プロジェクト・マネジメントのはずですが、多くの現場ではそのようになってはいません。私管理する人、みなさん開発する人という考え方が蔓延しています。開発者たちの作業状況を管理・監視するだけで、チームをリードしようとはしない管理職たちではプロマネの職責を果たすことはできないでしょう。 マネジメントの職位になると、いつしか自分が偉い人間になったという勘違いが始まり、現場への足も遠のいてしまいがちになります。そしていつしか現場の状況も見えなくなり、判断や指示に過ちが多くなってきます。開発組織のマネジメントこそ、常に自分はプロジェクトマネージャであるという意識を持ち行動し続ける必要があります。

開発現場のリアルな状況を把握するためには、日々現場における直接コミュニケーションを絶やさないようにする必要があり、毎日の現場巡回によるフォローに合わせて、次のような短時間の日次情報共有会議[62]の実行が最も効果的です。

【短時間の日次情報共有会議】

① 前日の進捗の実績をヒアリング＆アドバイスする。
② 本日の実行予定をヒアリング＆アドバイスする。
③ 担当者が抱えている問題点をヒアリング＆アドバイスする。

注意点：マネージャのための会議ではなく、開発者が抱える問題点の解決に役立つ会議にすること。

準備

◎有効な短時間情報共有会議は、密接なコミュニケーションを実現する。

---

62 **日次情報共有会議** プロジェクトメンバーにて毎日行われる短時間の情報共有会議。

マネジメントの役割と責任は重い

誹謗・中傷・非難の言葉ばかりしか聞かれない職場、はたまた誰も口を利かない沈黙の職場において、良い仕事がなされることはありません。このような職場にＰＭＢＯＫやＣＭＭＩ[63]を強制しても成果を上げることは難しいでしょう。いつも仕事が異常に忙しく物事の調査時間もなければ、新しい技術を学習する時間もなく、やっとのことで出荷した製品はクレームだらけ、というような環境で開発を行っている開発者たちは段々とやる気も元気もなくし、終には心や身体を病むことになってしまいます。

このような状況を生み出す多くの責任は、マネジメントにあると思われます。

このようにマネジメントの失敗は、人を困窮させ、組織を破壊し、取り返しのつかない事態を招いてしまいます。マネジメントが出来ない者をマネージャにしてはいけません。不適格なマネージャは再教育するか、それでもだめならその任を解くしかありません。

堕落した開発マネジメントの一例を下記に示します。

「このような内容では出荷承認は出せません」と品質保証担当の課長が言った。これに対して開発担当課長は「この程度の不具合残件やテスト残件で出荷できないとはどういうことなのか。今日出荷しなければ二日後の客先稼働ができなくなる。その責任をあなたは取れるのか」と言っていた。なんという本末転倒な言いぐさだろうか。これは脅しや恐喝と同じです。

さらに「市場でトラブルが起きたらどうするのか」と品証課長が言った。これに対して「そんなことやってみなければ分からない。出たらすぐ修正すればいいだけだ」などと言っている。

自分たちがなすべきテストも不具合修正も未完のままで、すぐ出せ、ハンコを押せなどとよく言えたものです。

同様の問題で今日も日本のトップ企業であるＫ製鋼、Ｎ自動車における品質データ改ざんや不適切な品質検査についてのニュースが流れており、一夜にして企業存亡の危機を自ら招いているのです。

われわれの職場において改善活動を命じている者が、普段から非合理的・情緒的な言動ばかり繰り返しているようでは誰もその指示に従うわけもありません。自分がやりたい放題をやっていては、下にいる者たちも隠れてやりたい放題になってしまうでしょう。そのような人たちに時間やお金を渡しても何の成果も得られず、投資した時間やお金は無駄に浪費されたような結果となるに違いありません。

準備

◎プロマネは課長・部長という職位よりも
プロジェクトマネージャという職務を果たすべし。

---

63 **ＣＭＭＩ**（Capability Maturity Model Integration、能力成熟度モデル統合版）　プロジェクトの成功指標であるＱＣＤを担保するプロセス（工程）がどの程度組織に定着しているのかを示す国際標準モデル。レベル１から５までの５段階で評価される。レベル５が最高レベル。

【事前準備プロセスのまとめ】

事前準備のプロセスにおける活動を整理すると次のようになります。

【Job Activity】

準備

【顧客情報の収集】

◎事前入手が必須な情報は次の四つ、

①顧客特有の思考・行動様式 ②納期情報 ③要求仕様の骨子 ④発注予算情報

事前準備プロセスは顧客情報収集および未経験開発に備える重要なフェーズです。プロマネにとって見積りに必要な顧客における顧客特有の思考・行動様式の情報、希望納期の情報、仕様骨子[64]の情報および予算情報の収集に全力を注ぐ必要があります。これらの情報はプロジェクトの成否にかかわる重要な情報となります。顧客の情報を知れば失敗のリスクの半分は消えると思っても間違いないでしょう。

準備

【社内営業組織への対応】

◎根拠のない値引き要求には稟議書の提示を求めること。

正当な値引き要請には改善活動による利益の還元を。

社内の営業組織経由で受注する場合においては、営業組織からの発注予算情報の収集が重要になってきます。開発組織が必要とする開発費および開発期間に満たない減額要求に対しては、開発組織の見積り内容および開発リスクなどの妥当性を説明しつつも、稟議書の提示要求などタフなネゴシエーション[65]が必要になってきます。また正当と思われる開発費の減額要求に対しては、内部留保の使用など柔軟に対応できるようにしておく必要もあります。

準備

【ステークホルダーとの信頼関係構築】

◎ステークホルダーとの信頼関係構築に必要なことは、

相手の期待を知りそれに応えること、および

こちらが相手から入手したいものとのバランスをとること。

---

64 **仕様骨子** 要求仕様の内、その中心となる重要な仕様。

65 **ネゴシエーション** 交渉力のこと。特に見積り交渉においては、強い交渉力が必要とされる。

準備

【未経験開発エリアへの準備】

◎未知・未経験エリアに対する事前準備は、プロトタイプによる事前検証を。

未経験エリアの開発が想定される場合には、プロトタイプによる事前検証が必須となってきます。未経験のＯＳや言語の使用にあたっては、事前にプロトタイプを作成し、その長所・短所および癖について一通りの評価をしておく必要があります。プロマネにおいてはプロトタイピング[66]に必要な予算・人材の事前手当が必要になります。

【Human Activity】

準備

【開発者における自覚の喚起】

◎自覚の喚起・当事者意識の回復には、コミュニケーションの活性化を。

◎有効な短時間情報共有会議は、密接なコミュニケーションを実現する。

事前準備の期間は、プロマネのもとに徐々にメンバーが集まり始める時期でもあり、プロジェクトのウォーミングアップを行う重要な期間となります。プロジェクト活動においてその基本となるのはコミュニケーションであり、意思疎通や情報共有がプロジェクトの成否を決定づけます。プロマネはコミュニケーションの活性化を図るために、プロジェクトの事前準備段階から毎日ベースの短時間情報共有会議を励行する必要があります。

準備

【マネジメントの意識変革】

◎プロジェクト失敗原因の８割は、プロジェクトマネジメントの失敗による。

◎プロマネは課長・部長という職位よりも、プロジェクトマネージャという職務を果たすべし。

プロマネは、プロジェクト失敗原因の８割はプロジェクトマネジメントの失敗にあるという現実を十分に認識し、課長・部長というような職位よりもプロジェクトマネージャという職務を果たすようにする必要があります。

---

66 **プロトタイピング** プロトタイプを作成すること。参照⇒「プロトタイプ」

第３章

計画のプロセス

概要

第３章　計画のプロセス

プロジェクトの枠組みを決める、計画のプロセスで必要なアクションについての解説です。

- 明確な仕様の早期凍結は、プロジェクト成功要因の一つとなる（３－１．）
- 仕様の早期凍結には、要求仕様におけるさまざまなリスクの解消が必要（３－２．）
- 要求仕様の構造化（ＷＢＳ）は、見積りおよび開発進捗管理のベースとなる（３－３．）
- 顧客要求事項の優先度の設定は、効率的な開発に貢献する（３－４．）
- 精度が高く妥当な見積り回答は、受注確率を高め妥当な開発条件を獲得する（３－５．）
- ＷＢＳに基づいたスケジューリングは、妥当な作業配分の元になる（３－６．）
- ＱＣＤ目標値の設定は、プロジェクトを成功に導く指標となる（３－７．）
- 開発規模・難易度に応じた開発体制の構築は、プロジェクトの成功要因となる（３－８．）
- 開発に必要なモノを、必要な時期に、必要な量を、手配しておく必要がある（３－９．）
- 計画のまとめとして、開発行動計画書であるプロジェクト計画書を作成する（３－１０．）
- プロジェクトの統合管理は、全てのマネジメント活動を目標に沿って機能させる（３－ｈ１．）

計画のプロセスは、プロジェクトを実行するために必要なすべての活動を計画し、最終的にはプロジェクト計画書としてまとめ上げるプロセスで、本章では前記の順に解説を進めていき、最後にすべての活動を統合するプロジェクトの統合管理についての解説を行います。

計画

【Job Activity】

3－1 早期の仕様凍結【スコープマネジメント】&【リスクマネジメント】

あいまいな要求仕様と無理な短納期･低コスト

プロジェクトの起点は、顧客あるいはベンダー[67]から提示される要求仕様および見積り依頼にあります。これはまた同時に、プロジェクトにおける問題の起点もここにあるということになります。開発の第一段階における要求仕様の品質および見積り回答の妥当性が、プロジェクトの成否を決定づけることになります。

顧客要求に関する主なリスクは次の通りです。

□ あいまいでいつまでも決まらない要求仕様

回避策 ⇒ 参照：ｐ４７「早期の仕様凍結」、ｐ４９仕様の理解、ｐ５０仕様調査

□ 無理な短納期･低コスト

回避策 ⇒ 参照：ｐ３２【身勝手な顧客に対する防御策】、
ｐ６７【見積りリスクの回避策】、ｐ１４３●獲得時間の最大化、等

これらのリスクの解消法は、「仕様の早期凍結」および「妥当な納期･開発費の獲得」ですが、これを実現するためには、それ相応の仕様知識、技術力および交渉力が必要になり、その開発組織の能力レベルが上がるに従ってその成果も拡大していきます。

計画

◎プロジェクト問題の起点である要求仕様の品質は、ソフトウェアの品質と量を規定する。

【必要時間と許容時間】（図３－１－１）

67 **ベンダー**（vendor） ソフトウェアないしはハードウェア製品のメーカーまたは販売会社のこと。ユーザー企業から発注されるソフトウェア開発の元請け企業となる場合が多い。

計画

早期の仕様凍結

早期の仕様凍結のチェックポイントは次のようになります。

◆【早期仕様凍結のチェックポイント】

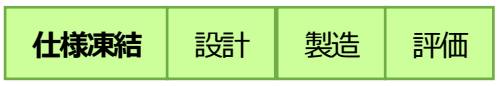
- □ 早期仕様凍結のために、顧客および関連各社の参加・協力の要請を行うこと。
- □ 仕様凍結の期限を切り、発注顧客側と受注開発側で合意をしておくこと。
- □ 顧客・開発会社間の直接コミュニケーションによる、要求仕様の期限内凍結を行うこと。
- □ 集中検討会ないしは合宿等にて、短期集中的に仕様決定を行うこと。
- □ 受注側においても提案型仕様凍結を行うこと。
- □ 顧客価値[68]の優先度順に、仕様凍結を行うこと。
- □ 顧客価値の高い仕様順に、開発着手すること。
- □ 開発仕様の目的・背景・範囲・内容を文書にて明確化すること。
- □ 基幹仕様未凍結状態ないしは疑問点・不明点を残したままで、先行開発着手は行わないこと。

計画

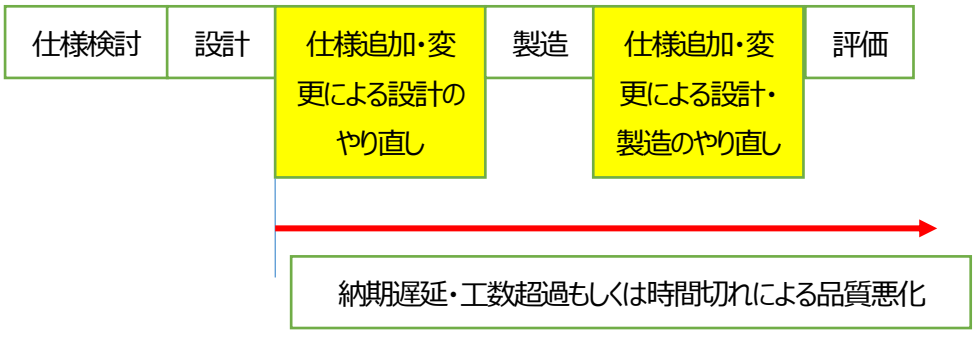
◎基本的な要求仕様は、設計着手前に凍結されなければならない。

【決まらない要求仕様がもたらす弊害】（図３－１－２）

<決まる仕様>



<決まらない仕様>



68 **顧客価値** 顧客がビジネス上重要視している機能やシステムのこと。

先行開発着手の得失について

要求仕様の決定が遅れ始めると、開発スケジュールを遅延させないために早期の仕様凍結をあきらめて先行着手[69]を始めるプロジェクトが後をたちません。それも開発の基幹仕様が決まっていないのにもかかわらず、先行着手に走ってしまっているのです。枝葉に相当するような仕様の先行開発ならば、後でやり直すとしても被害は微少で済みますが、基幹仕様の場合は、高い確率で大きなやり直しが発生します。
先行着手を行う開発者たちは、何をやるのかが決まってもいない仕様を、一体どうやって開発するのでしょうか。この場合、先行着手を別の言葉で言えば想定開発[70]ということになります。

早期の仕様凍結をあきらめた開発組織は、いつしか事前着手が当たり前の行為になってしまうだけではなく、二転三転し開発工程の後半になっても決まらない顧客の要求仕様に振り回されることになってしまいます。

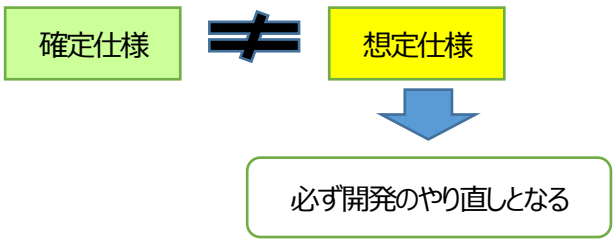
事前開発という想定開発によるやり直しに加えて、いつまでも決まらない要求仕様によるやり直しまでも背負うことになったプロジェクトは、必ずＱＣＤに大きな傷を負うことになります。先行着手で稼いだと思ったわずかな時間など何の効果もなかったことになります。

基幹仕様未凍結での開発着手を禁止するということは、非合理的な想定開発を禁止すると同時に早期の仕様凍結に全力で取り組まなければいけないということを意味しているのです。

この問題は、仕様決定の当事者になりにくい下請けのプロマネや開発リーダーをより深刻な状況に追い込むことになります。しかしこのような状態に追い込まれないためには、下請けのマネジメントは元請けのマネジメントに対して、顧客との仕様検討の場に下請けのプロマネや開発リーダーの参加を要請する必要があります。

早期仕様凍結を本当に実現したいと思うならば、元請け側がこの提案を断る理由はどこにもないでしょう。

【確定仕様と想定仕様】（図３－１－３）



---

69 **先行着手** 仕様未凍結状態で、開発側の想定仕様に基づいて開発を始めること。

70 **想定開発** 確定仕様に基づかず、自分の想定仕様による開発。多くの場合、やり直しが発生する。

計画

仕様の理解

優れた開発者における開発工程ごとの時間の使い方は、仕様の調査・検討である初期工程に重く、後になるに従って軽くなっています。さらに開発すべき対象をどれだけ正確に把握しているか、どれだけ早く理解しているか、がプロジェクトの成否の分かれ目になります。この勝負は見積り時および要件定義工程においてほぼ決定しています。プロジェクトを成功させる要因の重さは、"何を作るのか"が分かることで六割、"どのように作るのか"が分かることで三割、その他要因で一割程度だと思われます。

これらの事実を直感的に理解している開発者は、見積り時において、仕様の全体像の把握および主要な仕様の把握に全力を上げています。これができれば妥当な見積りが可能となり、基本設計の概要を描くことも可能になります。要するに、進むべき地図を先に明確に描いておくのか、それとも迷いながら進むべき道を探すのかの違いです。どちらが早く、しかも正確に目標に到達できるかは一目瞭然です。

計画

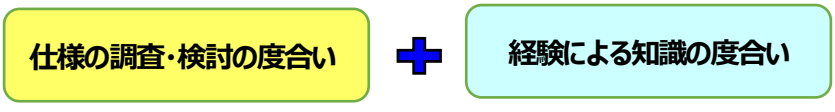
◎仕様の全体像および主要仕様の把握は、設計工程の前に済ませておくこと。

仕様凍結にあたってのチェックポイントは次のようになります。

◆【仕様凍結のチェックポイント】

- □ まずは仕様の全体像の把握から始めること。
- □ 要求仕様の背景や意味を必ず理解しておくこと。
- □ 仕様検討の段階で要求者と徹底的な仕様検討を行うこと。
- □ 疑問・不明点の発掘を行い、その解消に向けて、要求者に対し積極的な行動を取ること。
- □ 仕様決定のQ＆A[71]は直接対話による確認を行うこと。
- □ 不明なことは直ちに分かっている人・部署に聞くこと。
- □ 仕様検討にて新たに知り得た仕様や技術情報を、ドキュメント[72]によって他のメンバーに伝えること。
- □ 早期の仕様凍結を行うこと。
- □ 基幹仕様未決定で開発に着手しないこと。

【仕様の理解度】（図３－１－４）



[71] **Q＆A** Question＆Answer 質疑応答の略。ここでは仕様の不明点および疑問点に関する質疑応答のこと。

[72] **ドキュメント** 仕様書や設計書などに限らず管理用文書、ビジネス書類、メモなども含む全ての文書のこと。

仕様調査

要求仕様には必ず疑問点や不明点が多く含まれています。これらの問題をすべてクリアにしなければ間違いのない仕様を決定することはできません。仕様調査にあたってのチェックポイントは次のようになります。

◆【仕様調査[73]のチェックポイント】

- □ 仕様骨子の事前の把握と整理を済ませておくこと。
- □ 仕様調査は、顧客価値の重要度順および基幹的仕様から始めること。
- □ 重要な仕様・基幹仕様に関するＱ＆Ａは、直接対面コミュニケーションで確認すること。
- □ 仕様追加・変更の影響度[74]の事前調査を済ませておくこと。
- □ 全ての問題点・疑問点を掘り起こすこと。
- □ 仕様内容および変更内容は、チーム内（含む評価チーム）で即時的な情報共有を行うこと。
- □ 仕様調査のＱ＆Ａ情報や変更内容は、要求仕様書や設計書も同時に更新すること。
- □ 仕様決定事項や変更事項は、集中的かつ情報共有可能なシステムで管理すること。

なぜそんなに大人しいの？

仕様決定の遅延で何度もその後の工程が崩れた経験をしているのに、被害を受けている開発者たちは、なぜか大人しいのです。愚痴を言いつつ、ただ頑張るだけで良いのでしょうか。仕様提示者にクレームをつける勇気も気力もないのでしょうか。論理的な仕事を遂行しているはずなのに、一向に合理的な行動を起こせないような状態に何故なってしまったのか、それをどうしたら良いのか、もう一度自分自身で真剣に考える必要があるでしょう。

計画

◎仕様に関する疑問点・不明点は、設計着手前に完全に解消しなければならない。

【仕様の残存疑問点・不明点】（図３－１－５）

---

[73] **仕様調査** 要求仕様の疑問点や不明点を調査すること。この作業の質が、見積りの精度に大きく影響する。

[74] **影響度** 仕様変更の影響が間接的に及ぶ他のソフトウェア部分を特定すること。仕様変更影響度表により管理される。

要求仕様書の記述項目

要求仕様書の精度レベルを一定以上に保つためには、標準的な統一フォーマット[75]を使用し、記述必須項目および記述の粒度を指定した、要求仕様書の作成ガイドライン[76]が必要です。

要求仕様書の精度評価にあたっては、必須項目の記述の有無および記述内容のレベルをチェックし、見積りおよび設計を可能にするための条件をどの程度満たしているのかを評価する必要があります。

以下に要求仕様書の記述項目例を示します。

【要求仕様書の記述項目例】

1．システム企画　：要求仕様の意図・目的・背景が記述されていること。
2．システム概要　：システム構成図、ソフトウェア構成、ハードウェア構成、コード体系[77]等の明示。
3．要求仕様概要　：全ての要求機能を漏らさず、機能ごとに端的に記述されていること。
4．システム運用フロー[78]　：顧客側におけるシステムの業務運用フローが記述されていること。
5．要求仕様の詳細説明
　① 要求仕様詳細　：開発項目ごとに要求仕様の詳細説明が論理的に記述されていること。
　② 入出力情報の定義　：入力画面／出力画面・出力リスト等が論理的に示されていること。
　③ 内部データ処理の定義：ビジネスロジック[79]のインプット・プロセスロジック・アウトプット条件等の明示。
　④ 外部通信の定義　：送・受信データの定義、通信手順・通信フォーマットの定義等が明確であること。
6．非機能[80]説明
　① データ要件の定義（件数の最大・最小値、保有期間等）が明確であること。
　② 性能要件の定義（レスポンス[81]・パフォーマンス[82]の規定）が明確であること。
7．条件・制限事項　：運用条件、制限、信頼性事項、拡張性事項、セキュリティ事項の明示。
8．要求スケジュール　：開発開始・完了日、成果物出荷日、市場展開日程等の明示。
9．その他　：既存システム構成、既存システムからのデータ移行[83]等の情報および条件の明示。

要求仕様書記述項目表のサンプルの一部は次の通りです。

---

75 **フォーマット**　書式および構成があらかじめ設定されている文書のこと。
76 **ガイドライン**　あるものごとにおいて実行すべき指針を示したドキュメント。
77 **コード体系**　JAN・EAN・UPC などの商品コード。
78 **運用フロー**　客先におけるシステムの業務運用の流れを示したドキュメント。
79 **ビジネスロジック**　アプリケーション機能における処理方法の論理のこと。
80 **非機能**　非機能要件とは，性能や信頼性・拡張性・セキュリティなど，機能要件以外のもの全般を指す。
81 **レスポンス**　ソフトウェアの応答速度。たとえば、あるキーを押してから表示が行われるまでにかかる時間など。
82 **パフォーマンス**　あるソフトウェアに起動をかけてから全ての処理が完了するまでにかかる時間。処理速度とも言う。
83 **データ移行**　旧システムで使用していたデータを新システムに乗せ換えること。

計画

【要求仕様書　記述項目】（表３－１－１）

			開発要件 必要項目	見積要件 必要項目	今回提出 の項目
表紙					
変更履歴					
１．システム企画・目的				＊	
２．システム概要	２．１　システム構成図	２．１．１　全体構成図		＊	
		２．１．２　店舗機器構成図		＊	
		２．１．３　ﾈｯﾄﾜｰｸ図			
	２．２　ハードウェア構成				
	２．３　ソフトウェア構成				
	２．４　コード体系				
３．要求機能概要	３．１			＊	
（要求機能を漏らさず箇条	３．２			＊	
書きにする）	３．３			＊	
	３．４			＊	
	３．５			＊	
４．運用フロー					
５．要求機能説明	５．１　要求機能詳細	５．１．１			
	（開発項目毎に要求仕様の詳	５．１．２			
	しい説明をする。）	５．１．３			
		５．１．４			
		５．１．５			
	５．２　入出力情報	５．２．１　入力画面			
		５．２．２　入力モニター			
		５．２．３　出力画面			
		５．２．４　出力リスト			
		５．２．５　レシート			
	５．３　その他	５．３．１　集信データ			
		５．３．２　配信データ			
		５．３．３　オンラインフォーマット			
		５．３．４　メモリーバランス			
６．非機能説明	６．１　データ要件	６．１．１　データ件数		＊	
		６．１．２　データ保有期間		＊	
	６．２　要求性能			＊	
７．条件・制限事項	７．１　運用条件			＊	
	７．２　制限事項			＊	
	７．３　信頼性				
	７．４　拡張性				
	７．５　セキュリティ				
８．概略スケジュール	８．１　稼働スケジュール				
	８．２　展開スケジュール				
９．その他	９．１　従来システム構成				
	９．２　従来システムからの データ移行				

計画

◎過不足ない要求仕様書は、精度の高い見積りとスムーズな開発の基盤となる。

要求仕様書精度の簡易チェック法

要求仕様書の簡易的なチェック方法について説明します。

あいまいな要求仕様書であるか否かは、その要求仕様書を使って一部の重要な仕様部分について基本設計書が作成できるかどうかを検討してみることです。もし基本設計書の重要な部分が作成できないとしたら、それは要求仕様書に欠陥があると言うことになります。ほとんど基本設計書が書けないとすれば、その要求仕様書は全く使えないレベルのドキュメントだと言ってもかまわないでしょう。

また要求仕様書と対をなしている仕様書としては、総合テストにおいて使用される運用テスト仕様書があります。運用テスト仕様書は、開発されたソフトウェアが要求された通りに運用に沿って動作するかどうかをテストするためのドキュメントです。要求仕様書の一部分について、総合テストの担当者に運用テスト仕様書が書けるかどうかを試してみても要求仕様書の精度を確認できるでしょう。

要求仕様書の詳細チェックリスト

Check Timing：要求仕様書発行・受領時

要求仕様書精度を詳細に渡ってチェックしたい場合には、下記のチェックリストを使用すると良いでしょう。

要求仕様書になじみの薄い初級プロマネの人においては、実際の要求仕様書を数点選び、下記のチェックを行って見ればイメージをつかむことができるかと思います。

◆① 妥当であること

- □ 顧客やユーザーのニーズと一致していること。
- □ 上位のシステム要求仕様書などの関連する他のドキュメントとの矛盾がないこと。
- □ 未確定項目がある場合は、どのように合意するか、依頼者と合意形成方法を決めておく。

◆② あいまいでないこと

- □ 要求仕様書に記述されている要求が、ただ一通りに解釈できること。
- □ 要求仕様書の“良し悪し”を判断する手段や基準をもつこと。
- □ 「範囲」を読み取れるように要求を表現すること。
- □ 仕様は「仕様である」ことを明示し、説明は「説明である」ことを明示して記述すること。
- □ 要求仕様書では、記述内容が“特定”できる表現になっているものを“仕様”とすること。
- □ 要求仕様書の構成や内容は、後工程の読者に分かるように書くこと。
- □ 「等」や「ｅｔｃ」の文言は使用しない。使用する場合は、○月○日までに決めるとコメントをつける。

◆③ 完全であること

- □ 顧客やユーザーの情報システムに対するニーズ[84]が、漏れなく要求仕様書に記述されており、かつ図表の参照や用語の定義などの要求仕様書の形式が整っていること。
- □ 「境界」は早い段階で決めること。
- □ 「要求」のモレを防ぐために、カテゴリの分類や要求の分割・階層化に漏れがない、隙間がないことを確認できるようにすること。
- □ 要求仕様書には、「操作性」「保守性」「交換性」などの「品質要求」を記載すること。
- □ 階層化の基準として、以下を（状況によっては組み合わせて）使い、「隙間」なく分割すること。
  ⇒ 時系列分割（時間軸分割）／構成分割／状態分割／共通分割など。
- □ モレなく書くこと。
- □ 要求仕様の番号をテストケースの番号とひもづけし、テストケースにモレがないことを確認すること。
- □ 仕様をグループに分け、さらに集合を小さくし、混じり気のない仕様のグループを作る。
- □ ＜グループ名＞に要求の性質を持たせるためには、範囲を表したグループ名を選ぶこと。
- □ 「・・・・は、・・・・しない」という「否定表現」を避け、ｔｈｅｎとｅｌｓｅの両方を明らかにすること。

◆④ 矛盾がないこと

- □ 要求仕様書内部で矛盾や衝突がないこと。
- □ ほかの機能の仕様と衝突していることに気づくためにも、仕様は早期に展開すること。
- □ 早い段階で全体の仕様化を行うこと。

◆⑤ 重要度と安定度のランク付けがされていること

- □ 各要求について、重要度と安定度[85]を示す指標を明確につけておくこと。
- □ 確認中の仕様をそのまま記述し、変わる可能性があることを明記すること。

◆⑥ 検証可能であること

- □ 開発されたソフトウェアが、要求仕様書に記述された要求を満たしているかどうか確認可能であること。
- □ 検査部門の人に、「検査可能」という側面から要求仕様書のレビューを実施してもらう。
- □ 品質要求（「操作性」「保守性」「交換性」など）はテストでも確認すること。

---

84 **ニーズ** 要求ないしは要望のこと。顧客のニーズが漏れなく要求仕様書に記述されていることを確認する手段としては、顧客との仕様打ち合わせ議事録や顧客から提供される提案依頼書（ＲＦＰ：Request For Proposal）および顧客への提案書などがある。

85 **安定度** 仕様変更の可能性の高さ低さのことを仕様の安定度と言う。

計画

◆⑦ 変更が容易であること

□ 要求仕様書に対する変更が、容易に、完全に、一貫して行えるようになっていること。

ａ）目次や索引、明確な相互参照が整備され、使いやすい構造になっていること。

ｂ）冗長でない、つまり、同じ要求が要求仕様書内で複数個所に記述されていないこと。

ｃ）他の要求と混ざらず、各要求を独立・分離して表現して、互いに依存していないこと。

□ 重複なく書くこと。

□ 仕様書全体を「均一」に記述することにこだわらないこと。関係者間で共有できている認定仕様[86]まで、詳細に記載しなくてもよい。

□ 仕様番号の確定作業は、仕様化の最初の段階では行わないこと。グループ分け確定後に行うこと。

□ 似た記述が続く場合に、何が違うかをすぐに読み取れるようにすること。

◆⑧ 追跡可能であること

□ 要求仕様書に記述された個々の要求に関し、その起源が明確であり、開発が進行するに伴って作成された文書等との対応付けがとれること。

ａ）後方追跡可能性があること。

ｂ）前方追跡可能性があること。

□ 設計や実装の工程で明らかになった「仕様」は、要求仕様書に書き戻すこと。

□ 「要求」と「理由」をセットで表現すること。

□ 要求仕様には固有の記番号を付けること。

（参考資料：　ＩＥＥＥ８３０品質特性、ＵＳＤＭ）

計画

◎要求仕様書のチェックポイント

１．妥当であること　２．あいまいでないこと　３．完全であること　４．矛盾がないこと

５．重要度と安定度のランク付けがされていること　６．検証可能であること

７．変更が容易であること　８．追跡可能であること

---

86 **認定仕様**　その顧客において過去に十分な稼働実績がある、いわゆる枯れた仕様のこと。

要求仕様書の精度検証

要求仕様書の精度が悪い場合、正確な見積りもできず、さらには設計すら不可能になってきます。このような事態を避けるために、要求仕様書を出す側および受け取る側の両方において要求仕様の精度を常にチェックする必要があります。要求仕様書の精度低下の問題点を改善するために、要求仕様書を作成する側・受領する側双方において以下のような対策を実行する必要があります。 要求仕様書を作成する側（顧客ないしは開発会社側の代行部門）においての実行が困難な場合は、開発部門においてだけでも実行する必要があります。

要求仕様書の精度確保のためには次の２項目の実行が有効です。

① 開発要件を過不足なく網羅した要求仕様書ひな形の作成および運用

② 要求仕様書の採点評価

要求仕様書の完成度評価のための採点表のサンプルを下記に示します。

【要求仕様書　採点表】（表３－１－２）

ユーザー名：ＸＸＸＸ　　開発担当部署：ＸＸＸＸ

プロジェクト名：○○○○　　採点者：ＸＸＸＸ

０：　記述なし
１：　仕様としては不十分
０：なし　２：　多少不明点はあるが内容はわか　＊記載が８０点未満の場合、基本的には見積不可とする。
１：あり　３：　要件として十分　＊記述内容得点が６０点未満の場合、基本的には見積不可とする。

項番	分類	大項目	中項目	要求仕様内容	記述必須項目	記述の有無	記述内容得点	備考
1		1	1	表紙・変更履歴・目次	1	1	3	
2	目的	2	2-1	システム化の理由	1	1	3	
3			2-2	システム化前後の運用	1	1	3	
4			2-3	システム化のメリット	1	1	3	
5	概要	3	3	システム構成図	1	1	3	
6		4	4	ハード構成				
7		5	5	ソフト構成	1	1	2	
8		6	6	運用一覧				
9	運用	7	7	運用スケジュール	1	1	3	
10		8	8	概要ビジネスフロー（運用フロー）				
11	概要	9	9	業務一覧	1	1	3	
12		10	10	ビジネスルール定義				
13	機能	11	11	詳細ビジネスフロー				
14		12	12	画面一覧				
15		13	13	画面設計		0		
16		14	14	画面項目説明				
17		15	15	帳票一覧				
18		16	16	帳票設計				
19		17	17	帳票項目説明				
20		18	18	機能定義書	1	x	x	
21		19	19	画面遷移図				
22		20	20	要求データ一覧				
23		21	21	要求データ仕様書	1	1	2	
24		22	22	オンライン一覧	1	1	2	
25		23	23	要求性能	1	1	2	
26		24	24	その他				
				合計	12	11	29	

	記載の有無		記述内容得点
満点ポイント	12		36
得点ポイント	11		29
得点（１００点満点換算）	92		81

記述内容満点
＝12×3
＝36点

記載率

記述内容得点
＝（29／36）×100＝81点

本表の拡大版は、「付録図表５．要求仕様書　採点表」を参照のこと。

3－2 要求仕様に関するリスク【リスクマネジメント】

要求仕様に関するリスク管理表の一例を次に示します。

【リスク管理表（要件定義工程）】（表３－２－１）

	プロジェクト名：				初版発行日： 更新日：		文書番号：
	要因分類	QCD影響 Q	C	D	要件定義工程リスク	✓	チェック 検出日
ヒトに関するリスク	①他者依存的姿勢（自律性の放棄）	○	○	○	・顧客の動き待ちの姿勢の結果、納期遅延		
	②上位マネジメントの関与不足		◎		・顧客交渉戦略・能力不足		
					・大幅な仕様追加要求の丸呑み		
					・大幅な開発費削減要求の丸呑み		
					・大幅な納期短縮要求の丸呑み		
	③ユーザーの参加・協力度不足	◎	○	○	・顧客の参加・協力度が低い		
	④組織能力不足（未熟な組織文化、戦略の欠如）	◎	◎	◎	・見積りプロセスの手続きルール違反		
	⑤見積り能力	◎	◎	◎	・既存資産流用の可否判断		
					・顧客要件の精査		
					・フィット＆ギャップ調査能力		
	⑥要件定義能力	◎	○	○	・要件定義能力不足		
	⑦リーダーのプロジェクトマネジメント能力（外部交渉、タイムマネジメント、現場主義、見える化能力など）	◎	◎	◎	・顧客交渉戦略・能力不足		
					・大幅な仕様追加要求		
					・大幅な開発費削減要求		
					・大幅な納期短縮要求		
					・適時の支援依頼発信能力不足		
					・変更管理未実施による開発費の増加		
					・顧客に対する不都合な事実の隠蔽（故意による事故）		
					・要件定義体制不備（業務経験・知識不足の要員で要件定義を実施）		
					・要件定義遅延のため不十分なテストで客先リリース		
					・顧客要求の優先順位コントロール失敗による開発失敗		
					・タイムリミット管理能力不足		
	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）	◎	○	○	・新技術知識不足		
					・顧客業務知識不足		
	⑨プロセス管理の有無	◎	○	○	・プロセス管理の不足		
					・仕様凍結未確認のまま設計着手		
					・基本設計の承認なしで詳細設計着手		
	⑩コミュニケーション能力（阻害・ギャップ）	◎	○	○	・顧客のパートナー化失敗		
					・顧客とのコミュニケーション能力		
					・プロマネと上司間のコミュニケーション		
					・営業と開発部門間のコミュニケーション能力		
					・営業部との交渉力		
	⑪関連部署との連携不足	◎	○	○	・組織間の協調性・コミュニケーション		
モノ	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）	◎	○	○	・要求仕様書なし		
	⑬開発のベースの有無	◎	◎	◎			
	⑭開発環境の不備	◎	○	○			
カネ	⑮開発費不足	◎	◎	◎			
	⑯資源投入戦略の誤り（開発費投入時期ミス）	◎	○	◎			
情報	⑰あいまいな開発範囲（スコープ）	◎	◎	◎	・あいまいなスコープ		
	⑱あいまいな要求仕様	◎	◎	◎	・あいまいな要件		
					・あいまいな仕様		
	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）	◎	○	○	・情報の不備・不足		

本表の全体図は、「付録図表６． 要件定義工程のリスク管理表」を参照のこと。

3－3 要求仕様の構造化（ＷＢＳ　Work Breakdown Structure）

【スコープマネジメント】

要求仕様の確定後に必要なことは、要求仕様の構造化です。構造物とは、ある目的を果たすために複数の部品や部材を組合せて作られた物のことを指しますが、ソフトウェアも間違いなく構造物の一つです。

構造化とは、どの部品を、どのように組み立てるのかということを考える作業であり、ソフトウェア開発においては要求仕様書を元に、どのようなソフトウェア部品を作り、その部品をどの様に組み立てるのかを検討し、視覚化したものが作業細分化構造書（ＷＢＳ[87]）と言われるものです。

ＷＢＳは、各要求を実現するために必要なタスク[88]、成果物、担当者を全て定義する必要があります。その意味で、ＷＢＳはプロジェクトにて実現しなければならない開発範囲（スコープ）を完全に網羅した、スコープ定義書だとも言えます。

ＷＢＳの特徴ないしは要件を整理すると次のようになります。

- 顧客要求の全てを網羅している。
- 成果物と作業の二つの要素で構成される。
- 機能・作業の細分化の階層（レベル）を持っている。
- 最下位のレベルは、機能・作業の最小単位（ワークパッケージ[89]）で構成される。

ＷＢＳ内の各レベルの特徴

レベル１：プロジェクトの最終成果（納品されるシステム名等）
レベル２：成果物一覧の大項目（要件定義書、設計書等の開発フェーズを代表する成果物や作業）
レベル３：成果物一覧の中項目（大項目に属する成果物やタスク等）
レベル４：成果物の最小単位（最小単位の部品やタスクの作業手順等）

＊複雑なシステムにおいては、レベル３の下に、更にレベル４として小項目を設けることもあります。

ＷＢＳのレベルマップおよび概要は次の通りです。

---

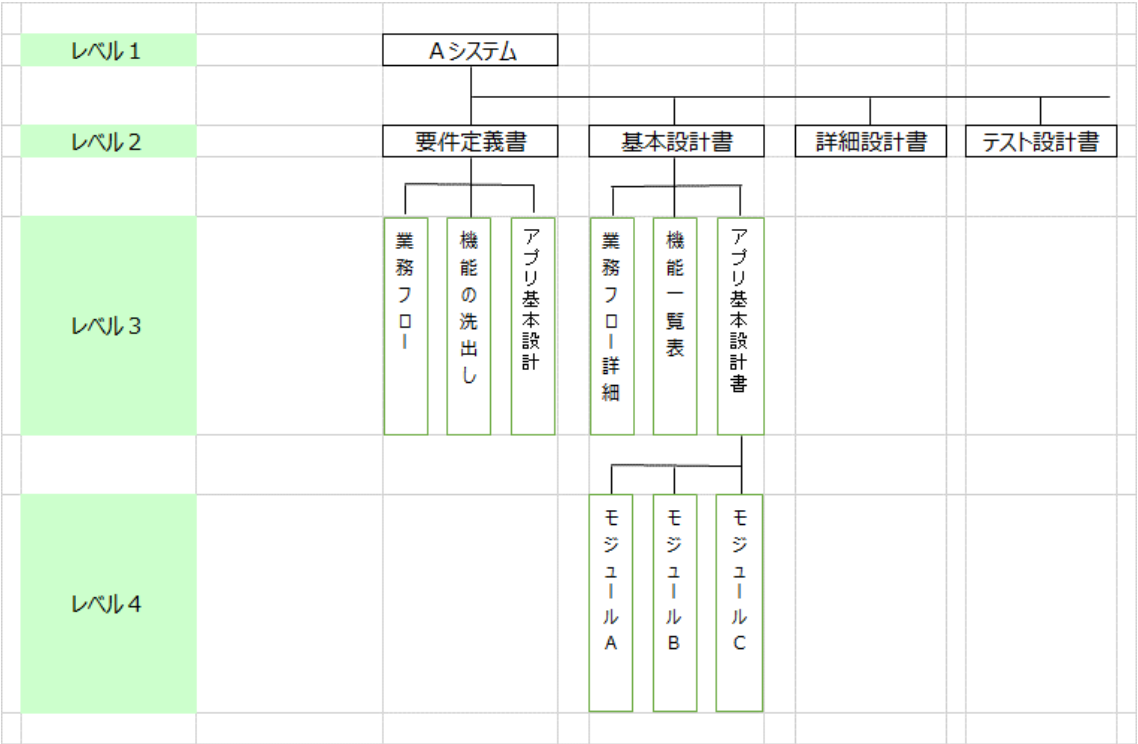
87 **ＷＢＳ**（Work Breakdown Structure、作業細分化構造書）　要求仕様の全てを細分化とともにツリー構造化したもの。最上位は納品されるシステム名等で、順に成果物の大項目・中項目・小項目に細分化される。最小単位はワークパッケージと呼ばれる。

88 **タスク**　アプリケーションソフトにおける最小の業務単位。コンピュータが処理する仕事の最小単位。

89 **ワークパッケージ**　ＷＢＳ（作業細分化構造書）の最小単位の仕様のこと。

計画

【WBS（Work Breakdown Structure）レベルマップ】（図3－3－1）



【WBS概要】（表3－3－1）

レベル1：Aシステム									
レベル2		レベル3		レベル4					
項番	成果物	項番	成果物	項番	成果物	工数	開始日	終了日	担当者名
1	基本設計書	1－1	アプリケーション基本設計書	1-1-1	モジュールA設計書				
				1-1-2	モジュールB設計書				
				1-1-3	モジュールC設計書				
・	・	・	・	・	・	・	・	・	・
・	・	・	・	・	・	・	・	・	・
・	・	・	・	・	・	・	・	・	・

計画

3－4 顧客要求の重要度順位の設定【スコープマネジメント】

基本的な仕様の確定およびＷＢＳの作成によって開発すべき顧客要求事項の内容はすべて明確に定義されました。続いて重要な作業として、プロジェクトの限られた時間と開発費を最大限有効に使うために、顧客要求事項の開発の優先順位を決めておく必要があります。

要求事項はすべて顧客の要求に基づいていますが、すべてが同じ顧客価値をもっているわけではありません。どの仕様がより顧客にとって重要かということは、要求仕様を決定する過程における顧客とのコミュニケーションの中で判断することができます。 重要度の設定作業は、まず複数の要求事項を顧客価値の重要度の順に、例えばＡ／Ｂ／Ｃの三つのグループに分けることから始めます。つまり開発の優先順位はＡ＞Ｂ＞Ｃのグループの順となります。 続いて各グループ内における優先順位の決め方として、Ａグループに属する仕様として例えばａ、ｂ、ｃが有った場合、ａを先に開発しなければｂもｃも動作できない場合は当然のことにａから着手することになります。このａをロードブロック[90]仕様（機能）と呼ぶことにします。同一の顧客価値内での優先順位はロードブロック仕様が優先権をもつことになります。ロードブロックの順に並べたものがプロセスの順になって来ます。 このように優先順位は、価値の順とプロセスの順という二つの意味合いを同時に判断する必要があります。

計画

◎優先順位の判断基準は次の二つ

１．顧客価値の重要度順

２．ロードブロック仕様の順

開発プロジェクトに起こりがちな優先順位無視の行動として以下のものがあり、プロマネは開発の全工程においてこれらの問題が発生しないように対策を行う必要があります。

【優先順位を無視した不正な開発行動】

- 仕様未決定状態における事前開発着手
- 要件定義書未完での基本設計の着手
- 基本設計未完での詳細設計の着手
- 詳細設計未完でのコーディング着手
- コーディング未完での評価テスト着手
- 総合評価未完でのソフトウェアのリリース[91]

---

90 **ロードブロック** 元の意味は道を塞いでいる障害物のこと。先に進むためにはその障害物を片付ける必要があると言うような文脈で使用される言葉。

91 **リリース** 完成したソフトウェアおよび関連成果物を顧客へ引き渡すこと。

計画

これらの優先順位を無視した行動の主な原因は、開発時間および予算の不足にあります。時間が不足した結果、あせって事前着手に走り、その結果多くの手戻り作業を発生させ、また複数の仕事を同時に処理できず不十分な作業内容による全体的な品質低下を招くことになります。

妥当な優先順位を決定するためには、その前に見積り交渉などにおいて妥当な開発期間および開発費の獲得が必須条件となります。

次に開発作業におけるものごとの優先順位の決め方を下記に示します。

【開発作業における優先順位の決め方】

- 仕事の意味・意図・背景を仕事の着手前に把握すること。
- 仕事内容をブレークダウンすること。
- ブレークダウンした仕事内容に優先順位を設定すること。
- タイムリミットが直近に迫っているものから処理をすること（緊急度優先）。
- 時間的余裕があるものは重要機能の順に処理すること。
- 同等の優先順位のものは、効果が高く実行が容易なものの順で、効果が低く難易なものは最後に。

＊ 優先順位が判断できない場合は早めに顧客や上長に相談すること。

＊ この仕事に許される自分の残り時間を毎日意識すること。

作業の優先順位の決定にあたっては、次の三つの視点での検討が必要になります。

【優先順位決定に必要な３つの視点】

１．一つの作業内における処理の順番の妥当性の視点

２．複数の作業の緊急度・重要度の優先順位判断の視点

３．手持ち時間と必要時間の比較判断の視点

この三つの視点における判断ミスおよび必要時間の確保の失敗は、ＱＣＤを大きく毀損する結果を招くことになります。

また、優先順位の誤った認識としては次のようなものがあります。

【開発作業における優先順位を誤らせる原因】

- 割り込み作業等の優先度判断の誤り
- 必須業務に優先順位はないという認識の欠如
- 納期第一優先の誤り（ＱＣＤはみな等しい優先価値をもっている）

計画

【優先順位の原則】

顧客の要求は機能単位に分割され、顧客価値の優先順位の高い順に開発される。

3－5 見積り【コストマネジメント】＆【タイムマネジメント】

見積りは、要件定義およびチームの技術能力とならんでプロジェクトを成功に導く三大要因の一つです。逆の表現をすると、見積り・要件定義・チームの技術能力はプロジェクトを失敗させる三大リスクとなっています。

見積り金額と開発期間（着手可能時期、開発完了時期）は、先に述べたＷＢＳの最小レベルの部材に要する金額と期間の総和として算出されることになりますが、見積り回答書を作成するにあたってはさまざまなリスクを考慮する必要があります。

ソフトウェアの価格

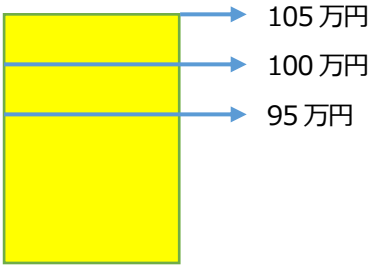
ソフトウエアの価格、特にカスタムソフト[92]の派生開発においては定価がないのが実状です。なぜなら顧客の要望に従って作成され、一本毎に内容が異なるため、いわゆる一品料理となり、全く同じものがこの世の中には存在しません。そのために価格についていくらが正当な価格なのか判断し難いのが実状です。

例えば１００万円程度の開発物件の場合を考えて見ると、この見積りにおいて１０５万円の場合、１００万円の場合、と９５万円の場合の違いを明白に区別できるでしょうか。 １０５万円と９５万円では実に１０ポイントもの差がでてしまいます。 優秀なプロマネならば発注者側の状況と自社側の状況を深く勘案しつつ、適正な価格の落とし所を見極めなければいけません。元値の１００万円という見積り内容にちゃんとした根拠があり、相手に納得させる能力さえあれば利益率の５％向上はそれほど難しいことではありません。

計画

◎有能なプロマネは＋５％の見積り額を獲得する。

【見積り価格】（図３－５－１）



100 万円程度の物件で、
105／100／95 万円の違いを
明確に説明することは難しい。
１千万円・１億円でも同様。

92 **カスタムソフト** 顧客からの特別な要求仕様に基づいて開発されるソフトウェア。対語は市販品であるパッケージソフトウェア。

計画

見積り回答書に対する認識

最初に見積り回答という行為に対する認識をしっかりと持つ必要があります。見積り回答書とは、法的な拘束力をもつ契約書です。受注者は発注者と約束したものを完成させる義務があり、発注者は受注者と約束した対価を支払う義務があります。この相互義務の履行を法的に約束した書類が見積り回答書です。見積り回答書は、単なるドキュメントとは訳が違うという認識が必要です。

計画

◎見積り回答書は、契約書である。

受注者にとっての見積り回答書の重要性

受注者の大きな目的の一つは、その仕事によって利益を上げることです。見積り回答書においては、達成すべき仕事内容、その対価および実行期間について約束してしまいますので、見積りミスや仕事の失敗は直ちに赤字を招き、企業の存続を危うくする場合もあります。

さらに見積り回答書は開発チームの仕事の原点であり、その後の開発における人・モノ・カネ・時間を規定するものであり、すべての書類やドキュメントの中でも最重要な書類です。見積り回答書は、その会社や組織のすべての実力を映し出す鏡といっても過言ではありません。

見積り回答書の基本的な要件は以下の二つだけです。

① 分かっている内容についてのみ見積ること
② 分かっていない内容については、見積りに含んでいないことを明記すること

すなわち見積り回答は、開発の対象となる機能について事前に過不足なく定義された要求書に基づいて行い、想像や想定に関することがらを一切含めてはいけません。

計画

◎見積りは分かっているものだけを見積ること。

計画

見積り回答書の品質

見積り回答書の形式的なチェック

妥当な見積り回答を行うためには、最初に見積りにおける記述モレ・考慮モレなどの単純なミスを防ぐために、見積り回答書の形式的なチェックが必要になります。モレなどのミスにつながりやすい項目としては、次のようなものがあります。

◆【見積り回答書の形式的なチェックリスト】

- □ 見積り回答期限の事前確認は行ったか。
- □ 見積り対象の仕様は明確になっているか。
- □ 見積りにインプット条件は全て網羅したか。
- □ 見積りにアウトプット条件は全て網羅したか。
- □ 仕様の疑問点・不明点は全て顧客に確認したか。
- □ 仕様変更が及ぼす影響範囲は特定したか。
- □ 特別な見積り条件の要求があった場合、その考慮は行ったか。
- □ 見積り範囲やリスクに関する条件を見積り回答書に記述したか。
- □ 見積り回答書に記述した開発範囲以外のものは別途見積りとするという文章を記述したか。
- □ 開発着手時期および完了時期を記入したか。
- □ 見積り有効期限を記入したか。
- □ ＊その他、過去の自他における見積り失敗項目を記述すること。

上記のチェックリストに、みなさんのプロジェクトで起こりがちなミスを加えれば、更に良いチェックリストになるでしょう。またこれらのミスを出さないような、見積り回答書の統一的なフォーマットやガイドラインの整備も必要となります。

計画

◎見積りガイドライン・見積りチェックリストによる単純ミスの防止を。

見積り精度の向上

次に必要なことは、見積りの内容自体の精度向上です。見積りの精度が低いために、妥当な開発費に対して異常に高い回答をすれば、顧客側の怒りを買い信用を失います。反対に誤って低い金額で回答してしまった場合、プロジェクトを成功裏に完了させることは不可能になってしまいます。開発期間についても同様のことが言えます。

一定の見積り精度を確保するための要点を次のチェックリストにまとめました。

◆【見積精度向上のチェックリスト】

- □ 見積り対象の仕様知識に習熟していること。
- □ 見積り対象システムのプログラム構造を理解していること。
- □ 見積り前に要求仕様の事前調査を済ませていること。
- □ 既存の設計書等が不備な場合、影響範囲に絞ったソースコードの調査を行うこと。
- □ 事前調査の結果をドキュメントに残しておくこと。
- □ 見積り対象仕様における、過去の開発の失敗事例を把握しておくこと。
- □ 見積り対象仕様の開発に必要な技術を保有していること。
- □ 過去の類似開発の見積り／実績データを参考にした見積りを行うこと。
- □ 見積り範囲（開発システムのスコープ）を明確にすること。
- □ 見積条件を明示すること。
- □ ソフト性能[93]設計の根拠となる、適用Ｈ／Ｗ等の性能値条件を明示すること。
- □ 要求仕様の精度レベルによって見積リスク係数を設定すること（非公開とする）。
- □ 流用部分はブロック図で表現すること（顧客に説明しやすい）。
- □ 添付資料の充実を図ること（運用フロー・データフロー等）。
- □ 納期によるステップ分けは別資料とすること。

大方の見積り方法は、経験に基づく手法で行われているようですが、単なる経験的な直感での見積りでは、失敗の危険性が非常に大きくなります。経験に基づくと言っても、何らかの経験知、すなわちデータに基づく必要があります。見積りに関する経験知データとしては、過去のプロジェクトにおける見積り値と実績値のデータがあります。毎回、見積りと実績の差異およびその原因・理由についての記録を取り、その改善対策を行うことで、チームの能力は強化され、次なる見積りの精度ははるかに高いものになります。全てのプロジェクトにおいて、見積りおよび実績値の記録および差異比較の振り返りは必須です。

---

93 **ソフト性能** ソフトウェアの動作速度、特に応答速度（レスポンス）および業務処理速度（パフォーマンス）は重要なソフトウェアの性能要件とされる。

計画

正確な見積りをするためには、まず仕様理解力と設計能力が必要です。見積りは基本的に、何をどう作るかが分かっていればできます。つまり明確な要件定義とそれを実現する設計が正しくできれば、見積りは可能です。要求仕様が明確なのに、正確な見積りができないということは、自分の仕様理解力と設計能力が未熟だということになります。見積り手法はあくまでも道具にすぎず、その手法があれば見積りができるというものではありません。ソロバンがあっても使えなければ意味がないのと同じです。まず自分の設計技術スキル向上と経験の積み重ねが必要です。

見積りの精度向上のためには上記の要件を満たす必要があり、当事者において能力不足の部分については他者の支援を仰ぐ必要があります。

計画

◎正確な見積りには、仕様理解力と設計能力が必要。

見積り方式

見積り手法の主なものには、①経験的手法 、②ＬＯＣ（Line of Code）[94]法 、
③ＦＰ（ファンクション・ポイント）[95]法などがあります。

①の経験的手法は広く行われている方法ですが、見積りの精度を上げるためには、過去のプロジェクトにおける、見積り対実績の差およびその原因についての分析データの蓄積が必要です。
②のＬＯＣ法は、工数＝ステップ数÷開発生産性ｘ（１＋間接費要員比率）ｘ余裕率で求められます。
③のＦＰ法は、工数＝基準値ｘ（０．６５＋調整値÷１００）で求められます。

②③の手法は、その開発組織としての今までの実績のデータの積み重ねがなければ、計算式のパラメータ値を決めることができません。 見積りの精度は見積り方式に依存するというよりも、見積り者が見積り対象の仕様ないしは機能に精通している度合いに依存しています。すなわち対象仕様・機能に関する開発経験および知識に決定的に依存しています。

計画

◎見積りの精度は見積り方式に依存するよりも、
仕様知識と開発能力に強く依存する。

---

94 **ＬＯＣ**（Line of Code） ソフトウェアの規模を表す指標のひとつで、ソースコードの行数を意味する。

95 **ＦＰ**（ファンクション・ポイント）ソフトウェアの機能数および複雑度による重みづけをした点数の合計から開発工数を見積る方式。

計画

見積りにおけるリスク

【リスクマネジメント】

見積りに関する主なリスクは次の通りです。

【見積りリスク】

① 要件定義書内容の不備（必要事項の記述なし・記述ミス、不明点・疑問点が多いなど）
② 要件定義書なしの見積り依頼（口頭・電話のみによる依頼など）
③ いつまでも決まらず二転三転する要求仕様
④ 短時間・短期間での見積り回答要求
⑤ 口頭や概算ベースの回答が正式回答として扱われる
⑥ 最初から仕切り値で開発費や納期を要求してくる場合
⑦ 特定の依頼者による、頻繁な受注につながらない見積り依頼
⑧ 見積り者における仕様の理解不足や事前調査の不足
⑨ 見積り者における交渉力の弱さ

見積り回答におけるリスク回避には次のような対策が必要になります。

【見積りリスクの回避対策】

① 見積り可能な要求仕様の提示がない見積り依頼は、出し直しを要求する。
② 妥当な見積り期間が設けられていない見積り要求には、必要期間を要求する。
③ 過剰な開発費削減・納期短縮要求に対しては、合理的根拠の提示を要求し安易な妥協はしない。
④ 口頭での見積り依頼に対しては口頭で返し、口頭レベルの依頼・回答は一切正式なものとしては扱わず、責任も持たない旨を通告しておく。
⑤ 公式のルートおよび承認のない見積り依頼は、出し直しを要求する。
⑥ 受注ヒット率が低い見積り依頼を頻繁に出す部署ないしは個人へ事情聴取を行う。
⑦ 短納期リスク物件においては、分割見積り・分割開発・分割リリース等の交渉を行う。
⑧ 複数社開発体制プロジェクトにおいては、自社責任範囲を見積り回答書に明記する。
⑨ 不慣れな仕様に関しては経験者の知見を借りる。

計画

◎リスキーな見積り依頼者が最大の見積りリスクである。
誰からの見積り依頼なのかに気をつけること。

見積りリスク回避対策の実行に当たって

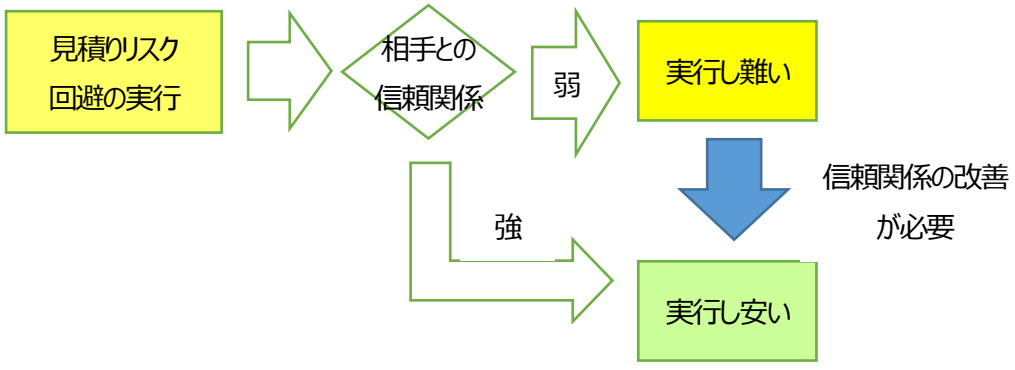
前記のリスク回避対策は、すぐに全てを実行することは無理なことでしょうが、これらのことは本来当たり前のことを当たり前に実行しようとしているだけのことです。

これらのリスク回避を行わなければ、まともなＱＣＤの達成はとてもできません。一つひとつの回避策の実行を重ねていくことが期待されます。良い製品を提供し、ＱＣＤを成立させるために必要ならば、一見無理だとか強硬に見えることに取り組まなければ、現状を打開することはできません。これらの施策を通すためには、まず良い品質の成果物を先に提供し続ける実績が必要で、良い製品を妥当な納期で出荷できた実績を積めば、見積り依頼部門も開発部門の要求に耳を傾けてくれるでしょう。

これらのリスク回避対策は、実際に開発組織において実行されたもので、架空の話ではないことを付け加えておきます。

そうは言っても、下請け開発会社が元請け開発会社に対して同様のリスク回避対策を行った場合、元請け担当者の怒りを買い、仕事を失うのではないのかという懸念の提起がありますが、元請けとの信頼関係が貧弱な場合はきっと怒りを買うことでしょう。そんな要求をする前に自分たちの開発品質をまともにしてください、と言われかねません。まずは自分たちの開発品質を合格レベルに向上させながら、これらの見積りリスク回避対策を順次実行していくことが順当なやり方でしょう。

【見積りリスク回避と相手との信頼関係】（図３－５－２）



概算見積り

正式見積りの前段階として、詳細仕様が未決定の状態で、概算見積りの要求を受ける場合がありますが、概算見積りは一見簡単なようですが、実はそう簡単なものではありません。概算＝いいかげん、では困ります。自分が精通している領域でなければ、概算見積りも詳細見積りもできないと言ってもよいでしょう。

調査をどの段階で見切るかと言う判断は難しいものだと思われます。自分における判断は、その見積り対象に対して、自分が保有する経験・知識および自分が利用できる他人の経験・知識の質と量によって決定されます。自分が利用できる経験・知識の質と量が大きい程、それによる判断は妥当性のあるものになります。

ある程度分かっているものならば、概算的見積りも可能でしょうが、利用できる経験・知識の総量（質）が少なければ、大雑把な見積りも困難になります。

細かく分析して工数や時間を見積もる詳細見積りは、いわゆる「定量的」見積りですが、大雑把に見当をつけるだけで良いのなら下記方法を試してみてください。

【概算見積りの方法】

① 要求仕様の全体像を、その目的・意味・背景を含めて把握すること。
② 自分で見積り可能なものとそうでないものを分ける。
③ 未経験項目および重要リスク項目をリストアップする。
④ それぞれに対して、必要工数を大雑把に３段階（大・中・小）または５段階に分けて記入する。
各段階に要する工数は、例えば、大＝１ヶ月、中＝２週間、小＝１週間等に決めておく。
重要リスクについても、もしそれが起きた場合についても同様に工数の大・小を記入しておく。
自分で決められないものについては経験者や上長の意見を聞く必要があります。
⑤ これらの仕事を何人で実行するかを想定しておく。
⑥ 非常に大雑把ですが、上記④の開発期間の合計を⑤の人数で割ったものが想定されるスケジュール期間になります。金額についても同様の方法で算出します。上記のやり方を、仕様検討工程・設計工程・製造工程・評価工程に分けて算出すれば、ある程度妥当な数字が出てくるでしょう。

そうは言っても、要求仕様の骨子も決まっていないような物件については、概算見積りすら不可能であり、このような場合には、要求者に対して早く要求仕様の骨子を決めるように促す必要があります。

計画

◎概算見積りはあくまでも参考値。
正式稟議承認には正式見積りを、の注意書きを忘れずに。

見積り回答書

一般的な見積り回答書に記載が必要な項目は次に示した通りです。

１．発行年月日
２．宛先、宛先（写し）
３．見積り回答書　：概算／正式　＊概算または正式を明示する。
４．承認印欄　：最終承認印欄・担当部長印欄・担当課長印欄
５．見積り依頼書番号欄、見積り回答書番号欄
６．開発名記述欄、ベース開発名記述欄
７．開発内容記述欄
　＊開発内容および範囲について漏れなく明示すること。
　＊見積りの根拠となる説明資料を添付すること。
８．開発条件記述欄
　＊ソフトウェア性能設計の根拠となる、適用ハードウェア等の性能条件を明示すること。
　＊要求内容不明等で見積りから除外した項目を明記しておくこと。
　＊開発内容記述欄に記述のないものは、別途見積りにする旨を明記すること。
９．開発費記述欄
　＊回答書表紙には開発費合計を記述し、主要項目ごとの明細開発費は添付資料とする。
１０．開発スケジュール記述欄
　□リードタイム[96]　□アクチュアル　＊リードタイムまたはアクチュアルスケジュールを明示する。
　＊アクチュアルスケジュールの回答は、着手可能年月日および開発完了年月日を明記する。
　＊リードタイムスケジュールの回答は、〇〇ヶ月の表記を行うこと。
１１．見積り担当者記述欄　：所属名・担当者氏名
１２．見積り有効期限年月日
１３．添付資料　□有：ｘｘ枚　□無

[96] **リードタイム**　開発に要する期間を示したもので、開発開始・終了等の時期は明示されていない。

3－6 スケジュールの作成【タイムマネジメント】

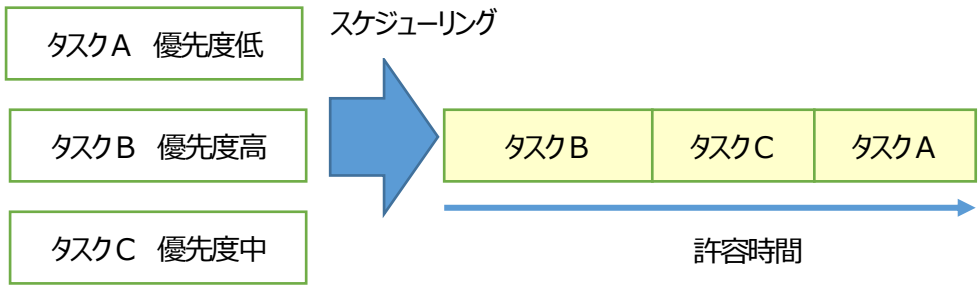
スケジュールの作成、すなわちスケジューリングとは、一言でいうと「持ち時間に仕事を割り振ること」です。やるべき仕事（タスク）をすべてブレークダウンし、それを持ち時間であるスケジュール表に投影させる事が、本来のスケジューリングです。 またスケジュール表は時間と仕事の相関関係を視覚化したものです。

スケジュールの調整が必要になった場合、先にスケジュール表の線引きをいじって、帳尻を合わせようとしてはいけません。 スケジュールの調整はまず仕事内容の見直しと、その裏付けである人員体制の調整から始めるべきで、その結果を新たに持ち時間に投影すべきです。また各タスク（仕事）間の関連性・継続性を意識することで、時系列的にバラバラになりがちなタスクを線として連続したものとして把握することも重要です。

計画

◎スケジュールの作成とはやるべき仕事（タスク）をすべてブレークダウンし、
それぞれをやるべき順番にプロジェクトの持ち時間に割り振って視覚化したもの。

【スケジューリング】（図３－６－１）



スケジュールの作成は、単純に言えばＷＢＳでブレークダウンした最小単位（レベル４）のタスクを時系列の順に並べることで作成されます。

例えば基本設計が、A・B・Cの三つのモジュールで構成されていた場合で、Aモジュールの設計に 2 週間、BおよびCは各１週間を要する場合の作業スケジュールは次のようになります。

【小日程表】（表３－６－１）

						作業日程		
項番	成果物	工数	開始日	終了日	担当者名	○月○日	△月△日	
1-1-1	Aモジュール設計書	2週間	○月○日	△月△日	a	→	→	
1-1-2	Bモジュール設計書	1週間	○月○日	△月△日	b	→		
1-1-3	Cモジュール設計書	1週間	○月○日	△月△日	c	→		

大日程および中日程表（ＷＢＳのレベル１～３）は、小日程表の積み上げ作業を行うことで作成されていきますが、顧客と合意済みの日程との食違いが発生しないように人員の投入数の調整や人材の入れ替えなどによる調整が必要になります。

一方、Aモジュールの設計が終わらなければBおよびCモジュールの設計に着手できない場合のスケジューリングは下記のようにすべきですが、日程が詰まっている場合に、何らの対策もないまま並列スケジューリングを無理やりしてしまうと、結局スケジュールは破綻してしまう結果となります。

【不適切なスケジューリング】（表３－６－２）

						作業日程		
項番	成果物	工数	開始日	終了日	担当者名	○月○日	△月△日	△月△日
1-1-1	Aモジュール設計書	2週間	○月○日	△月△日	a	→	→	
1-1-2	Bモジュール設計書	1週間	○月○日	△月△日	b	✖→		○→
1-1-3	Cモジュール設計書	1週間	○月○日	△月△日	c	✖→		○→

一般的な大日程は、調査、仕様検討、基本設計、詳細設計、製造、単体テスト、結合テスト、総合テストの各工程の順に、必要なタスクを先に着手すべきものから順に記述していきます。

大／中日程表のサンプルは「付録図表　７．スケジュール表（大／中日程）」を参照のこと。

計画

3－7 プロジェクト目標の設定【スコープマネジメント】

プロジェクト目標の代表的なものとして、ＱＣＤの目標すなわち品質目標、コスト・利益目標、納期・生産性目標の三つをあげることができます。

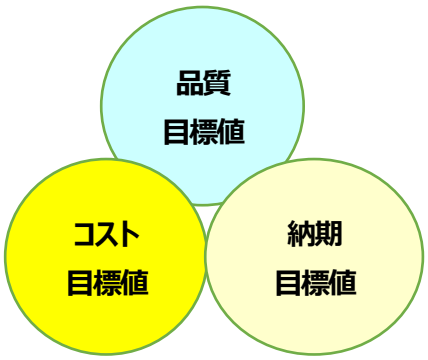
プロジェクトのＱＣＤ目標値は、見積り回答書にて示した数値内で設定することが絶対的な条件となります。次に、所属する上位組織のＱＣＤ目標に沿ったものが要求されますが、いずれにしても、その組織ないしはプロジェクトが持っている実力値以上の目標を達成することは極めて困難です。組織などの目標値の設定は、例えば三か年計画などに基づいて徐々に引き上げられていくことが現実的であり、達成の可能性も高くなります。 そのような意味で、プロジェクトの目標の設定にあたっては、最初に開発組織のＱＣＤの実力値を知るところから始める必要があります。

またＱＣＤの各目標は、同時に達成される必要があり、どれか一つないしは二つが達成されたとしても他の目標が達成されなかった場合は、成功プロジェクトとは言えません。

計画

◎ＱＣＤの目標値は顧客および社内経営層に約束したものであり、
どれか一つでも未達成ならば失敗プロジェクトとなる。

【ＱＣＤ目標値の設定】（図３－７－１）



スローガンではなく、目標数値の設定を！

計画

目標とは何か

目的[97]とは、「○○のために」という動機や理由を示すものです。一方、目標[98]とは、目的を達成するための具体的な手段を示すもので、「△△によって」というものです。例えば東京オリンピックについていえば、目的は「東京オリンピックを成功させるために」ということで、その目標は「何年何月までに１００億円を投じて５万名収容のスタジアムを完成させること」ということになるでしょう。「仕様理解力の向上」、「技術力の向上」、「評価業務の数値化」、「網羅性のある設計書の作成」、などという目標は、まだスローガン[99]レベルであり、具体性に乏しく、実現性の高い目標設定とはいえません。

われわれ実務レベルの開発者においては、自分の身近な仕事における、具体性のある問題を、数字をもって目標設定とする必要があります。例えば、仕様不具合を見抜けなかったことによって過去に発生した不具合の発生率・件数およびロス工数を具体的に数値で示し、それらの原因・真因を特定することで、具体的な対策案を提示し、改善効果として不具合件数を３０%削減する、ということを目標に設定するということです。

計画

◎目的とは、「○○のために」という動機や理由を示すもの。
目標とは、目的を達成するための具体的な手段を示すもの。

目標の見つけ方

より良い仕事をするためには、何が必要でしょうか。自分が把握できていない顧客の要望とは何でしょうか。コミュニケーションを図るとは、具体的にどのような手段で、何をするのでしょうか。無駄な作業にはどのようなものがありますか。本来やるべきことで、今までできなかったことにはどのようなものがありましたか。それぞれについて具体的な内容を書き出してみる必要があります。これらの具体的な内容から、自分の仕事における本当の目標が見つかります。まずは要求仕様の疑問点・不明点の解消および早期凍結に全力を注ぐ必要があります。さらに良い仕事を実現するためには、今までの失敗を振り返り、その改善行動を実行することが最も効果的な方法です。

計画

◎目標は、自他においての失敗や無駄の中にいくらでも見つけることができる。

---

97 **目的** ある行為を行う動機や理由を示したもの。「～のために」と表現されることが多い。

98 **目標** 目的を達成するための具体的な手段を示したもの。「～によって」と表現されることが多い。

99 **スローガン**（slogan） 目的や理念を表わしたイメージしやすい標語やキャッチコピーなど。

計画

目標の立て方

たとえば不具合発生件数０件ということは目標に相応しいと言えるでしょうか。不具合０件というのは一見、具体的でりっぱな目標のように見えますが、これはいわゆる「気合」や「意気込み」のたぐいのスローガンに過ぎません。１００ステップのプログラムでバグ０件はできるかも知れませんが、１万ステップのものではたぶん不可能でしょう。

「目標」と呼べるのにふさわしい条件は、まず現実的で達成可能であること、および測定可能な数値で示されることです。もし不具合件数を目標にするのなら、自分および他人における１ＫステップあたりのＸＸ不具合件数（単体テスト時、結合テスト時、総合テスト時）の過去の実績を記録に残しておき、目標値として自社の最高レベル～平均レベルの間の値を設定し、前期の自分の実績データと比較して今期は、３０％改善のＸＸ不具合件数目標＝○○件／１ＫステップなどをＸＸ設定すべきでしょう。これが合理的ないしは妥当性のある目標というものです。

計画

【目標の条件】

① 現実的で達成可能であること。

② 測定可能な数値で示されること。

【目標の立て方の順番】

① 現状の問題点を洗い出し、その数値化を行う。

② 問題点の真因を明確にし、改善策を立案する。

③ 改善策の期待効果に見合った目標値を設定する。

計画

品質目標値の設定

【品質マネジメント】

品質について

品質には製品品質[100]と業務品質[101]の二つがあります。製品品質とはモノ自体の品質であり、業務品質とはそのモノを作成するヒトの業務行為の品質のことです。

製品品質（モノの品質）は、業務品質（ヒトの品質）によって規定されてしまいます。ヒトがモノ（ソフトウェア）を作っている限りにおいては、業務品質レベル以上の製品品質の結果は生まれません。

計画

① 品質には業務品質と製品品質がある。
② 業務品質が上がらなければ製品品質はあがらない。

業務品質（ヒトの品質）の設定

業務品質とは、ヒトの業務における行動の切り口から見た品質であり、プロジェクトチームの開発能力のことを意味します。端的に言えばヒトの能力であり、具体的業務能力としてはプロジェクトマネジメント能力、プロセス遂行能力、技術能力、生産性レベルなどがあります。

業務品質の主な指標例としては以下のものがあります。

【業務品質の指標例】

① 仕様品質：仕様ミス件数／率（／Kstep）、深刻度ＡＢＣ分布、変更影響度表作成率
② 設計品質：設計ミス件数／率（／Kstep）、深刻度ＡＢＣ分布
③ 製造品質：製造ミス件数／率（／Kstep）、深刻度ＡＢＣ分布
④ 評価品質：評価ミス件数／率（／Kstep）、深刻度ＡＢＣ分布、バグ発見数／率
⑤ 構成管理[102]品質：構成管理ミス件数／率（／Kstep）、深刻度ＡＢＣ分布
⑥ プロセス管理品質：プロセス管理表実施率
⑦ リスク管理品質：リスク対策件数／率（／Kstep）、深刻度ＡＢＣ分布、リスク管理表実施率
⑧ 課題管理品質：課題対策件数／率（／Kstep）、深刻度ＡＢＣ分布、課題管理表実施率
⑨ 各レビュー品質：レビュー指摘件数／率（／Kstep）、深刻度ＡＢＣ分布、レビュー実施率

---

100 **製品品質** 物自体の品質。例：ソフトウェアの不具合発生率等。

101 **業務品質** 人間が行う業務能力の品質。例：作業ミス率等。

102 **構成管理** ソフトウェア開発におけるソースコードや設計書などの成果物に対して変更履歴を管理し、必要に応じたバージョンの成果物を取り出すことを可能にする。一般的に構成管理ツールが用いられ、ビルド管理、リリース管理、バージョン管理などを通して成果物の一貫性を保つようにしている。

計画

深刻度ＡＢＣ分析[103]は、障害内容を顧客の視点から見て、その深刻度のレベルを高いものの順に、Ａ：重度障害、Ｂ：中度障害、Ｃ：軽度障害と定義し、その発生件数・率などによって品質の傾向を把握するものです。

計画

◎ヒトの業務品質を向上させるためには、継続的な業務改善活動を！

製品品質（モノの品質）の設定

製品品質とは、モノ（ソフトウェア）の切り口から見た品質であり、開発能力（業務品質）以上の製品品質が生まれることはありません。製品品質の主な指標例としては以下のものがあります。

【製品品質の指標例】

- 単体テスト品質：不具合件数／率（／Kstep）、深刻度ＡＢＣ分布
- 結合テスト品質：不具合件数／率（／Kstep）、深刻度ＡＢＣ分布
- 総合テスト品質：不具合件数／率（／Kstep）、深刻度ＡＢＣ分布
- 市場品質：市場障害件数／障害率、深刻度ＡＢＣ分布

製品品質における深刻度ＡＢＣの例を以下に示します。

Ａ重度障害の例：全システムダウン、会計数値違算、大規模な取引・発注・在庫管理機能不正等

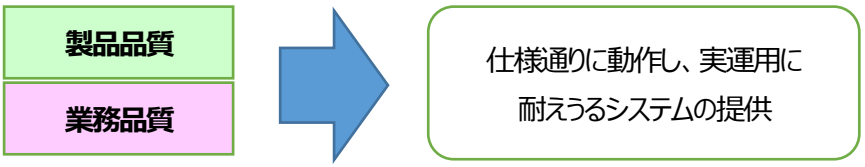
Ｂ中度障害の例：一部システムダウン、一部端末のロック、一部取引・発注・在庫管理機能不正等

Ｃ軽度障害の例：会計数値に影響しないレベルの不適切または誤った印字・表示・操作シーケンス等

計画

◎製品品質は、不具合発生件数・密度および深刻度の３点により評価すること。

【２つの品質で支えるシステム】（図３－７－２）



---

103 **ＡＢＣ分析** 障害内容を顧客の視点から見て、その深刻度のレベルを高いものの順に、Ａ：重度障害、Ｂ：中度障害、Ｃ：軽度障害と定義し、その発生件数・率などによって品質の傾向を把握するためもの。

その他の製品品質指標

製品品質については、ソフトウェアの信頼性の視点を中心に示してきましたが、バグが出ないだけでソフトウェアの品質が高いとは言えません。ハードウェアの世界において知られている優れた製品の５つの特性についても、指標化および目標値の設定が必要になります。 本来、総合テストはこれらの５つの特性検証の場であって、単なるバグ潰しの最終工程ではないのです。

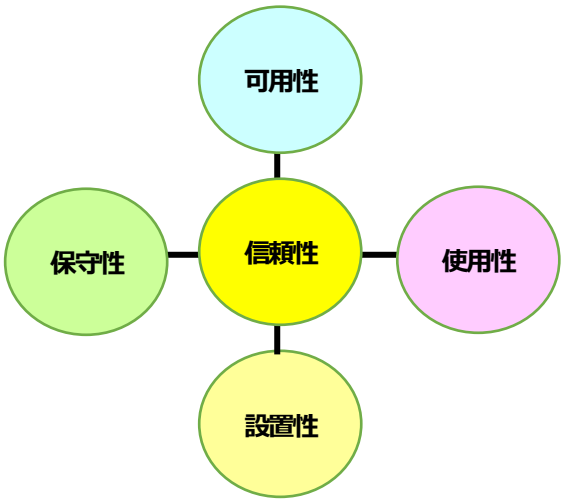
【優れた製品の５つの特性：ＲＡＳＵＩ[104]】

- リライアビリティ Reliability：信頼性（不具合・障害率の低さ）
- アベイラビリティ Availability：可用性（継続的稼動能力）
- サービサビリティ Serviceability：保守性（保守・メンテのしやすさ）
- ユーザビリティ Usability：使用性（使いやすさ）
- インストーラビリティ Installability：設置の容易性

計画

◎優れた製品の５つの特性：
①信頼性、②可用性、③保守性、④使用性、⑤設置容易性

【優れた製品の５つの特性】（図３－７－３）



[104] **ＲＡＳＵＩ** 優れた製品の５つの特性である、Reliability：信頼性（不具合・障害率の低さ）・Availability：可用性（継続的稼動能力）・Serviceability：保守性（保守・メンテのしやすさ）・Usability：使用性（使いやすさ）・Installability：設置の容易性、の略称。

計画

品質目標の達成に向けて

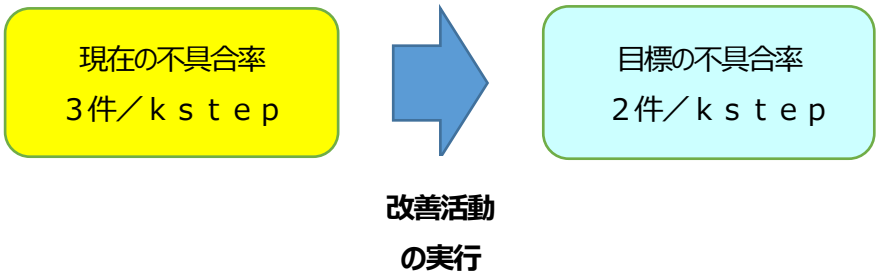
いずれの指標もソフトウェア業界全体を網羅できるような統一的基準値で表すことは困難であり、同一組織における過去・現在の実績値の変動の比較によって目標値を設定するのが現実的です。

業務品質を上げるためには、プロジェクトマネジメント能力の向上、プロセス遂行能力の向上、技術能力の向上、生産性の向上などが必要になりますが、このようなヒト依存的な能力の向上を実現するためには、上記に示した品質の達成目標を目指した改善活動を、個人の自主性にまかせるのではなく、組織的な取り組みとして実際のプロジェクトの中で実行する必要があります。

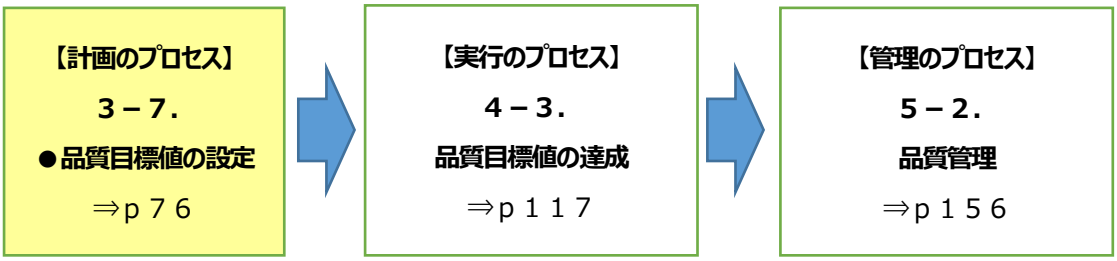
計画

◎品質の目標値は過去・現在の実績値の平均を見て設定すること。

【不具合率の削減に向けての活動】（図３－７－４）



品質に関する本章以降の解説の流れは下図のようになっています。



コスト・利益目標値の設定

【コストマネジメント】

損益管理表

コストないしは利益目標を設定するためには、プロジェクトにおける売上高・製造原価・粗利・販売管理費・営業利益などの関係を表わす損益管理表[105]を理解しておく必要があります。

プロマネ初心者にとってはなかなかイメージが湧きにくい管理表の一つですが、先任プロマネが所有している実際の損益管理表で一度説明を受ければ、より理解が深まるものと思います。

損益管理表は次の図の通りです。

【損益管理表】（表３－７－１）

①売上高	
②製造原価	②-１　直接材料費（外注費、購入部品費等）
	②-２　直接労務費（社内人件費、経費等）
	②-３　間接費（製造間接費、仕損費等）
③粗利	③＝①－②：粗利は売上総利益とも呼ばれます。
④販売管理費	④-１　一般管理費（管理部門の経費）
	④-２　販売部門費（販売部門の経費）
	④-３　研究開発費（研究部門の経費）
⑤営業利益	⑤＝③－④

計画

売上高＝製造原価＋販売管理費＋営業利益
製造原価＝直接材料費＋直接労務費＋間接費
粗利（売上総利益）＝売上高―製造原価
販売管理費＝一般管理費＋販売部門費＋研究開発費
営業利益＝粗利―販売管理費＝売上高―製造原価―販売管理費

105 **損益管理表**　プロジェクトにおける売上高と費用（コスト）を対比して、その差額として利益を示すもの。一般的には損益計算書と呼ばれている。

計画

上記はそれぞれ次のように言い換えることもできます。

売上高はプロジェクトが受注した金額。

製造原価[106]は開発チームが開発に要したコスト。直接材料費[107]、直接労務費[108]、間接費[109]など。

粗利[110]は売上高から製造原価を引いたもの。

販売管理費[111]は開発チーム以外の管理・販売・研究部門等に要する費用（コスト）。

営業利益[112]は売上高からすべてのコスト（製造原価・販売管理費）を引いたもの。

上記を構成表で表すと次のようになります。

【損益管理表の構成表】（表３－７－２）

①売上高	②製造原価	②-１直接材料費 （外注費、購入部品費等）
		②-２直接労務費 （社内人件費・経費等）
		②-３間接費 （製造間接費、仕損費等）
	③粗利	④販売管理費
		⑤営業利益

この図からも分かるように開発プロジェクトは、売上高（受注金額）から粗利（販売管理費＋営業利益）を除いた製造原価以内で、開発に関するすべての費用をまかなう必要があります。

一般的に粗利部分は、開発会社全体の経費や利益をまかなうために一定の%が決められており、見積り時に開発費用（製造原価）に上乗せされ、見積り合計（売上高）となります。

計画

◎プロジェクトは製造原価以内で開発を実行しなければならない。

106 **製造原価** 直接材料費、直接労務費、間接費など開発チームが開発に要した費用（コスト）。

107 **直接材料費** 外注費および開発ツールや備品等の購入品に要した費用。

108 **直接労務費** 開発人件費および旅費・交通費・通信費・消耗品等経費に要した費用。

109 **間接費** 開発を間接的にサポートする開発管理部門等の費用。

110 **粗利** 売上高から製造原価を差し引いたもの。売上総利益とも言う。

111 **販売管理費** 開発チーム以外の管理・販売・研究部門等に要する費用（コスト）。

112 **営業利益** 売上高からすべてのコスト（製造原価・販売管理費）を差し引いたもの。

コスト･利益目標値の設定

プロジェクトマネージャがコントロール可能な領域は基本的に製造原価のみで、製造原価が下がれば粗利は増え、逆に製造原価が上がれば粗利は減ってしまいます。

損益管理表の各項目の費用におおまかな%を割りつけてみると下記のようになります。

【損益管理表の構成配分率】（表３－７－３）

①売上高 １００%	②製造原価 ７０%	②-１直接材料費　３０% （外注費、購入部品費等）
		②-２直接労務費　３５% （社内人件費、経費等）
		②-３間接費　５% （製造間接費、仕損費等）
	③粗利 ３０%	④販売管理費　２５%
		⑤営業利益　５%

プロジェクトにおいて粗利を３０%確保するためには、開発に使える費用は受注金額の７０%になってしまいます。プロマネはこの金額以内に開発コストを抑える必要があります。因みに製造業における粗利率は３１．９%、営業利益率は２．１%であり、情報通信業ではそれぞれ５７．４%、２．０%（2005 年、中小企業庁調査）となっており、一見粗利率が高そうに見えても販売管理費が非常に高い割合を占めた結果、営業利益率はわずか２%程度というのが実態なのです。

計画

◎粗利を３０%以上確保しなければ、まともな利益は獲得できない。

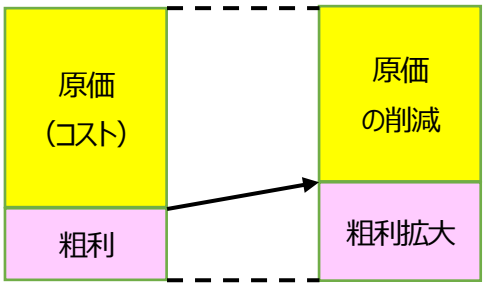
仮に上記表の数値でプロジェクトのコスト目標すなわち製造原価目標を設定した場合、売上高が１０００万円ならば、開発費の目標は７００万円に設定することになります。

さらにプロジェクトがプロジェクト自身の利益を獲得する必要がある場合は、開発費を６００万円に設定し、プロジェクトの利益として１００万円を残す必要があります。この資金はプロジェクトチームの次の開発のための事前準備や調査の費用として使用することで、大きなリスクの事前解消に役に立つことでしょう。

このような施策はプロジェクトチームにおける機敏性･自律性･リスク回避能力を各段に高めるでしょう。

いずれにしても所定の目標以上のコスト削減（利益の確保）を達成するためには、組織的な改善活動をプロジェクト活動の中に織り込むこと、すなわちリスクの事前解消･失敗の削減･仕事の効率化･無駄の排除などの具体的な活動の裏付けが必要です。

【粗利の拡大】（図３－７－５）



コスト・利益目標値の例は以下の通りです。

【利益指標】

- 粗利額・率
- 営業利益額・率

【コスト指標】

- ロス（手戻り）率（対開発費）＝ロス工数／開発費（各工程毎）
- ロス（手戻り）削減率（対開発費）＝削減ロス工数／開発費（各工程毎）
- ロス（手戻り）率（対ステップ数）＝ロス工数／１Ｋステップ（各工程毎）
- ロス（手戻り）削減率（対ステップ数）＝削減ロス工数／１Ｋステップ（各工程毎）

・設計工数効率化率＝設計工数／１Ｋステップ ・製造工数効率化率＝設計工数／１Ｋステップ ・評価工数効率化率＝評価工数／１Ｋステップ ・管理工数効率化率＝管理工数／１Ｋステップ	・直接労務費・間接労務費の比率 ・工程別開発費配分率 ・見積り時間の短縮化率／時間 ・見積り外作業工数削減率／金額

- ソフト共用モジュール登録数・再利用率
- ドキュメント作成時間の短縮化率／時間
- チェックリスト共有化率／件数

コスト・利益に関する本章以降の解説の流れは下図のようになっています。

【計画のプロセス】
３－７．
●コスト・利益目標値の設定 ⇒ｐ８０

⇒

【実行のプロセス】
４－４．
コスト・利益目標値の達成
⇒ｐ１２６

⇒

【管理のプロセス】
５－３．
コスト・利益管理
⇒ｐ１６４

計画

納期・生産性目標値の設定

【タイムマネジメント】

納期の目標とは、単に納期の期日を守るということではないと言うことは誰でも分かっていることですが、切羽詰まって時間切れになってしまったプロジェクトが、多くのバグを内在したままでソフトウェアをリリースしてしまうことが少なからずあります。

納期が顧客にとって本当の価値を持つ大前提は、リリースされる製品が顧客の要求を満たしていると同時に品質が確保されていると言うことです。 言い換えれば、納期の目標とは、納期の期限内に品質を確保するための目標を設定することだと言えます。このことを実現するためには開発の各工程においても同様のことが実現される必要があります。すなわち次のようなことです。

- 要件定義の納期までに、品質が確保された要件定義を実行しなければならない。
- 基本設計の納期までに、品質が確保された基本設計を実行しなければならない。
- 詳細設計の納期までに、品質が確保された詳細設計を実行しなければならない。
- コード製造の納期までに、品質が確保されたコーディングを実行しなければならない。
- 総合テストの納期までに、品質が確保された総合テストを実行しなければならない。

計画

◎納期は品質が確保されなければ意味がない。

以上のことを実現するためには、それぞれの工程に許される期間内にそれぞれの成果物の品質を確保する必要があると言う当たり前の結論になりますが、問題のポイントは、許された時間よりそのプロジェクトが必要とする時間が長い場合にあります。必要時間[113]が長ければ当然時間不足となり品質の確保は不可能だと言えます。必要時間とは、言い換えればそのプロジェクトの実行能力（実力）を表わすものです。必要な時間を許容時間[114]内に収めるためにはプロジェクトの生産性能力を上げる以外には方法はありません。

計画

◎納期・品質の同時達成には生産性能力の向上が必要となる。

結論を言えば、納期目標の設定とは、要件定義／基本設計／詳細設計／コード製造／総合テストなどに関して期限までに品質を確保した成果物を OUTPUT するための生産性目標値の設定を行うと言うことになります。具体的には下記のような各成果物に対する品質レベルの目標値および時間あたりの生産効率が目標値となります。

---

113 **必要時間** ある仕事を実行するために必要な時間。実行する人や組織の能力により長短の差が出て来る。

114 **許容時間** ある仕事を実行するために許される一定の時間。

計画

各工程の納期が守られることは当然のこととして、それぞれの工程における生産性指標の例としては次のようなものがあります。

【生産性指標の例】

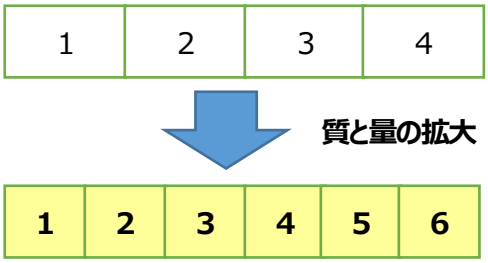
- 要件定義の生産性＝要件定義費用／要件定義時間
- 設計の生産性（設計効率）＝設計費用／設計時間
- 製造の生産性（製造効率）＝製造費用／製造時間
- 総合テストの生産性（テスト効率）＝総合テスト費用／総合テスト時間
- 納期達成率＝納期達成ＰＪ数／全ＰＪ数（ただし品質・コスト目標が達成されたＰＪのみ）
- 失敗による手戻り費用率＝手戻り費用／工程費用（開発各工程毎）
- 障害対応費用効率＝障害対応費用／全開発費

これらの指標値に基づいたプロジェクトの目標値の設定を行う必要があります。

計画

◎納期目標の設定は、成果物の精度目標および効率目標の設定に拠る。

【生産性：時間当たりの成果物】（図３－７－６）



納期・生産性に関する本章以降の解説の流れは下図のようになっています。

【計画のプロセス】
３－７．
●納期・生産性目標の設定 ⇒ｐ８４

【実行のプロセス】
４－５．
納期・生産性目標の達成
⇒ｐ１３１

【管理のプロセス】
５－４．
進捗管理
⇒ｐ１６８

計画

目標値設定による期待効果

【スコープマネジメント】

目標値の設定は、データに基づいた開発（データドリブン開発[115]）がプロジェクト遂行の第一歩と言え、ソフトウェア開発を科学的・工学的に遂行する基本的な手段となります。

数値に基づく開発は、プロジェクトの代表的な問題である、あいまいさや情緒的な判断を一掃する役割を果たします。

目標値の設定は次のような効果を生み出します。

【目標値設定による期待効果】

- 現状の開発力が数値として明確になる。
- 改善すべき問題を数値的に把握できる。
- 改善結果が数値的に明らかになる。
- 到達すべきゴールが数値的に見え、モチベーションの維持につながる。
- あいまいさや情緒性を排し、合理的ないしは妥当性のあるチームを育成する。

計画

◎目標は数値として表すことで、実行可能な目標となる。

---

115 **データドリブン開発** データを根拠にした開発活動のこと。⇒参照：４－２．ＱＣＤ目標値の達成●データドリブン開発とは何か

計画

目標をあきらめない

【スコープマネジメント】

仕事の目標は品質・コスト・納期の達成であると言われているのにもかかわらず、現実はコスト・納期の限界で品質を犠牲にしている場合が少なからずあります。コスト・納期の限界で多くの開発工程が時間切れとなり、それぞれの仕事が中途半端な未完成の状態で次の工程に回され、最後に市場において多くの障害を生んでいるのです。これは仕方ないといってあきらめても良い問題ではありません。中途半端な仕事でコストや納期を守ったつもりでも、結果として障害対応によってさらに多くのコストや時間を浪費することになり、最も大切な顧客の信用を失う結果を招いてしまいます。

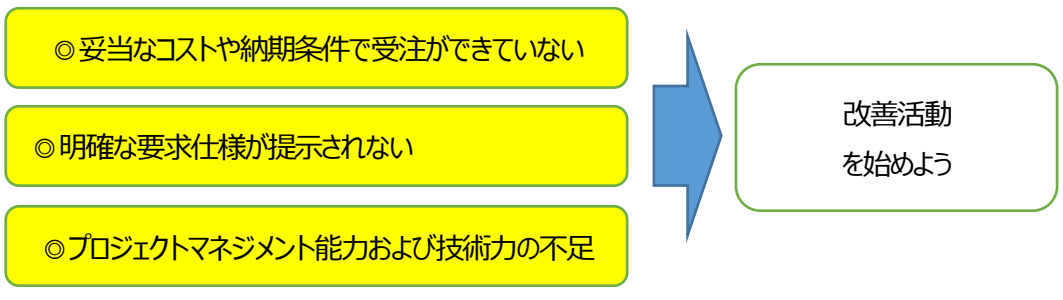
なぜコスト不足・時間不足になっているのか真剣に考える必要があります。多くの問題は、要件定義や見積りなど開発の前工程に存在していることは誰でも気づいていることですが、多くの人はその改善に着手しようとはしません。改善活動は仕事には含まれていないと考えている開発者やマネージャが意外に多いのです。

改善活動こそ仕事の中核業務であるという意識を持つ必要があります。

失敗プロジェクトの一番の問題は、妥当なコストや納期条件での受注ができていないこと、二番目の問題は明確な要求仕様が提示されないこと、三番目の問題は合理的なプロジェクトマネジメント能力および技術力の不足にあります。この問題を解決するためには現在までの自分たちの失敗を直視し、その真因を把握し、その改善目標を設定し、改善活動を実行・継続する必要があります。

目標の達成をあきらめたらその時点で失敗が確定してしまうことになります。

【失敗プロジェクトの原因ワースト３】（図３－７－７）



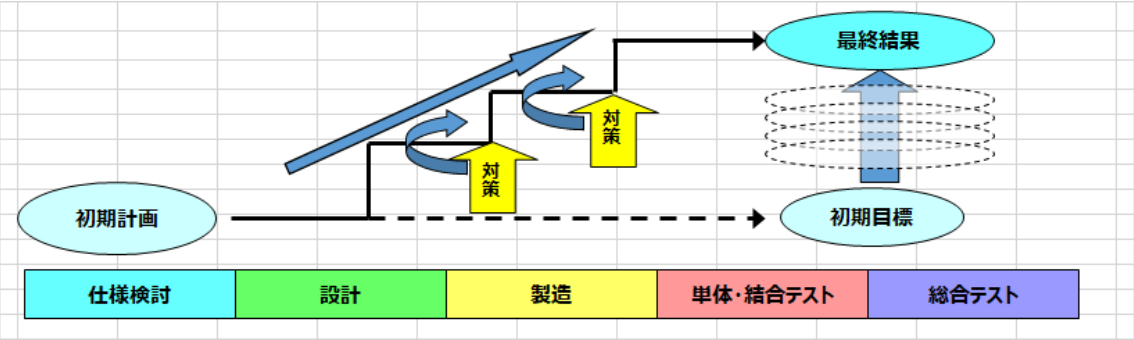
計画

◎目標は、あきらめなければ必ず達成可能である。

【目標は動いている】

到達目標は、初期に想定した位置から時間の経過・環境・条件の変化に伴い必ず変化していくものです。動的に動く目標をヒットするためには、変化を見逃さずタイミングの良い対策を連続して実施することが必要です。この事を図で表すと次のようになります。

【目標は動いている】（図３－７－８）



プロマネは、開発の進行とともに変化していく顧客の要求や開発の状況を見誤ることなく、それぞれの問題に対して適切な対策を適切なタイミングで打ち続けることで、プロジェクトの成功という最終目標を達成する必要があります。

3－8 プロジェクト体制の構築【人的資源マネジメント】&【調達マネジメント】

プロジェクトに必要なリソースの内で最も重要なものは人間そのものです。物・資金・情報などさまざまなリソースを使うのも結局人間であり、人間の出来不出来によっては、これらのリソースが十分にあったとしてもそれを生かし切ることはできません。 結局プロジェクトの成否の鍵はどのような人材でプロジェクトの体制を構築するのかにかかっています。

プロジェクト体制構築の要件 【人的資源マネジメント】

プロマネは社内全体の他のプロジェクトの状況を常に把握しておくことで、新規プロジェクトに投入すべき人材の確保を確かなものにしておく必要があります。未経験者ばかりしかいないプロジェクト体制などは、最初から失敗が約束されているようなものです。プロマネは、開発の規模や難易度を把握し、それに見合ったスキルの人材を用意しておく必要があります。多くの場合、自分のプロジェクトに必要な人材を適時に集めることは難しく、そのためには社内の組織を初めとして外注組織においてどのような人材がいて、適材者がいつプロジェクトに参加可能なのか、という情報を普段から把握しておく必要があります。

プロジェクト体制構築にあたって必要な要件は次のようになります。

◆【プロジェクト体制構築の要件】

- □ 名ばかり管理職をプロジェクトの責任者につけないこと。
- □ 上級者から初級者に至るまで、適切なスキルの人材で、指揮系統が明確なピラミッド構成を築くこと。
- □ 社内で人材の確保が難しい場合は、協力会社なども含めて、広く人材の確保に努めること。
- □ メンバーの役割を明確にし、それに適合する人材を割り当てること。
- □ メンバーは、開発要件に必要な技術的能力およびチームプレーに必要なコミュニケーション能力（報告・連絡・相談）などを備えていること。
- □ メンバーのスキルに関しては、組織全体で個人スキル管理カードなどにより常時管理を行うこと。

上記の要件で結成された開発チームは、数次にわたるプロジェクト活動および改善活動を通して、下記のような優れた能力を持つ開発チームに成長していくことが期待されます。

計画

◆【優れた開発チームの能力特徴】

- □ 自律的思考[116]・行動に基づいた自己決定力
- □ 意欲ある人材でプロジェクトを構成するプロジェクト構築力
- □ 顧客との日々の直接コミュニケーションによる協調力
- □ チームにおける日常的で密接な直接コミュニケーション力
- □ 顧客からの変更要求に対する俊敏な対応力
- □ 顧客価値優先度の把握力
- □ ドキュメントの常備はもとより、ちゃんと動作するソフトウェアの素早い開発力
- □ 一定のペースを持続できる開発力
- □ 無駄・不要な仕事の削減力および業務効率化能力
- □ 定期的・効果的振り返りによる、学習能力および行動の変更・調節力

（参考資料：アジャイルソフトウェア開発宣言[117]）

計画

◎確実なチーム体制構築は、プロマネにおける人材能力の見極め力と人材獲得力にある。

開発体制不備による失敗リスク

【リスクマネジメント】

開発体制の不備はプロジェクトの失敗に直結してしまいます。開発体制不備に関するリスクには以下のようなものがあります。

◆【開発体制不備のリスク】

- □ 開発要件を満たすプロマネ・開発者の不足
- □ 開発者のトレーニング不足
- □ 未経験分野・新規業界参入の場合の準備不足
- □ 新技術採用場合の準備不足

これらのリスクは開発実行の前の事前準備段階において解消される必要があります。要件定義工程や設計工程に入ってから、必要な人材が足りないと騒いでも手遅れなのかも知れません。

計画

◎必要な人材の獲得は事前準備段階で済ませておくこと。

---

116 **自律的思考** 自分自身でものごとを直接観察し、その意味するところを判断すること。

117 **アジャイルソフトウェア開発宣言** アジャイルソフトウェア開発宣言は、アジャイルソフトウェア開発とその諸原則を公式に定義した文書である。この諸原則はウォーターフォール開発にも有意義なヒントを与えている。

計画

外注開発体制の構築

【調達マネジメント】

分業体制が一般的になっている状況下において、元請けベンダー側にとっては、どれだけ有能な外注会社を抱えているのかがプロジェクト成功のファクターの一つになっています。とは言っても外注会社が有能であるためには元請ベンダー側は、さらにそれよりも有能な組織である必要があります。元請がプロジェクトマネジメントもよく分からない、要件定義も設計もよく分からないようでは下請会社もついて来なくなるでしょう。

外注会社における開発力の成長・維持のためには、定期的な外注会社の評価やＱＣＤ改善のための提案書の提出などが必要になります。

外注会社に対する評価項目は、技術力・品質・コスト・納期の実績等に基づいた下記の項目などが必要でしょう。

【外注会社評価項目】

① 保有技術およびそのレベルの実績推移
② 発注品の難易度評価
③ 品質の実績推移（不具合発生件数・率・深刻度）
④ 品質管理体制の評価
⑤ 技術者のランク別の単価推移
⑥ 取引金額の実績推移
⑦ 納期・生産性の実績推移
⑧ 協力度評価

計画

◎有能な外注は有能な元請プロマネの下で育つ。

統合的なプロジェクト体制の構築

【統合マネジメント】

従来の開発体制は、ベンダー側にてプロジェクトの統合管理、要件定義およびシステムの基幹部の開発および評価を担い、不足要員を国内下請け会社で補う形が一般的でしたが、現在は外形的にはベンダー側における開発や評価業務は下請け会社にシフトされており、特に製造工程はコストメリット追求のために海外オフショア[118]会社に発注されている場合が多いようです。

これらの分業は、開発の工程単位で行われることが一般的で、この方式を工程別分業方式[119]と呼びます。

この工程別分業方式が持っている致命的なリスクとしてプロジェクト統合管理の欠落があります。

統合管理および統合コミュニケーション[120]の欠落

２・３人のチームで要件定義からリリースまでの全工程を担う小さなプロジェクトでは、コミュニケーションや情報の断絶などは起こりませんでしたが、開発規模の増大により各工程が別々の担当者で行われはじめると断絶が起こり始めます。さらに一つの開発プロジェクトの各工程を異なった複数の会社で担当した場合は、断絶は深刻な状態にまで進行し、失敗のリスクが非常に高くなってしまいます。このことは皆さんすでに経験済みのことでしょう。

それにもかかわらず各社各担当で分断されたコミュニケーションや情報の溝を埋めるための統合的なプロジェクトマネジメントはなかなか実行されません。

ベンダー側の言い分は、あの工程はＡ社に発注したのだから、責任をもって仕上げるのが当然などと主張していますが、その一方で要求仕様の早期凍結という自社の義務は果たされていません。これでは下請け会社は、自分の責任を全うすることは不可能になってしまいます。複数社で構成された開発プロジェクトが成功するためには、より上位にある会社が全体を取り仕切る統合マネジメントを実行する必要があります。この重要な義務を果たさない限りプロジェクトが成功することはないでしょう。

月に一度程度の管理職どうしで行われる形式的なパートナー会議では、本当の意味でのプロジェクト内の情報共有を図ることはできません。少なくとも各社のプロマネや開発リーダーを集めた情報共有および問題解消のための会議を週一回程度は実行する必要があり、元請け会社は各社間の調整および全体的なＱＣＤコントロールの指揮をとる必要があります。

計画

◎プロジェクトの統合マネジメントは、
コミュニケーションの分断を防ぎ、プロジェクトを成功に導く。

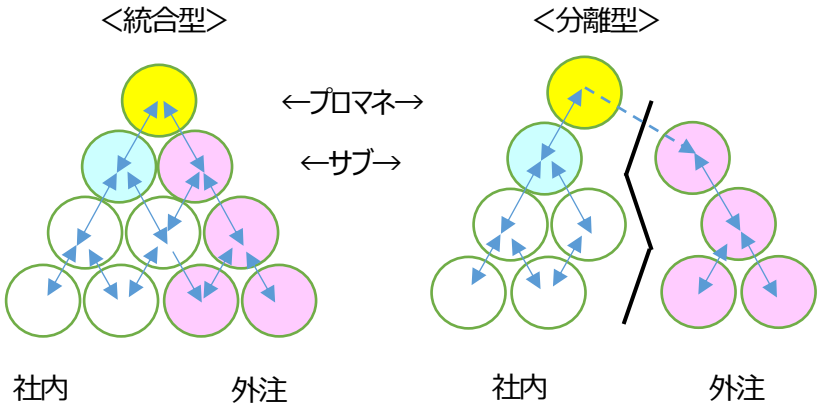
---

118 **オフショア**（Offshore） ソフトウェア開発に関する業務を海外の会社に委託する形態のこと。

119 **工程別分業方式** ソフトウェア開発の各工程を別々の会社で分業して担当している方式。

120 **統合コミュニケーション** 開発工程ごとに担当会社や担当者が異なる場合に発生するコミュニケーションの溝を埋めるための計画的な情報共有会議の実施等。 参照⇒「３－ｈ１．プロジェクト統合管理」

【統合的なプロジェクト体制】（図３－８－１）



3－9 開発環境の手配【調達マネジメント】

リソースの用意として忘れられがちなものが開発機器や評価機器の準備です。特にファームウェア[121]開発におけるターゲットマシン[122]と呼ばれる開発ソフトウェアが搭載される機器の手配には注意を払っておく必要があります。ターゲットマシンはハードウェア開発部署ないしは製造部門から入手する必要があり、通常そのリードタイムには長期間を要する場合も珍しくありません。特に依頼先が他社の場合は小まめなフォローが必須です。

手配の約束や依頼は行ったが、その後のフォローを忘れていたために、マシンの入手時期が数週間も遅れてしまう場合もあり、スケジュールに大きなインパクトを与えることになります。

また近年においては厳しいセキュリティが求められる社会情勢下にあり、開発中においては顧客の重要なデータを取り扱うことも多く、これらの取り扱いに関しては社内ルールのもとに厳重な取り扱いを行う必要があります。特にクレジット・電子マネー・ポイントなどのデータの取り扱いに関しては、顧客の承認を得た専用のセキュリティルームの用意が必要であり、本番環境での開発およびテストで使用されるこれらのデータは、顧客所有の現金と同じであるという認識を開発者全員で持つ必要があります。

これらのデータの流出問題は、賠償問題に直結し、会社の経営を揺るがした事例も少なからずあります。

---

121 **ファームウェア** ファームウェア(firmware)とは、電子機器に組み込まれたコンピュータシステム（ハードウェア）を制御するためのソフトウェアで、ソフトウェアを ROM 等の集積回路にあらかじめ書き込まれた状態で、機器に組み込んだもの。

122 **ターゲットマシン** ファームウェアを搭載するハードウェア機器。

3－10 プロジェクト計画書の作成【統合マネジメント】

プロジェクト計画書[123]はプロジェクト計画プロセスの総まとめであり、開発実行の行動計画書となるもので、プロジェクトをスムーズに運営し、ＱＣＤ目標を達成するために必要な活動内容を計画書という形で明文化したものです。プロジェクト計画書には、見積り内容、プロジェクトのスコープ、スケジュール、コスト、リスク、ステークホルダーとのコミュニケーション方法などの記述が必要です。

プロジェクト計画書の目次の例を以下に示します。

【プロジェクト計画書】

１．概要

（１）プロジェクト計画書の目的

（２）プロジェクト概要（開発目的、スコープ、納品物、リリース基準、開発プロセス、プロジェクト環境）

（３）見積り内容（見積結果等）

２．ソフトウェア開発計画

（１）プロジェクトチーム（開発体制、役割と責任）

（２）リソース計画（プロジェクトに必要なリソース、トレーニング計画等）

（３）日程計画（マイルストン）

（４）プロジェクト進捗管理（小日程表、進捗報告、定例会議、レビュー、顧客への報告等）

（５）リスク管理計画

３．ソフトウェア品質保証計画（ガイドライン、レビュー計画、テスト計画、テスト範囲）

４．ソフトウェア構成管理（ＳＣＭ）計画

（１）構成の確定（管理対象、管理文書、管理ツール等）

（２）構成管理のプロセス

（３）構成管理状態の把握（ステータス報告、状況把握の頻度。報告書の配布）

（４）構成管理ステータス監査（監査方法、監査の頻度）

５．外部委託管理計画

（１）外部委託選定理由及び見積り

（２）外部委託計画内容

６．システム導入計画（システム導入日程等）

７．その他の情報（製品情報、書類番号等）

---

123 **プロジェクト計画書** 開発実行の行動計画書となるもので、プロジェクトをスムーズに運営し、ＱＣＤ目標を達成するために必要な活動内容を計画書という形で明文化したもの。プロジェクト計画書には、見積り内容、プロジェクトのスコープ、スケジュール、コスト、リスク、ステークホルダーとのコミュニケーション方法などの記載が必要。

計画

プロジェクト計画書作成にあたって下記の要件をチェックすることで漏れを防ぐことができます。

◆【プロジェクト計画書要件チェックリスト】

- □ 開発目的は明確になっているか。
- □ プロジェクトの開発範囲は明確になっているか。
- □ 納品物は明確になっているか。
- □ 納品物の検収基準（リリース基準）は明確になっているか。
- □ 開発プロセスは明確になっているか。
- □ プロジェクト環境（開発環境・テスト環境・実稼働環境）が明確になっているか。
- □ 参考にした過去のプロジェクトのデータが明確になっているか。
- □ 見積結果が添付されているか。
- □ 商品企画書、開発指示書が添付されているか。
- □ 開発体制及び役割と責任は明確になっているか。
- □ 顧客から提供されるリソースは明確になっているか。
- □ プロジェクトで必要なリソースは明確になっているか。
- □ 他の組織から提供されるリソースは明確になっているか。
- □ 日程計画（スケジュール表、プロセス管理表）は立てられているか。
- □ 進捗管理方法（進捗管理表、定例会議、レビュー、進捗報告）は明確になっているか。
- □ リスク計画は立てられているか。
- □ ソフトウェア品質保証計画は立てられているか。
- □ ソフトウェア構成管理計画は立てられているか。
- □ 外注管理計画（外注委託選定理由、見積り、計画内容）は立てられているか。
- □ システム導入計画は立てられているか。

上記に示されたようにプロジェクト計画書は、プロジェクトの全体像をその根拠と共に表したもので、社内経営層の承認のもとに、顧客をはじめとした重要なステークホルダーと共有されるべき資料です。

【Human Activity】

3－h 1 プロジェクト統合管理【統合マネジメント】

バラバラに散らばったものはシステムとして機能しません。分散したもの同士を一定のルールに従ってまとめ、ある目的を果たすように機能させたものがシステムです。人も物も、同じ原理で社会的な機能を果たしています。ものごとをシステムとして機能させる本質は、この"まとめる"という作用ないしは行為のことです。このまとめ役を担う者をリーダーと呼び、ばらばらな人間集団を統合管理することで、人や物に一定のルールを持たせ、ある目的に従った機能を果たさせることができます。開発におけるソフトウェアを初めとした全ての成果物は開発者の分身だと言えます。開発者がルールを逸脱すれば、作成された設計書もソフトウェアもルールを逸脱する結果となります。プロジェクトを統合管理するプロマネおよびリーダーの役割や責任は重いのです。

プロジェクト統合管理の役割

一つのプロジェクトが複数社にて構成されている場合、中心となるベンダーにおいては各社をとりまとめる統合管理を実行する必要があります。プロジェクトにおける統合管理とは、複数の開発者や組織間における密接なコミュニケーションを通した情報共有、開発進捗の足並みを揃えること、問題・課題の相互調整、各社間の協力体制および共通の開発ルールの確立などを意味しています。

これらの統合管理が行われないプロジェクトは、必ず失敗するだろうと言うことは誰にでも分かることですが、現実のプロジェクトでは統合管理が行われていないことも珍しくありません。

卑近な例として、複数の会社が同一モジュールのソース変更を行う場合などにおいては、変更理由記録・変更履歴やソースにおけるコメント記入など、後日のデバッグや不具合修正に必要な記録をとるような共通のルールを取り交わしておかなければ、開発者たちは間違いなく混乱状態に陥ります。

複数社・複数組織がからむプロジェクトにおいては、ベンダー側におけるプロマネおよび開発リーダーは下記に示したようなプロジェクトの統合管理の役割を果たす必要があります。

計画

【プロジェクト統合管理の役割】

① 関係各社における進捗の相互調整
② 関係各社における問題・課題の相互調整
③ 関係各社間の情報共有
④ 関係各社間の協力体制および共通開発ルールの確立

統合管理を阻害するもの　～工程別分業方式の欠陥

近年オフショア開発の進展に伴って、要件定義はベンダー、設計は国内下請、製造は海外オフショア会社、評価は国内下請もしくはオフショアなどと、一つのプロジェクトの各工程を複数の会社で分業するような開発形態が増えてきました。このような分業体制は、多くのリスクや問題点を内在させています。実際この工程別分業方式による開発で多くのプロジェクトが失敗しています。

この方式の最大の問題点は、情報の共有が極めて困難であるということです。この工程別分業方式においては、各工程が異なった会社に分離・分散されたことにより、共有すべき情報の分断化が発生しやすくなり、開発に多くの問題を起こすだろうということは容易に分かる道理のはずですが、何故か多くの元請ベンダーにおいてはその認識が極めて薄いようです。

ソフトウェア開発は、一社で全てを実行しても失敗する危険性が高いのにもかかわらず、どのようなリスクがあるのかも深く考えず、単にコスト競争の旗印のもとに、複数社に分割発注することは危険きわまりのない行為だと言えます。工程別分業方式における主なリスクは次の通りです。

【工程別分業方式におけるリスク】

① 統合管理および統合コミュニケーションの欠落
元請会社において、下請け各社の進捗管理や品質管理をまとめる統合管理および統合コミュニケーションの実行がなされにくいこと。

② 統合的プロジェクト体制の構築失敗
ベンダーも含めた各社における開発力、すなわち必要な人材や体制の投入状況に関する検証がなされにくいこと。

③ ドキュメント・ベースの開発の喪失
ベンダーも含めた各社における、完全な要求仕様書や設計書などのドキュメントに基づく開発および評価の実行がなされにくいこと。

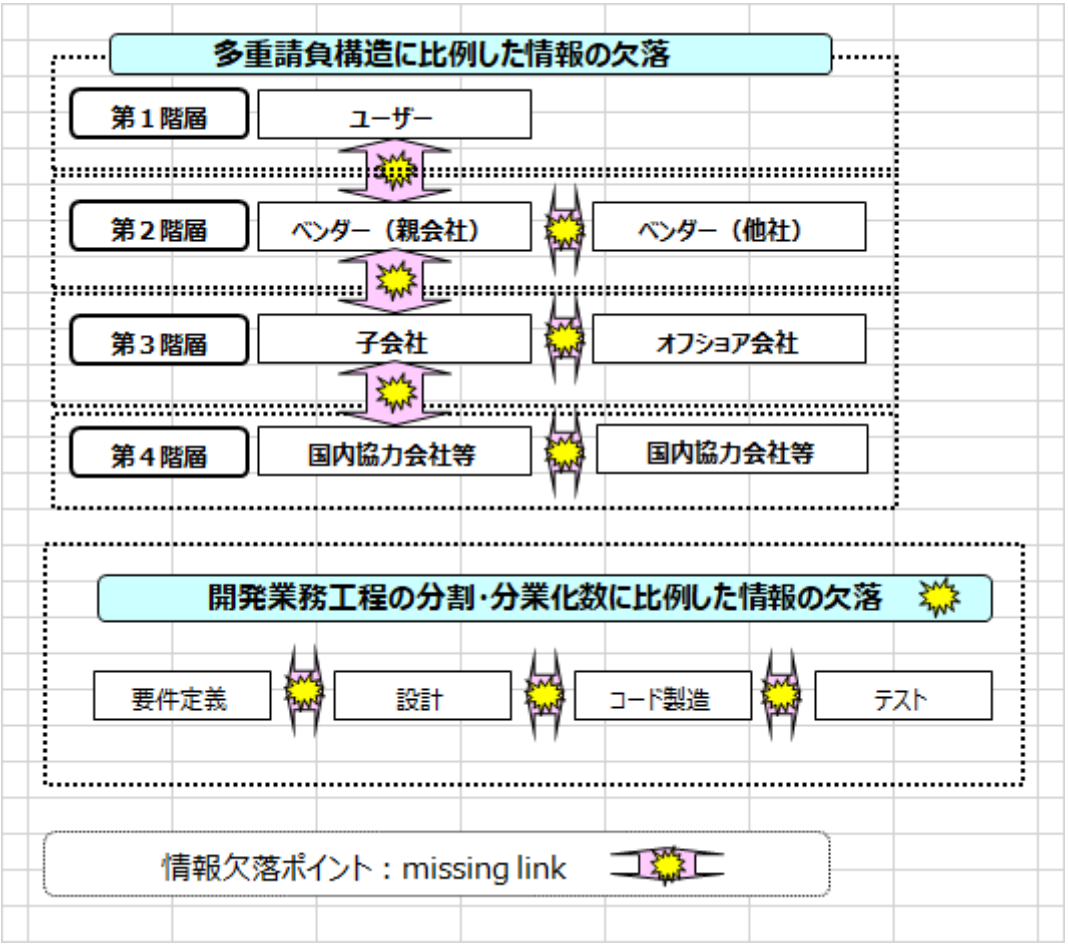
ベンダー側において、上記三点のリスクが解消されることによって、各社・各担当者は本当の意味での協業が可能になり、プロジェクトを成功に導くことができるでしょう。ベンダーの下に有機的に統合された分業体制を共有分業と呼ぶならば、そうでない分業体制は悪しき分離分業と言えるでしょう。

上記三点に関しては、分業の有無にかかわらず、一つの開発組織内においても本来実行されなければならない、プロジェクトを成功させる重要な要素だと言えます。

下記は分離分業によって生じるコミュニケーションのギャップ[124]を示したものです。

この図を見ても分かるように多重請負[125]の階層が増えるごとに、また工程間の分業が進むごとに情報共有は困難になり、情報のモレや誤解が多発することになります。このような状況を十分に認識し、プロジェクトにおける統合管理および統合コミュニケーションを着実に実行する必要があります。

【コミュニケーション・ギャップ】（図３－ｈ１－１）



[124] **コミュニケーション・ギャップ**（Communication Gap） 組織間・工程間で発生するコミュニケーションの断絶のこと。

[125] **多重請負** 一つのプロジェクトを元請・下請け・孫請けというような多段階の請負構造で形成していること。

リーダーシップ

リーダーシップ[126]とプロジェクトの統合管理とは不可分の関係にあります。リーダーシップとはプロジェクト関係者たちの統合管理に他なりません。プロジェクトチームにおけるリーダーシップとは、メンバーたちの先頭に立って道を切り開き、障害物を取り除き、成功を阻害する者たちの盾となり、進むべき方向を指し示し、チームのメンバーが最大限の力を発揮できるようにすることだと言えます。メンバーたちの後方に立って全員突撃と叫ぶ大将ではなく、俺に続けと先頭を駆け抜けるリーダーが求められているのです。

プロジェクトにおけるプロマネやリーダーが抱えるリスクは次の通りです。

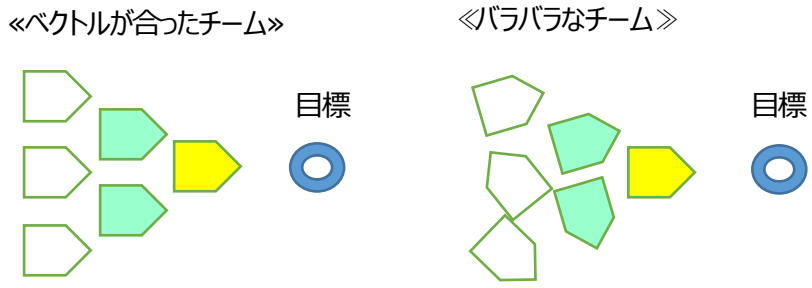
計画

【プロマネが抱えるリスク】

- ベンダー側におけるプロジェクト統合管理能力の低さ
- チームの統合管理や情報共有の弱さ
- リーダーの過重負荷
- 組織編制能力や組織統制能力の低さ
- 人間関係調整力の弱さ
- 経験不足・知識不足がもたらす柔軟性の欠如
- メンバー育成の困難さ

プロジェクトの統合管理にはさまざまな問題が集約されています。前記のリスクや問題点を一気に解決することは難しいことですが、自社における弱点を認識し、それらを改善活動のテーマとして取り組み、組織的な活動として拡大していくことができれば、将来的には素晴らしい開発組織を構築することができるでしょう。

【リーダーシップ】（図３－ｈ１－２）



126 **リーダーシップ** メンバーたちの先頭に立って道を切り開き，障害物を取り除き，成功を阻害する者たちの盾となり，進むべき方向を指し示し，チームのメンバーが最大限の力を発揮できるようにすること。すなわちプロジェクトを統合管理すること。

【計画プロセスのまとめ】

本章の解説の流れをまとめると以下のようになります。

【Job Activity】

計画

【開発条件の計画化】

１．早期の仕様凍結	：何を開発すべきかを設計着手前に明確に決定する。
２．要求仕様の構造化（ＷＢＳ）	：要求仕様を機能単位に分割し構造化する。
３．顧客要求の重要度順位の設定	：要求仕様の開発優先度を明確にする。
４．見積り	：ＷＢＳに基づいて工数および開発期間を算出する。
５．スケジュールの作成	：ＷＢＳおよび優先度に従った開発スケジュールの作成。
６．プロジェクト目標の設定	：品質･コスト･納期（生産性）の目標を設定する。
７．プロジェクト体制の構築	：開発に必要な体制を構築する。
８．開発環境の手配	：開発に必要な部材の手配および環境を整備する。

９．プロジェクト計画書の作成

上記のすべての情報を集約し、開発実行の基本計画書を作成する。

計画のプロセスは、プロジェクトが具体的な開発行為の着手を可能にするために必要なすべての条件を整えるプロセスだと言えます。 １．から８．までは行為の優先順位に並べられた開発着手を可能にするための条件であり、９．で作成される「プロジェクト計画書」はこれらの全ての条件を集約したものです。

プロジェクト計画書は次の「実行のプロセス」における基本的な行動計画書となります。

【Human Activity】

計画

【プロジェクト統合管理の役割】

① 関係各社における進捗の相互調整

② 関係各社における問題・課題の相互調整

③ 関係各社間の情報共有

④ 関係各社間の協力体制および共通開発ルールの確立

おそらくプロマネの仕事の中では、この統合管理が最も難しい仕事の一つだと言えます。複数の会社に分散された各工程を、血の通ったコミュニケーションで結ぶためには、単に合理的な思考・行動だけでは不十分であり、幅の広い人間力の発揮が必要になって来ます。 経験の浅いプロマネにおいては、小規模な開発において、協力会社までも含んだプロジェクト統合管理のノウハウを徐々に蓄積していく必要があります。

第４章

実行のプロセス

概要

第４章　実行のプロセス

計画を実行に移すＱＣＤ目標の達成に必要なマネジメント活動の基本についての解説です。

- リスクの解消により、プロジェクトの成功を確かなものにする（４－１．）
- データに基づく開発の実行により、ＱＣＤ目標値の達成を目指す（４－２．）
- ドキュメントベース開発の実行により、品質目標値の達成を目指す（４－３．）
- プロフィットドリブン開発の実行により、コスト・利益目標の達成を目指す（４－４．）
- 優先順位ベース開発の実行により、納期・生産性目標値の達成を目指す（４－５．）
- 改善活動の実行により、大幅なＱＣＤの改善を目指す（４－６．）
- 時間不足を解消するために、獲得時間の最大化と失う時間の最小化を目指す（４－ｈ１．）

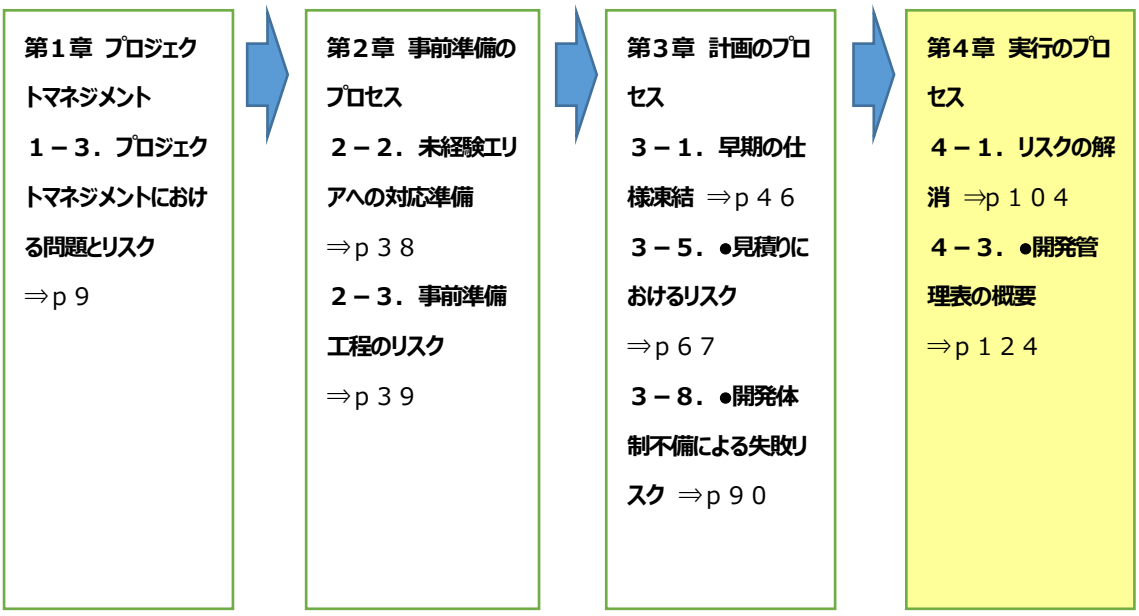
実行のプロセスは、プロジェクト計画に従って、プロマネが開発チームの指揮をとり、具体的な開発活動を実行するプロセスです。このプロセスは、主にＱＣＤ目標の達成のためのマネジメントだと言えます。

本章においては、最初にリスクの事前解消への取り組み方について検討を加えたあとで、開発をスムーズに進めるための基本的な方法であるデータドリブン開発（データに基づいた開発行動）、ドキュメントベース開発（ドキュメントに基づいた開発行動）、プロフィットドリブン開発（利益志向型の開発）および優先順位ベース開発、の四つの方法について順に説明を行います。最後に、プロジェクトのＱＣＤに決定的な影響を与える改善活動の実行、および最大の問題である時間不足の解決法について述べていきたいと思います。

【Job Activity】

4－1 リスクの解消【リスクマネジメント】

リスクは開発の全工程に渡って散在しており、第３章までのプロセスにおいて触れたリスク項目は次の通りでした。



第１章においては、「１－３．プロジェクトマネジメントにおける問題とリスク」においてプロジェクトマネジメント全般におけるリスクについて触れました。
第２章においては、「２－２．未経験エリアへの対応準備」および「２－３．事前準備工程のリスク」、
第３章においては、「３－１．早期の仕様凍結」、「３－５．●見積りにおけるリスク」、
「３－８．●開発体制不備による失敗リスク」において、要件定義および見積りなど開発の初期工程におけるリスクについて触れました。
本章においては、すでに触れたこれらのリスクも含め、開発の実行プロセス中に内在するリスクを中心として、開発全体のリスク対応に関するまとめを行います。

実行

リスクと課題の違い

リスクと課題[127]の違いについて質問を受けることがありますが、この二つについて改めて定義をしておきます。

リスクとは、現時点では問題として現れてはいないが、今後問題として現れる可能性のあるものです。

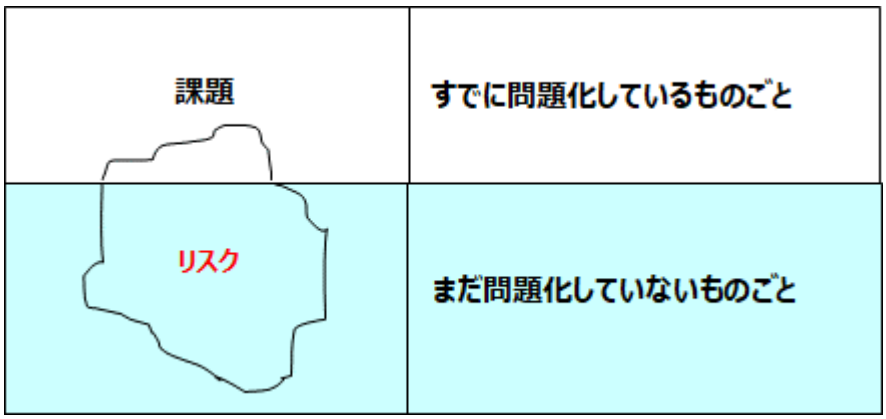
一方、課題とは、現時点ですでに問題化してしまっている問題を指しています。

リスク管理表[128]と課題管理表[129]の二種類の管理表が存在する理由はこの違いによるものです。問題になる可能性のあるリスク項目と既に問題化している課題項目を一つの管理表で管理することは不適切であり、まだ問題化していないからリスク対応はとりあえず良いなどという勝手な判断の原因になってしまうこともあります。

実行

◎リスクとは、現時点では問題として現れてないが、今後問題化の可能性のあるもの。
課題とは、現時点ですでに問題化してしまっているもの。

【リスクと課題】（図４－１－１）



127 **課題** 現時点ですでに問題化してしまっているものごと。

128 **リスク管理表** そのプロジェクト特有の想定リスクおよび開発組織で繰り返し起こしている障害やミスなどの過去の失敗を中心としたリスクを記載した管理表で、開発の工程ごとのリスク管理表が必要とされる。

129 **課題管理表** すでに表面化している問題をその期限までに解決するための管理表。

実行

プロジェクトにおけるリスクの探し方

リスク管理表の作成にあたって、自分のプロジェクトにおいては全てリスクだらけで何が本当にリスクなのかよく分からないという声を聞くことがあります。その結果リスク管理表は作成されず、リスクが問題化して初めて対応をとるというような事態に陥る場合が少なくありません。

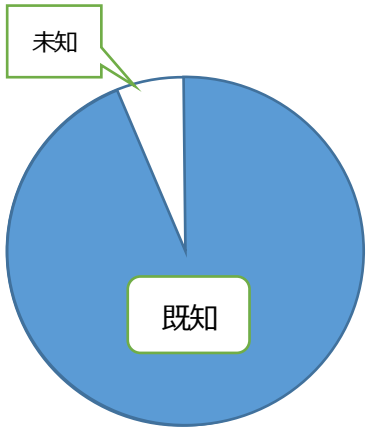
このような事態を避けるためには、リスクには既知のリスクと未知のリスクの二通りがあるということを知っておく必要があります。既知のリスクとは過去の失敗の履歴そのもののことです。未知のリスクとは今まで経験したことがないような新技術や複雑な仕様に関するリスクのことです。本書の対象である中小規模の派生開発プロジェクトにおけるリスクの９９％は既知のリスクだと言っても良いでしょう。

まずは自分の開発組織で繰り返し起こしている障害やミスなどの失敗について、顧客および開発双方においてダメージの大きなものから順に記録をしていけばリスク管理表は容易に作成することができます。

実行

◎リスクには既知のリスクと未知のリスクの二通りがある。

【既知のリスクと未知のリスク】（図４－１－２）



実行

三つの時点で押さえるリスク

プロジェクトにおけるリスクは全工程にわたって存在していますが、各工程に均等な割合で散在しているわけではなく、ほとんどの重大なリスクは要件定義および見積り工程に集中していると考えて間違いないでしょう。とりあえず開発工程を前・中・後の三つに分けてどのようなリスクが散在しているのかを見てみることにします。解消すべき三つの時点のリスクは以下の通りです。

◆開発の入り口で押さえるべきリスク

- □ 適正な見積りにて妥当な開発期間と開発費を獲得すること。
- □ あいまいな要求仕様の明確化および早期の仕様凍結に全力を注ぐこと。
- □ 要求仕様書を常時メンテナンスし、"使えるドキュメント"として設計に引き継ぐこと。
- □ ＱＣＤの数値目標の設定を行うこと。

◆開発中に押さえるべきリスク

- □ あらゆる無駄の排除とリスク項目の解消を実行すること。
- □ 設計書をリアルタイムでメンテナンスし、常にプログラム内容との同一性を保持しておくこと。
- □ 失敗の真因・再発防止策をドキュメントとして残すこと。
- □ 創意工夫を設計書やガイドラインなどのドキュメントに残すこと。

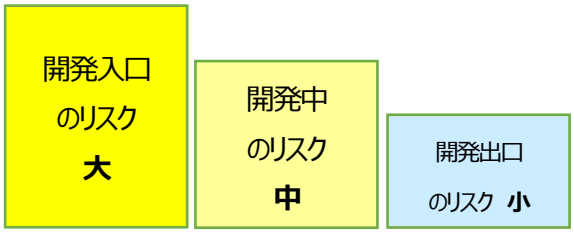
◆開発の出口で押さえるべきリスク（ラップアップミーティング[130]の実行）

- □ＱＣＤの目標値と実績値を比較・分析し、問題の原因を数値データと共に明確にし、対策した結果をまとめておき、プロジェクト完了報告書にて他チームとも情報共有を行うこと。
- □ 失敗あるいは創意工夫の記録を振返り、次の開発および他の開発者に申し送ること。

実行

◎リスクの三つの押さえ所、開発の入り口・中間地点・出口。

【三つの時点のリスク】（図４－１－３）



130 **ラップアップミーティング** 開発完了時に行われる、開発行為の全体を振り返る会議。

開発現場にころがるリスクの事例

リスクの察知は、プロジェクト自体の情報のみならず顧客の情報および全ての関係する組織の情報収集によって行われますが、特に有効なリスク情報は関係者たちとの直接コミュニケーションから得られる場合が多いものです。

突然遅延する進捗報告、肝心な要件が決まっていないこと、残っているはずの予算がないこと、突然の仕様の追加や変更、メンバーの突然の長期欠勤など、開発プロジェクトにおけるリスクにはさまざまなものがありますが、多くのリスク問題は結局コミュニケーションに起因する問題が多いのです。これらのリスク回避には毎日の積極的なコミュニケーションが必要不可欠です。

下記は現場におけるリスクの事例を１７のカテゴリーに整理分類したものです。

非常にたくさんのリスクが存在していることにあらためて驚き、新人のプロマネにおいては、とてもこのようにたくさんのリスクを一人では回避できないと思われることでしょうが、これらのリスク回避は決して一人のプロマネが背負いきれるものではなく、サブプロマネやプロジェクト全員で取り組むべき問題です。またあなたが現在担当しているプロジェクトのリスクは、この内のいくつかのリスクに限定されるのかも知れません。

◆【現場におけるリスクの事例】

１７のリスク項目とリスク内容は以下の通りです。

①不条理な顧客要求	□ 決まらない要求仕様
	□ 無理な短納期・低コストの要求
②見積り問題	□ 不十分な仕様調査・仕様理解
	□ 見積り交渉力の弱さ
③相互義務の不履行（元請・下請間、組織間、上司・部下間）	
	□ プロジェクト統合管理の不在
	□ 説明責任の欠如（丸投げ）
	□ 情報共有・コミュニケーションの不足
④仕様・納期問題	□ 仕様決定遅れ
	□ 短納期
⑤コミュニケーション不良	□ 顧客・開発チーム間の意思疎通・情報共有の不良
	□ チーム内の意思疎通・情報共有の不良
⑥情緒的開発姿勢	□ 感情的・情緒的な思考・行動
	□ 数値目標の不在
⑦モチベーション問題	□ モチベーションの低下
⑧本質の把握ミス	□ 自分の頭で考えない自律性の未熟さ

⑨優先順位問題
- □ 割り込み作業等の優先度判断ミス
- □ 必須業務に優先順位はないという認識の欠如
- □ 納期第一優先の誤り

⑩ドキュメント・ベース開発の欠如
- □ 開発行為の科学的根拠（ベースライン）の喪失
- □ 低品質なドキュメントが招くＱＣＤの失敗
- □ 文書化能力の低下

⑪実務能力問題
- □ 知識不足（技術・仕様・システム構造知識、評価テスト知識）
- □ 仕様決定能力の低さ
- □ 設計・製造能力の低さ

⑫リーダーシップ問題
- □ マネジメントされないプロジェクト管理

⑬チームプレー問題
- □ 相互義務の不履行・相互扶助の不在

⑭手抜き問題
- □ 必要工程の中断ないしはスキップ

⑮時間認識問題
- □ 仕様凍結期限意識の欠如
- □ 期限・タイミング意識の欠如
- □ 許容時間・必要時間認識の欠如

⑯人材育成／ノウハウの継承問題
- □ 個人のスキル向上の疎外
- □ 組織能力向上の疎外
- □ プロジェクトＱＣＤの低下

⑰学習能力の欠如問題
- □ 失敗に学ばない
- □ 振り返り（ラップアップ）の未実行
- □ 改善活動の未実行

以上述べてきたように、詳細な事前調査とリスクの事前解消を行っていれば、開発の見積りを間違えたとか、仕様を間違えたとか、開発期間や費用が足りないとか、トラブルの多発などという問題も発生しにくいでしょう。

実行

設計工程におけるリスク管理表の一例を次に示します。詳細サンプルは「付録図表８．設計工程のリスク管理表」を参照のこと。

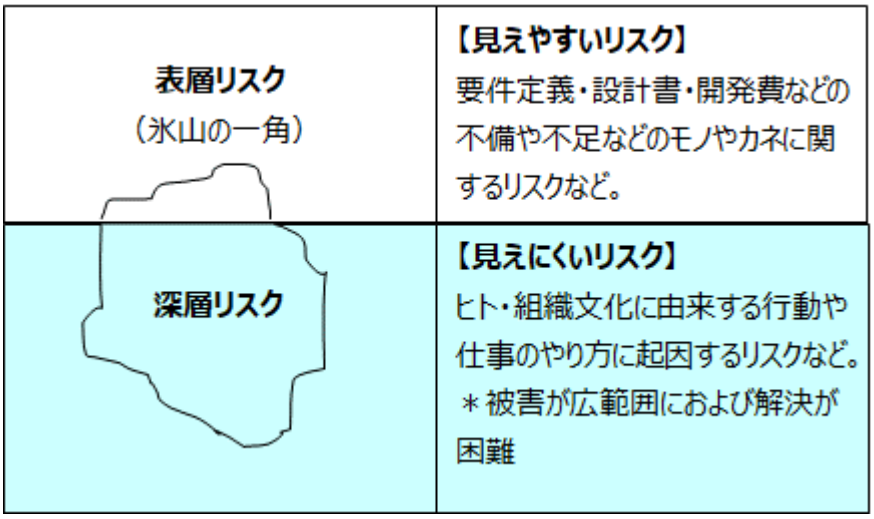
【設計工程のリスク管理表】（表４－１－１）

	プロジェクト名：				初版発行日： 更新日：		文書番号：
	要因分類	QCD影響 Q	C	D	設計工程リスク	✓	チェック 検出日
ヒトに関するリスク	①他者依存的姿勢（自律性の放棄）	○	○	○	・協力会社への丸投げ（開発統合責任・製品最終責任の放棄）		
	②上位マネジメントの関与不足		◎				
	③ユーザーの参加・協力度不足	◎	○	○			
	④組織能力不足（未熟な組織文化、戦略の欠如）	◎	◎	◎	・オフショア開発		
	⑤見積り能力	◎	◎	◎			
	⑥要件定義能力	◎	○	○			
	⑦リーダーのプロジェクトマネジメント能力（外部交渉、タイムマネジメント、現場主義、見える化能力など）	◎	◎	◎	・未経験技術者の大量投入 ・基本設計レビューの未実施 ・安易なフレームワークの改修		
	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）	◎	○	○	・フレームワーク構築能力 ・異常系技術力の不足 ・タイミング設計能力の不足 ・排他処理設計能力の不足 ・非同期制御設計能力の不足 ・リトライ設計能力の不足 ・ログ技術力・認識不足 ・オープンソフト対応技術力不足 ・メモリーリーク対応技術力不足 ・性能（パフォーマンス・レスポンス）技術力不足		
	⑨プロセス管理の有無	◎	○	○	・プロセス管理不足 ・基本設計レビューの未実施 ・性能要件（パフォーマンス・レスポンス）レビューの未実施		
	⑩コミュニケーション能力（阻害・ギャップ）	◎	○	○	・組織間の協調性・コミュニケーション		
	⑪関連部署との連携不足	◎	○	○	・協力会社・オフショア先などとのコミュニケーション能力		
モノ	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）	◎	○	○	・要求仕様書なしの設計 ・貧弱なドキュメントやガイドライン		
	⑬開発のベースの有無	◎	◎	◎			
	⑭開発環境の不備	◎	○	○			
カネ	⑮開発費不足	◎	◎	◎			
	⑯資源投入戦略の誤り（開発費投入時期ミス）	◎	○	◎			
情報	⑰あいまいな開発範囲（スコープ）	◎	◎	◎	・あいまいなスコープ		
	⑱あいまいな要求仕様	◎	◎	◎	・あいまいな要件		
	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）	◎	○	○	・情報の不備・不足		

見えやすいリスクと見えにくいリスク

第２章においては事前準備工程のリスク、第３章においては要求仕様に関するリスク、本章においては設計工程におけるリスクを取り上げてきましたが、工程別リスクの視点を見えやすいもの／見えにくいものという視点で見てみた場合を図で表すと次のようになります。

【見えやすいリスクと見えにくいリスク】（図４－１－４）



見えやすいリスクを表層リスク[131]、見えにくいリスクを深層リスク[132]として分類した場合、表層リスクは氷山の一角にしか過ぎず、実は多くのリスクが深層リスクとして存在していることに気づきます。また表層リスクの多くは深層リスクに起因しているものが非常に多いのではないかと思われます。

見えにくい深層リスクは、本書における Human Activity（人間的活動）すなわちヒトや組織文化に由来する場合が多く、一旦これらのリスクを見逃した場合の被害は広範囲かつ重大な問題を引き起こす結果になるものと思われます。

131 **表層リスク** 要件定義・設計書・開発費などの不備や不足などのモノやカネに関するリスクで表面化しやすいリスクのこと。

132 **深層リスク** ヒト・組織文化に由来する行動や仕事のやり方に起因するリスクのこと。

実行

失敗に備えるコンティンジェンシープラン

コンティンジェンシープランとは何か

コンティンジェンシープラン[133]とは不測事態対応計画とか、緊急時対応計画などと訳されていますが、予期しない失敗に直面した場合に備えるための計画のことです。

どのように綿密に考えられた計画であったとしても、計画した時点では予測できなかったリスクが問題化し危機状態に陥るプロジェクトも珍しくはありません。

この不測事態対応案が初期の計画において立案されていない原因としては、この計画でうまく行くはずだという思い込みや、この計画で行くしかないという戦略的思考[134]の狭さや、このような代替計画を考えること自体が自分の計画に自信がないと他人に思われたくない、というような体面ばかりを重んじる情緒的な考え方や、単に面倒くさいという怠惰性にあります。

◎コンティンジェンシープランとは、予期しない失敗に直面した場合に備えるための計画のこと。 実行

具体的なコンティンジェンシープランの例

コンティンジェンシープランはプロジェクトの達成目標であるＱＣＤを支えるヒト・モノ・カネ・時間・情報に対して計画しておく必要があります。下記にそれぞれについて列挙してみます。

【ヒト】

１．一日 24 時間をフルに使うため、２交代制あるいは３交代制の可能性について考えておく。
開発機器類の夜間休止時間をなくすことや開発要員の負荷の平準化を図る効果があります。

２．開発のボトルネック[135]になりそうな人材の予備投入に備えて、レスキュー能力のある他の組織や人材の目星をつけておく。特にプロジェクト全体の指揮をとるプロマネ人材の増強は、開発チーム全体がデッドロック[136] 状態になった場合に大きな効果を発揮します。単に設計者やプログラマーの増員では乗り切れない場面も多々あることに注意する必要があります。

３．代替がきかない要員、とくに一人だけのプロマネや開発のキーマン[137]が不慮の病気や事故で欠員となった場合の代替要員について他部署・協力会社等広く人材の目星をつけておく。

４．万が一の障害発生に備えて、年末年始や連休中における待機計画を作成する。

---

133 **コンティンジェンシープラン**(Contingency Plan) 予期せぬ事態に備えて、予め定めておく緊急時対応計画・不測事態対応計画。

134 **戦略的思考** 計画を成功させるための方策を考えること。

135 **ボトルネック**（Bottle neck） 進行の障害や妨げとなるもの。

136 **デッドロック** 行き詰ってしまった状態。いわゆる、にっちもさっちも行かない状態のこと。

137 **キーマン** プロジェクトの成否のかなめとなる中心人物。 キーパーソンとも言う。

実行

【モノ】

１．予備の開発機器・テスト機器の確保の可能性の検討を行う。

機器の不足による生産性の低下に効果があります。

２．顧客交渉により成果物の優先順位の見直しを行う。

顧客に約束した成果物は約束した日にすべてリリースすべきですが、それが不可能だと判断された場合は速やかに顧客に報告すると同時に、顧客と相談の上、再度成果物の優先順位の見直しによる分割リリースの道を探らざるを得ないこともあります。一定のペナルティを覚悟する必要があるでしょう。

３．システム移行時の万一の失敗に備えて、旧システムに戻すための手順を考えておく。

【カネ】

１．予備資金を用意しておく。

２．戦略的リスク物件プロジェクトにおいては、経営層からの支援資金の約束を取り付けておく。

【時間】

１．余裕日・余裕時間を計画に織り込んでおく。

２．開発進捗遅延発生に備えて、延期が可能な業務、中止が可能な業務の目星をつけておく。

【情報】

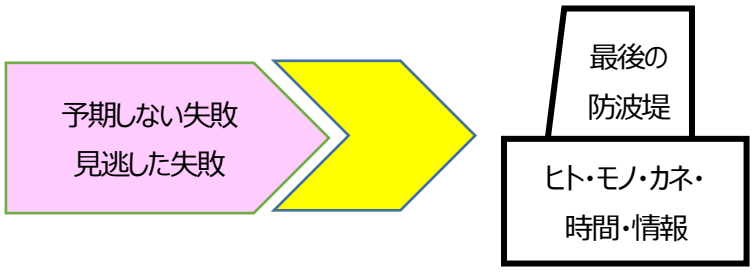
１．情報漏洩に備えたセキュリティ対策を講じておく。顧客データ・開発データを持ち出せない仕組み、外部からの未承認データを持ち込ませない仕組みの構築を行っておく。

リスクのないプロジェクトなど存在しませんが、中小のプロジェクトはリスクが低いと見くびっていると、思わぬ失敗に見舞われないとも限りません。ソフトウェア開発全般においてどのようなコンティンジェンシープランがあるのかについて知っておけばきっと役に立つ機会があるでしょう。

実行

◎コンティンジェンシープランは、あなたのプロジェクトを救う最後の手段となる。

【コンティンジェンシープラン】（図４－１－５）



実行

4－2 ＱＣＤ目標値の達成☆データドリブン開発の実行

【ＱＣＤマネジメント】

データドリブン開発とは何か

データドリブン開発とは、開発の推進にあたってデータを根拠にした活動を推し進めるという考え方です。

データを根拠にするとは、事実を証明する数値に基づいた資料や論拠を使うと言うことです。具体的には開発における目標の設定、問題の分析、および開発行為のすべてに必要なコミュニケーションや、要件定義書・設計書などのドキュメントの記述内容は、全て数値・数字の裏付けが必要だと言うことです。

難しく複雑な開発という仕事を無事に完了させるためには、感情論を排し、合理的な方法が必要になります。合理的な仕事のやり方の基本が、データに基づくやり方およびドキュメントに基づいたやり方です。数値で示されるものでなければ、目標も結果も適切に判断することはできません。

データに基づいて開発を遂行するということは、自分ないしは自チームの過去の QCD 等のデータおよびそれらから導き出された目標値データに基づいて仕事をするということです。データに基づかない仕事のやり方は、出たとこ勝負のやり方でプロの仕事とは言えません。

実行

◎基本的なデータはＱＣＤのデータ。

データドリブン開発の基本的なプロセス

開発の基本的な数値はＱＣＤに関する数値です。開発チームにおける現在のＱＣＤの実力値を知り、目標とするＱＣＤの値を設定し、プロジェクト終了時に達成されたＱＣＤの数値の振り返りを行うことがデータドリブン開発の基本的なプロセスだと言えます。

プロジェクトを開始する前に、過去のプロジェクトにおける品質に関するデータを収集しておく必要があります。この過去のデータの平均値が現時点における開発チームの実力値を表わしています。これまでの失敗の反省に基づき、改善対策を実行した結果、得られる期待成果[138]としての数値がプロジェクトの目標値となります。この目標値はプロジェクトの終了時に得られた実績数値との比較を行うことで、次のプロジェクトにおける目標指標を算出することができます。

実行

◎過去のデータ、目標とするデータ、結果のデータの三つでドライブする。

ＱＣＤの目標値の設定および具体的な指標例については、「３－７．プロジェクト目標の設定」におい

138 **期待成果** 達成したいと考えている目標数値。

て触れた通りですが、本章においてはそれらの目標値を達成するために必要な活動について解説をします。

データドリブン開発を構成する三つの要素

データに基づく開発は、次の三つの開発コンセプト[139]において推進されることで、ＱＣＤ目標値の実現を確かなものにします。

１．ドキュメントベース開発[140]	Ｑ：品質目標値の達成
２．プロフィットドリブン開発[141]	Ｃ：利益・コスト目標値の達成
３．優先順位ベース開発[142]	Ｄ：納期・生産性目標値の達成

それぞれの開発コンセプトの実行はＱＣＤのすべてに好影響を与えますが、上記はＱＣＤのどれに強力に効果をあらわすのかということを表わしたものです。

続く、４－３、４－４、４－５の各節において、ＱＣＤの順に上記三つの開発コンセプトの実行について説明を行います。

データドリブン開発の効用

データに基づいた開発をすると言うことは、最短時間・最小エネルギーで物事を成し遂げる最善の方法として、合理的なやり方を採用するということです。合理性とは、物事の筋道を数値によって明らかにし、その数値の意味するところによって現状を把握し、それに従った行動を行うということです。日常業務の多くの物事は数値によって表現が可能であり、合理性の効用として以下のものが考えられます。

【合理性の効用】

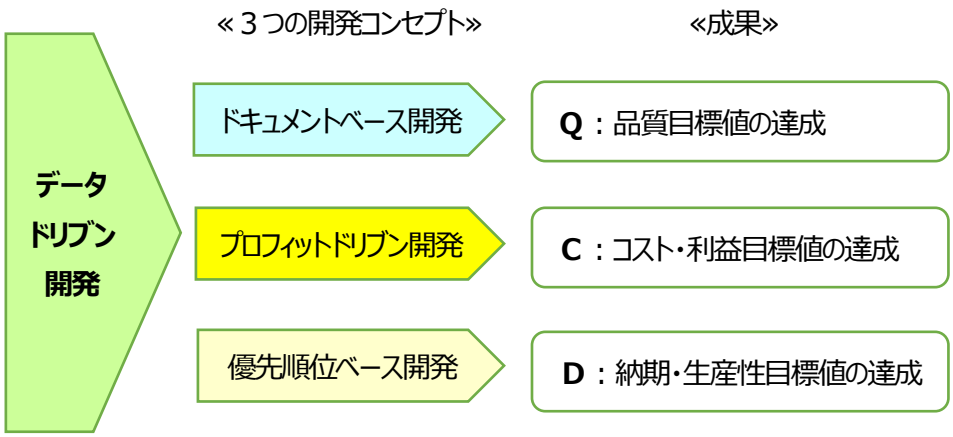
① 種々に異なる問題の解釈を、だれの誤解も招かない一つの形式で表すことができる。
② 問題の量や質など、度量衡に関する全てのことを具体的かつ明確に示すことができる。
③ 数値化された原因や結果は、衆人の検証に耐えうる証拠となり得る。
④ 数値化することで見えない事象、すなわち見えない原因や結果を見えるようにできる。
⑤ 最短の時間と最小のエネルギーで問題を解決できる。

多くの問題は、物事を数値・数理で表すことによって解決可能です。特に開発上の問題は、品質・コスト・時間に関するデータや数値の把握によらなければ絶対に解決することはできません。数値でとらえるべき問題を情緒的な判断でとらえようとしても適切な解は得られず、時間とエネルギーの無駄使いとなるだけです。

---

139 **コンセプト** ものごとの基本的な概念を言い表したもの。ものごとの骨格を表わす考え方。
140 **ドキュメントベース開発** 口頭だけに依存せず、書類・書面に基づいて実行される開発。
141 **プロフィットドリブン開発** 利益目標の達成を主眼として、同時にコスト目標の達成を実現する開発。
142 **優先順位ベース開発** 顧客価値の高い要求仕様の開発を優先させ、不要不急な開発を避けること。

【データドリブン開発】（図４－２－１）



4－3 品質目標値の達成☆ドキュメントベース開発の実行

【品質マネジメント】

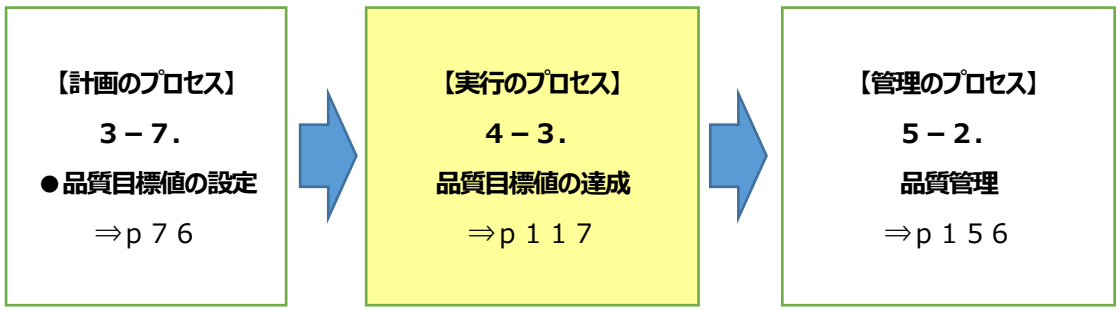
ドキュメントベース開発とは、開発業務のあらゆる面において、必ず何らかの書類・書面に基づいた情報の伝達や情報の共有を実行するという考え方のことを指しています。

あらゆる構造物を作るためには設計図が必要です。ソフトウェアの開発も当然のことに設計図に基づかなければプログラムは作成できません。

品質目標値の達成は、主にこのドキュメントベース開発の実行および後述の「４－６．改善活動の実行」によって実現されます。

品質目標値の達成

計画のプロセスの３－７．において、品質目標値の設定について解説を行いましたが、本章においてはその目標値達成のために必要なプラクティスについての解説を行います。品質に関する解説の流れは下図のようになっています。



どのような目標値であれ、それを達成するために必要なことは以下の３点だけです。

実行

◎品質目標値達成のための３原則

① 現在の品質状態を数値で表すこと。

② 改善の目標値を決めること。

③ 目標値の達成に必要な改善活動を実行すること。

実行

業務品質（ヒトの品質）の向上

プロジェクトの業務能力を高めるためには、プロジェクトの構成員である開発者個人の業務能力を高める必要があります。そのためには個々の開発者における現在の業務能力のレベルを把握し、その改善目標値を設定し、必要な改善活動を実行する必要があります。

ヒトの業務品質を数値で把握するために適した指標としては、下記の二つが一般的なものです。

①業務実行ミスの件数および率　　②業務内容の実行件数および率

第３章で取り上げた業務品質の指標例は次の通りでした。

①仕様品質、②設計品質、③製造品質、④評価品質、⑤構成管理品質、⑥プロセス管理品質、⑦リスク管理品質、⑧課題管理品質、⑨各レビュー品質

これらの指標値は、チーム全体の値および個人のデータの両方を収集する必要があり、それぞれの弱点を明確にし、それらを改善するための対策を組織的な改善活動として実行する必要があります。改善活動を個人レベルに任せていては、ほとんど改善は実行されないと思ったほうがいいでしょう。

また個人のデータに関しては「個人スキル管理票」などとしてまとめておけば、個人のスキル育成計画やキャリア形成計画の資料として活用できます。いずれの業務品質の指標も、ソフトウェア業界全体を網羅できるような統一的基準値は公表されていませんので、自組織における過去・現在の実績値の変動の比較によって判断する必要があります。

製品品質（モノの品質）の向上

チームおよび個人の業務品質が向上してくれば当然のことに製品の品質も向上してきます。

製品の品質は、ヒトの業務品質が製品の上に現れてきたもので、品質状況を端的に表すのに適した指標だと言えます。

また品質の改善対策を検討する場合、業務品質の指標と製品品質の指標の両面から問題の真因を追求すれば、より具体的な対策案にたどり着くことができます。

例えば、製品品質の単体テスト品質に問題があった場合、単に単体テストのやり方だけに原因を求めるのではなく、関連する業務品質である仕様品質・設計品質・製造品質も含めた原因追及が、真因をつきとめるのに役に立ち、適切な改善活動につなぐことができます。

第３章で取り上げた製品品質の指標例は次の通りでした。

①単体テスト品質、②結合テスト品質、③総合テスト品質、④市場品質

実行

◎すべての品質指標は改善活動のテーマであり、
業務品質も製品品質も、組織的な改善活動を実行しなければ上がることはない。

実行

ソフトウェア開発に必要なドキュメント

開発におけるドキュメントには主に、設計関連、開発管理関連、およびビジネス関連の三種類があります。

設計関連ドキュメントには、何を作るかを示した要件定義書（要求仕様書）、どういう風に作るのかを表した設計書やガイドラインなどが含まれます。また開発管理表は、設計書にて定義されたものを完成させるためのリスクの排除・課題の解決・開発の段取り・進捗を管理するためのもので、リスク管理表、課題管理表、進捗管理表、予算管理表などがあります。以下に一覧で示します。

【設計ドキュメント】

- □ 要求仕様書（要件定義書）
- □ 設計書：　基本設計書、詳細設計書、業務運用フロー、データフロー 、システム論理構成図、プロセスフロー、ソフト構造図、修正影響度表、インターフェース仕様書、など。
- □ 手順書：　見積りガイドライン、プロセスガイドライン、設計手順書、コーディング規約書、評価手順書、構成管理手順書、など 。
- □ ソースコード（コンピュータ言語記述文）
- □ 技術メモなど。

【開発管理ドキュメント】

- □ 計画書：　プロジェクト計画書、テスト計画書、など。
- □ 開発管理表：プロセス管理表、リスク管理表、課題管理表、予算管理表、進捗管理表、労務管理表、機材管理表、成果物管理表、障害管理表、など。

【ビジネスドキュメント】

- □ 見積書
- □ 企画提案書

実行

◎整備状況・精度レベルの確認が必要な３種類のドキュメント

①設計ドキュメント ②開発管理ドキュメント ③ビジネスドキュメント

実行

開発ドキュメントの現状

開発者たちはドキュメントの現状について次のように語っています。

【開発者たちの生の声】

☆ システムの全体を表した資料がない。
☆ 品質は上流工程の要件定義書や設計書で決定する。テストだけでは品質は改善しない。
☆ 客先要件の実現方法が明確に書かれていない。
☆ 客先要件や仕様内容の記述が乏しく、ロジック記述に偏っている。運用テストに使えない。
☆ 開発内容に対するコンセプトや背景や経緯についての資料がない。
☆ 機能を実現するために必要な情報が設計書に記述されていない。
☆ 誰のために書かれたものか分からないドキュメントが多い。
☆ 作成者以外が見ても分かる内容・レベルになっていない。
☆ 内容が更新されていない。
☆ 納期が間に合わなくなったら、テスト作業やドキュメントを省略してしまう。
☆ 整理・管理されていないため調査・検索ができない。
☆ モジュール間の連携部分の説明が貧弱。
☆ 障害解析に使用できるレベルの内容になっていない。
☆ 障害報告書で技術的側面から図表等を使用した報告ができていない。
☆ 既存ソフトの流用の可否を判断できる資料になっていない。

このような状況を招いている原因については次のように語っています。

【できない理由】

☆ 前任者がドキュメントを更新していなかったから、自分もできなかった。
☆ 協力会社に設計以降の工程を丸投げする方が安く・納期短縮もできたから。
☆ 発注元が外注に丸投げする状況が続き、発注元の担当者が自ら設計する技術がなくなってきた。
☆ 発注元からドキュメントを要求されることもなく、ドキュメント作成工数が削減できた。
☆ バージョンアップ開発時にはソースベースで調査・改修も何とかなった。
☆ 時間に余裕がなく、とりあえず自分が分かるレベルの内容記述だけになった。客先・ＳＥや他の開発者・協力会社が理解できる内容にはなっていない。
☆ 時間も人もなく、限定した内容だけの記述しかできず、更新が必要な資料にも手をつけられなかった。

◎プロマネは自分のチームの本当の状況を現物確認した方がよい。

実行

実行

ドキュメントの不良が招いた失敗事例

貧弱なドキュメントがどのようにひどい状況をもたらすのかを知るために、ＩＰＡ・ＳＥＣ[143]が公開しているプロジェクトの失敗事例の中からドキュメントの不備による失敗事例を紹介します。

①「正式な要求仕様書がなく仕様検討を始めたが、二転三転する要求により変更が多発した。
短納期のため、本来開発着手前に提示、または合意すべき要求仕様書がなく、また、ベンダーも強く要求していなかった。その結果、些細な点で要求仕様のズレが発生。完成度の低いシステムとなった。」

②「開発ガイドラインがないために保守性、流用性が悪化してしまった。
開発ルール、特にプログラミングの標準化、構造化のルールが未確定、またはメンバーに未浸透なため、プログラムが分かり難く、問題発生時の修正困難に加え、追加変更、移植性に多くの費用と工数を要した。」

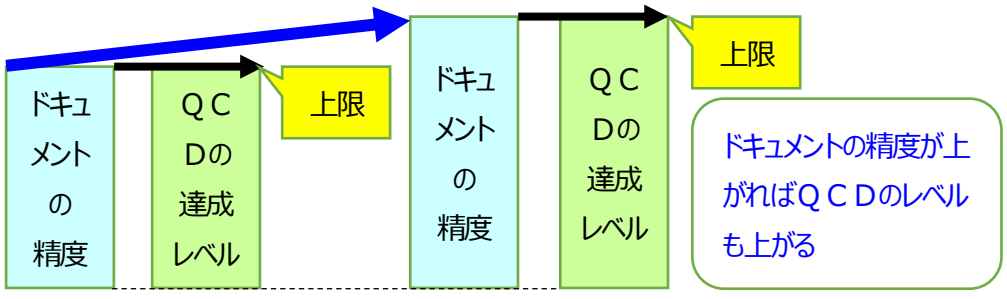
③「手抜きドキュメントによるオフショア開発
概要設計だけを示して委託したが、海外ソフトハウスはまだ設計できるほどのレベルにまでなっていなかった。そのため、海外ソフトハウス側でも勝手な思い込みで作り込みを続けた。プロジェクト側も、スキル不足で上がってくる中間成果物を十分に把握できず、システム規模が増え、作業遅れが発生した。」

④「計画書も手順書もない評価業務
作業手順の誤りやテスト・データの誤りによる再テスト・修復という無駄な作業が多発し、テスト用マシンのCPU 時間の半分以上を消費していた。これにより、テスト作業も遅延していた。」

実行

◎妥当なレベルのドキュメントなしでは、ＱもＣもＤも達成することは不可能。

【ドキュメントの精度とＱＣＤの達成レベル】（図４―３－１）



143 **ＩＰＡ・ＳＥＣ** 独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）の略称。

ドキュメント精度の簡易チェック法

基本設計書の簡易チェック方法

簡易的に基本設計書の完成度をチェックする方法として、その基本設計書を使ってある重要な仕様の一部分について詳細設計書を書いてみればはっきりするでしょう。ちゃんとした詳細設計書が書けなければ、その基本設計書はまだ不完全だということです。詳細設計が書けない部分が、基本設計書における欠落部分や矛盾している部分なのです。

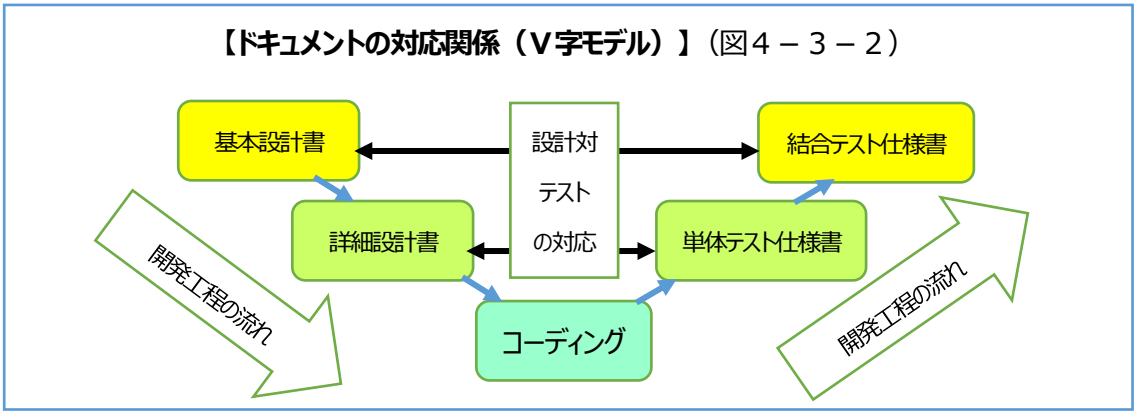
また基本設計書は要求された機能の基本的な骨格を定義した設計図であり、基本設計書と対をなしている仕様書としては結合テストにおいて使用される結合テスト仕様書があります。結合テスト仕様書は開発されたソフトウェアの基本的な機能が正しく動作することをテストするためのドキュメントです。基本設計書の一部分について結合テストの担当者に、結合テスト仕様書が書けるかどうかを試してみても基本設計書の精度が確認できるでしょう。

詳細設計書の簡易チェック法

簡易的に詳細設計書の完成度をチェックする方法として、その詳細設計書を使ってある重要な仕様の一部分についてプログラムを書いてみればはっきりするでしょう。ちゃんとしたプログラムが書けなければ、その詳細設計書はまだ不完全だということです。プログラムが書けない部分が詳細設計書における、欠落部分や矛盾している部分なのです。

また詳細設計書は要求された機能を最小機能単位に分割し、それを定義した設計図であり、詳細設計書と対をなしている仕様書としては単体テストにおいて使用される単体テスト仕様書があります。単体テスト仕様書は開発されたソフトウェアの最小機能単位が正しく動作することをテストするためのドキュメントです。詳細設計書の一部分について単体テストの担当者に、単体テスト仕様書が書けるかどうかを試してみても詳細設計書の精度が確認できるでしょう。

【ドキュメントの対応関係（Ｖ字モデル）】（図４－３－２）



実行

リアルタイムなドキュメントのメンテナンスが必要な訳

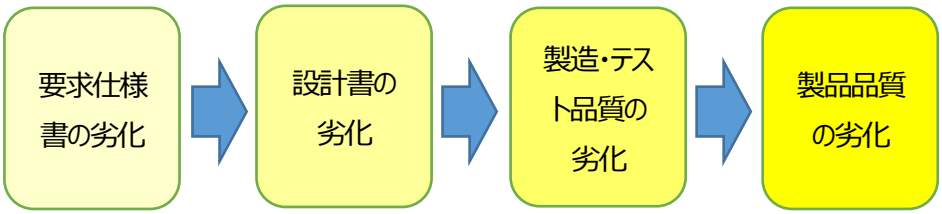
黙っていてもドキュメントは更新されません。更新されていないドキュメントの内容は何が「正」で、何が「誤」なのかを判断することができません。段々とメンテされなくなったドキュメントは、ついには誰にも使用されない大量の紙クズとなってしまいます。開発ドキュメントは常にその鮮度を保っておく必要があり、生鮮食品の一種だと思った方がよいのです。常に鮮度を保つ管理をすることが、不要な労力を省き、スムーズな開発を実現することにつながります。

信頼できる設計書がなければソースコードの解析による開発となり、まちがいなくその設計品質は劣化します。設計品質が劣化すれば製造品質も評価品質も劣化します。いつまでたっても設計書は更新されないまま、品質劣化の悪循環が繰り返されることになります。設計書の前の要求仕様書がなければもっと大変な結果となります。

実行

◎ドキュメントの誤記・抜け・追加は、気づいた時点でリアルタイムな修正が必要。

【ドキュメントの劣化が招く、品質の劣化】（図４－３－３）



開発管理表の概要

【リスクマネジメント】

プロマネにとって開発管理表はプロジェクトの進行をコントロールする重要なドキュメントです。
下記は主要な開発管理表についての役割を整理したものです。

【開発管理の三種の神器】（図４－３－４）

◎**プロセス管理表**	主要イベント・成果物の実行チェック
◎**リスク管理表**	全工程にわたるリスクの掘り起こしと解消
◎**課題管理表**	日々発生してくる課題対策の立案と解決

プロセス管理表

プロジェクト管理の基本中の基本はプロセス管理です。プロセス管理は分かりやすく言うと仕事の段取りであり、プロセス管理表はプロジェクトの重要なイベントを時系列順に並べた開発手順のチェックシートだともいえます。各々の詳細な工程に対して、やるべき作業内容、誰が実行するのか、作成される成果物は何か、などを記述し、実施予定日・実施日や備考などを記録することで、個々の作業が適切な内容で確実に実施されたことを保証する証拠となるものです。

第１章の「１－４．プロセスをマネジメントする」および「付録図表２．プロセス管理表」を参照のこと。

各工程が複数の会社にて分業化されている場合は、各工程の担当会社ごとに担当工程のプロセス管理表を作成・運用し、発注元の会社においては全体をまとめた統合プロセス管理表の作成・運用が必須です。

リスク管理表

リスク管理表は開発全工程にわたるリスクを管理するものです。プロジェクトにおけるリスクとしては次のようなものがあります。

仕様凍結遅れ、仕様変更、進捗状況、責任の所在、開発規模、他組織との関連、要員の確保、外注コントロール、短納期、プロセスの遵守、見積り精度、性能問題、保守性問題、プログラム構造、設計ミス、製造ミス、評価ミス、構成管理ミスなど。

これらの情報を網羅したリスク管理表の作成およびリスクの解消はプロジェクトの成否を分ける必須の業務です。

第４章の「４－１．リスクの解消」および「付録図表４、６、８の各リスク管理表」を参照のこと。

課題管理表

リスクが実際に問題化したものが課題です。課題管理表は、日々発生してくる課題をその対策アクションとともに記録するための管理表です。課題管理のポイントは、効率的な課題解決のために各々の課題に優先度をつけること、およびその実行を確実なものにするために実行担当者名および期限を明記することです。

課題とリスクの違いがよく分かってない人が意外と多いのですが、リスクはまだ問題が表面化してはいないが表面化する危険性のある潜在的な課題のことです。リスクはそれが表面化する前にその芽を摘み取っておくような対策や行動が必要です。例えば仕様凍結が遅れそうな心配があるなら、こちらから先に仕様提案をするとか、仕様を決める責任部署に人員強化の依頼を出しておくとか、をすることがリスク回避のアクションとなります。

一方課題は必ず解決されなければならない問題です。課題はあなたの都合を待ってはくれません。期日までにその問題を解決しなければ、その問題はさらに大問題へと発展してしまい、ＱＣＤに大きな傷を負わせ、プロジェクトを止めてしまう結果を招くかもしれません。

プロマネの毎日の仕事は、課題の解決だと言っても言いすぎではないでしょう。プロジェクトにおいてはリーダーから担当者にいたるまで実に多くの課題・問題を抱えています。何も問題を抱えていませんと発言する人は、何も仕事をしていない人だと思った方がいいでしょう。課題管理表なしではまっとうに開発は、やり遂げられないものです。

【課題管理表】（表４－３－１）

下記の課題管理表の記述欄は左から課題№、優先度、警告、発生日、期限、完了日、記入者名、発生工程、課題項目名、課題内容、課題対応策、処理担当者名、対応状況、備考欄となっています。

No	優先度	警告	発生日	期限	完了日	記入者	発生工程	課題項目	課題内容	課題対応策	担当	発生作業／進捗状況／結果	備考
1													
2													
3													
4													
5													
6													
7													

サンプルの拡大版は「付録図表９．課題管理表」を参照のこと。

４－４ コスト･利益目標値の達成☆プロフィットドリブン開発の実行

【コストマネジメント】

プロマネにとって、会社の開発組織が利益志向型なのか原価志向型なのかということは非常に重要な意味を持っています。その利益についての考え方の違いが、開発組織やプロジェクトの基本的な性格や運営方針を根本的に異なったものにしてしまいます。

利益志向型の組織をプロフィットドライブ組織と呼び、原価志向型[144]の組織をコストドライブ組織と呼びます。 利益獲得を強く意識した組織としては営業組織や独立採算制組織があり、一方原価低減を強く意識した組織としては間接業務組織があります。

独立系ソフトウェアハウス[145]やシステムインテグレータ[146]における開発組織はプロフィットドライブで、ハードウェアメーカー所属の開発組織はコストドライブであるのが一般的な形態だと思われます。

コスト･利益目標値の達成は、プロフィットドリブン開発の実行および後述の「４－６．改善活動の実行」によって実現されます。

実行

◎利益志向型はプロフィットドライブ組織、
原価志向型はコストドライブ組織。

コスト･利益目標値の達成

計画のプロセスの３－７．においてコスト･利益目標値の設定について解説を行いましたが、本章においてはその目標値達成のために必要なプラクティスについての解説を行います。コスト･利益に関する解説の流れは下図のようになっています。

【計画のプロセス】
３－７．●コスト･利益目標値の設定
⇒ｐ８０

→

【実行のプロセス】
４－４．コスト･利益目標値の達成
⇒ｐ１２６

→

【管理のプロセス】
５－３．コスト･利益管理
⇒ｐ１６４

---

144 **原価志向型** コスト目標の達成を中心とする考え方。売上げ拡大の志向が弱くなる。

145 **ソフトウェアハウス** ソフトウェアの受託開発やソフトウェアパッケージの開発を主要業務としている企業。

146 **システムインテグレータ** 情報システムの開発において、コンサルティングから設計、開発、運用･保守･管理までを一括請負する情報通信企業。略して SIer（エスアイヤー）とも呼ばれる。

実行

コストが下がれば利益が上がるように、両者は表裏一体の関係にありますが、コストを下げる方に目を向けるか利益を上げる方に目を向けるかで、モチベーション的には大きく異なってくるものです。始終コスト削減ばかりを言われると萎縮しがちになりますが、利益拡大を目指した方が気持ちは明るくなるような気がします。

利益・コスト目標値の例は以下の通りでした。

【利益指標】

- □ 粗利額／率
- □ 営業利益額／率

【コスト指標】

- □ ロス率、ロス削減率
- □ 直接労務費・間接労務費の比率
- □ 工程別開発費配分率
- □ 見積り時間の短縮化率・時間
- □ 見積り外作業工数削減率・金額
- □ 設計工数効率化率、製造工数効率化率、評価工数効率化率、管理工数効率化率
- □ ソフト共用モジュール登録数・再利用率
- □ ドキュメント作成時間の短縮化率・時間
- □ チェックリスト共有化率・件数

利益を拡大するためには、「売上を上げる」か「コストを下げる」に尽きますが、売上を上げるためには確かな見積り力と顧客の説得力が必要となり、コストを下げるためには失敗のロスを減らし技術力の向上による生産性の向上を図る必要があります。

実行

◎すべての利益・コスト指標は改善活動のテーマであり、
組織的な改善活動を実行しなければ上がることはない。

実行

プロフィットセンターとコストセンター

開発組織の採算の仕組みは大きく分けて、独立採算方式と予算方式の二つがあります。独立採算方式は利益確保をその組織の責任で行うもので、プロフィットセンターとも呼ばれています。一方、予算方式はコストセンターとも呼ばれるもので、その組織の費用はコスト扱いとして処理され、原価目標の達成責任は有りますが、直接的に利益確保の責任を問われることはありません。

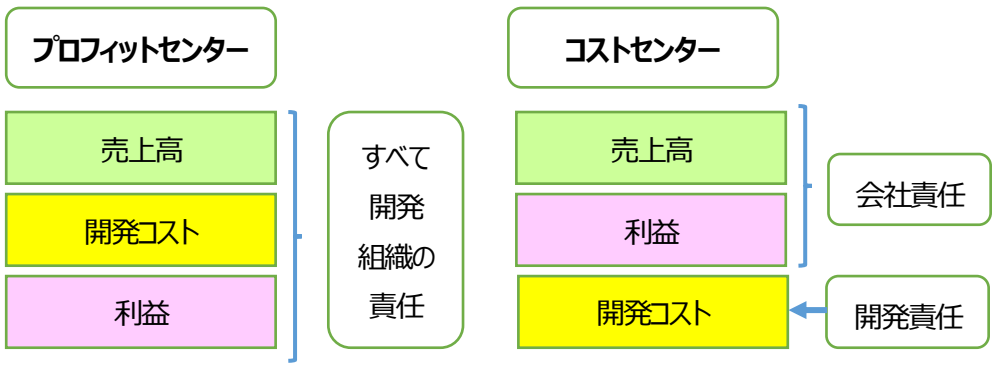
プロフィットセンター組織には大きな権限が与えられる一方、利益確保という重い責任を果たす義務があります。そのために、このタイプの組織は自律的な組織であることが求められ、厳しい競争環境におかれるため、開発者の成長にとっては良い効果を及ぼす可能性が高いものと思われます。このように自律的なプロフィットセンター組織は、必然的に将来まで見据えた競争力のある組織を目指す傾向が強くなるものと思われます。

一方、コストセンター組織は、会社においてはコスト扱いとなり利益確保の義務は発生しませんが、一定のコスト内で計画された成果物を出す義務が生じます。このタイプの組織は利益確保の責任がないために、仕事の遂行が予算消化的になってしまいがちであり、開発者たちを前向きにドライブする力が弱くなります。端的な例をあげるとお役所仕事的になってしまうということです。 コストセンター組織は利益確保の競争にさらされないために、開発計画においても自己都合的になりやすく、競合他社との競争意識も希薄になりやすい傾向を持っています。

実行

◎独立採算方式はプロフィットセンター、
予算方式はコストセンター。

【プロフィットセンターとコストセンター】（図４－４－１）



プロフィットドライブのすすめ

コストドライブな組織においては、投入資本と成果利益の結びつけ動機が弱く、コスト削減の指向性も弱くなり、極端を言えば投入予算は消化するためのもの、成果は出たとこ勝負的な状況に陥りやすくなります。

結果として高コスト・低品質の組織に陥ってしまい、ＱＣＤにおいて有効な成果を生み出すことが難しくなる可能性が高くなります。一般的なソフト開発組織においては市場との距離感が発生すると、コストドライブの考え方に陥りやすくなります。

一方プロフィットドライブな組織においては、日々低下する価格競争に勝ち残るべく、最小投資で最大利益を得るべく、徹底的なＱＣＤ達成のための改善アクションを継続的に実行することで、自らの組織力も筋肉質に強化されていく可能性が高くなります。 何もしなくても銀行口座に給料が振り込まれることを良しとする様なサラリーマン的社員が増加しているならば、組織の構成員に対してプロフィットドライブな考え方の浸透および実行が急務でしょう。

なおプロフィットドライブとは、市場における商品価値が商品価格を決定すると言う考え方であり、決して利益至上主義ということを意味するものではありません。どのような犠牲を払ってでも利益の確保を至上命令とするような組織において、組織員の安心・安全がおろそかにされてしまった場合、会社に利益は貯まったが社員は疲弊し組織は弱体化してしまうことになるでしょう。

【プロフィットドライブ】（図４－４－２）

プロフィットドライブとは

売上高 ↑	**売上高を上げ、同時に**
開発コスト ↓	**開発コストを下げることによって**
利益 ↑	**利益を拡大すること。**

実行

プロフィットドライブなプロジェクト運営の仕方

プロフィットドライブなプロジェクトとは、自律的なプロジェクトであるということであり、具体的に言えば利益をチームの力で稼ぎ出し、チームに必要なものは自分ですべて賄い、プロジェクトの運営をチーム自身の考え方で推し進めるということになります。

約束の利益を会社に還元し、余剰の利益の一部は将来のプロジェクトのための準備金として備蓄し、さらに一部を顧客や営業組織に還元するような組織を実現する必要があります。これは決して夢のプロジェクトではなく実現可能なプロジェクトなのです。

そのようなプロジェクトにおける行動指針は次のようになります。

【プロフィットドライブなプロジェクトの行動指針】

① 早期の仕様凍結に組織の全力を注ぐ。
② 顧客価値を見極めた開発費の見積りを行う。
③ 不条理なコスト削減には合理的論拠でもって対抗する。
④ リスク解消・失敗の回避・コスト削減・生産性向上・無駄の排除のための改善活動を継続的に行う。
⑤ 無料奉仕的な仕事はしない。無料サービス的な仕事には請求書を出す。

実行

◎改善活動の継続的な実行が、プロフィットドライブを実現可能にする。

4－5 納期・生産性目標値の達成☆優先順位ベース開発の実行

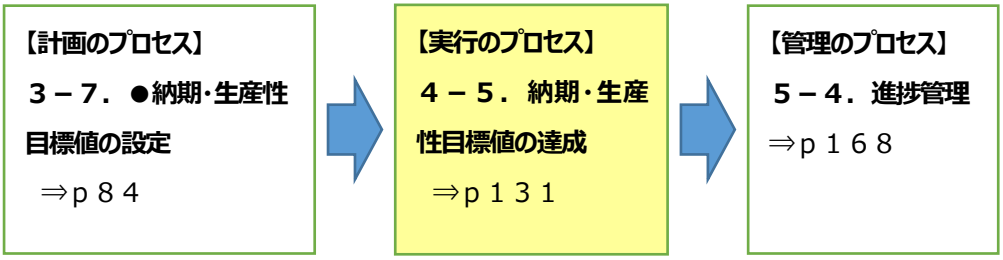
【タイムマネジメント】

一般的な物事の実行において、その着手する優先順位は非常に重要なことです。瑣末な問題に多くの時間を費やしていては、本当に重要な問題の解決に遅れをとってしまい、計画自体が中途で頓挫してしまうことがよくあるものです。

納期・生産性目標値の達成は、主に優先順位ベース開発の実行および後述の「４－６．改善活動の実行」によって実現されます。

納期・生産性目標値の達成

計画のプロセスの３－７．において納期・生産性目標値の設定について解説を行いましたが、本章においてはその目標値達成のために必要なプラクティスについての解説を行います。納期・生産性に関する解説の流れは下図のようになっています。



納期そのものは名目上の目標に過ぎず、納期は製品の品質やコストの目標が達成されて初めて意味を持つ指標となります。不具合が多発するような製品を納期にリリースすることなど納期を守ったなどとは言えません。

意味のある納期を支える指標としては、時間効率に関する生産性の指標が意味のある指標になります。

納期・生産性の目標値の設定については、第３章の計画のプロセス３－７．において説明した通りですが、これらの生産性目標値を達成するためには、本章で示しているすべてのプラクティスの実行が必要になります。

◎納期の達成とは、
見かけ上の納期を守ることではなく、
品質目標・コスト（利益）目標も含めて同時に達成すること。

実行

第３章の３－７．で示した生産性指標を再度示しておきます。

【生産性指標】

- 要件定義の生産性＝要件定義費用／要件定義時間
- 設計の生産性（設計効率）＝設計費用／設計時間
- 製造の生産性（製造効率）＝製造費用／製造時間
- 総合テストの生産性（テスト効率）＝総合テスト費用／総合テスト時間
- 納期達成率＝納期達成ＰＪ数／全ＰＪ数（ただし品質･コスト目標が達成されたＰＪのみ）
- 失敗による手戻り費用率＝手戻り費用／工程費用（開発各工程毎）
- 障害対応費用効率＝障害対応費用／全開発費

実行

◎すべての生産性指標は改善活動のテーマであり、

組織的な改善活動を実行しなければ上がることはない。

開発業務における実行の優先順位

一般的なものごとの実行の優先順位は次の通りです。

１．顧客価値の高い順。

２．次に、効果が大きく実行が容易なものの順。

３．タイムリミットが必須のものはその期限までに実行しなければならない。

優先順位に従った開発業務の実行手順は次のようになります。

要求仕様を顧客価値の重要度順に整理すること

顧客価値の重要度の順位は、顧客との密接なコミュニケーションを図ることで、概要要求仕様の検討段階で判断可能です。発注側･受注側において仕様の概要が分かった時点で、仕様開発の優先度の順位を決定･合意することが重要です。優先度の順位は、リリース時期の先･後を考慮に入れた上での顧客価値の重要度の順とすべきでしょう。

実行

◎無駄な開発の排除のために

① 顧客価値のない要件を排除すること。

② 顧客価値の低い要件の優先順位を下げること。

顧客価値の重要度順に仕様を凍結していくこと

仕様凍結にあたっては、仕様を全部一時に決めようとしないで、重要度順に凍結していくことが必要であり、更に顧客価値を満足させる要件定義の実行、すなわち開発目標の明確化が必要です。これらは要件定義にあたって、顧客と開発チームの密接なコミュニケーションをベースに、顧客の要求について顧客価値の優先順位付けを行い、優先度の高いものを顧客の必要とする時期に提供していく計画を立て、さらには顧客価値の低い開発行為の優先順位を下げることにより、顧客にとっても必要性の低い開発行為を排除すること、などを通して顧客の要求と開発行為を一致させることによって実現されます。

これらの要件定義における基本的な取り組みの実行は、要件およびその範囲（スコープ）の明確化をメリハリのあるものにし、その変更管理のコントロールを容易にします。さらに要件の仕様化にあたっては仕様の階層構造的記述（ＷＢＳ）が、仕様モレ防止・解釈違い防止・矛盾仕様の発見・正確性・分かりやすさに寄与するでしょう。

実行

◎有効な開発に要するエネルギーの最小化を図るために

① 顧客の要求と開発行為を一致させること。

② 顧客価値の優先度の順・必要とされる時期に成果物を提供していくこと。

集中仕様凍結会議の実施

仕様検討に時間がかかりそうな場合は、日時を決めた上で顧客と直接顔を付き合せ、こちらからも対案を出し、検討会や合宿で一気に仕様決定を行う必要があります。相手が決めてくれるのを待っているような受け身の姿勢では何も改善できません。

顧客価値の重要度順に凍結した仕様から開発着手を行うこと

顧客との間で合意された優先度の順位に従って、詳細仕様の決定、ＷＢＳの作成に基づいたスケジューリングを行い開発に着手します。 顧客との間で合意された開発の実行順位に関する情報は、逐次プロジェクトチーム内の全員に共有される必要があります。

【顧客価値の高い順の開発実行】（図４－５－１）

実行

ＱＣＤには優先順位はつけられない

プロジェクトの成功と失敗について、日経ＢＰ社による２００９年の「プロジェクト実態調査８００社」によればプロジェクトの成功率は３１．１%だったという分析結果が出ています。
（Q　品質は成功：51.9%、C　コストは成功：63.2%、D　納期は成功：54.6%）

この調査結果から見えることは次の通りです。
開発会社は表向き品質や納期優先だとは言っていますが、多くの失敗プロジェクトにおけるＱＣＤの結果を見る限りにおいて、その実態はコスト、すなわち利益第一優先で、見かけ上納期を守ったように装い、品質を犠牲にしている姿が見えてきます。
すなわち、現実のＱＣＤの優先順位はコスト（利益）＞納期＞品質の順になっています。
これで良いのでしょうか？これで良いわけがありません、顧客は怒ります。

会社を潰さないようにする視点で見れば、コスト優先になるのでしょう。コストはマネージャ以外には見えにくい指標ですから、一般の開発者に見える部分では納期優先となってしまいます。約束の時期を外すことはできないために、最後の評価・デバッグの時間が不十分だったとしても、タイムアウトということになり、バグ含みで出荷されるソフトウェアが後を絶ちません。コストを守るために、必要な開発時間をカットして納期の帳尻を合わせ、そのために品質を犠牲にしているという図式です。もしこの通りの優先順位で実行したら、顧客の満足や品質第一が優先だと言っていることは嘘になるでしょう。悪い品質のものを買わされて被害を受けるのは顧客自身です。

品質・コスト・納期に優先順位はなく、本来同列の価値を持つものです。皆必須条件なのです。どれかだけを優先し他を犠牲にすることはできません。人間において、脳を取りますか心臓を取りますか選んで下さいというのと同じです。

やるべきことは、その様な究極の選択をしなければならないような状況に陥らないようにすることです。失敗するプロジェクトの主な原因は、仕様凍結遅れ・開発プロセスの悪さ・リスク回避の失敗などにあります。つまり「何をつくるか」「どうつくるか」が不明確なため、大きな時間ロスを生んでいるということです。現実的に色々な悪条件が存在するプロジェクトにおいては、上記のことを十分に承知した上で、妥当な品質・妥当なコスト・妥当な納期を追及することがポイントになります。

実行

◎ＱＣＤに優先順位を設けてはいけない

実態は

コスト（利益）＞納期＞品質

実行

4－6 改善活動の実行【ＱＣＤマネジメント】

改善活動と言えば、大方の人たちは時間の余裕がある時に仕事のついでに行うＱＣサークル活動のようなものだと思っています。 時間がないから、予算がないから改善活動ができないという声をあちこちで聞きます。では、時間やお金があったら本当に改善活動ができるのでしょうか。そのような人たちに時間と金を与えても、結局形ばかりの活動になり、何らの実利も得られないでしょう。成長の意欲のある人は、他人から言われずとも知恵を出して、時間を生み出し、利益を生み出すための改善活動をすでに実行しています。成長意欲のない人には改善の意義も理解できず、改善の動機も湧かないのでしょう。

そういうわけで、時間がない、資金もない、要求仕様も二転三転して定まらないというような、愚痴ばかりを並べたてるばかりで、一向にそれらの改善活動も実行せず、仕事の内容は手抜きや丸投げが蔓延するばかりで、製品は市場でバグだらけで、損益も万年赤字続きが続いているのです。

改善活動の意義

このような荒廃した環境を作るのも復興させるのも人間、とくに組織をリードする人たちです。プロマネの役割と責任を自覚する必要があるでしょう。個人および組織に、実利の獲得と人間的成長を同時にもたらす運動の一丁目一番地は、コミュニケーションの活性化を基盤とした改善活動に他なりません。改善活動は開発業務の中核業務であり、暇な時についでにやるようなものではないのです。

改善活動がもたらす成果には次のようなものがあります。

【改善活動がもたらす実り】

1．ＱＣＤ数値の大幅な改善
2．チームメンバーの技術能力の向上
3．チームリーダーのプロマネ能力の向上
4．コミュニケーションの活性化による組織能力の向上
5．全員の自律（自立）能力およびモチベーションの向上
6．会社の成長・発展

改善活動の苦痛を嫌って現在の地獄的な状況を我慢し続けるのか、一時の新たな負担を背負う覚悟を決めて改善活動の実りを手に入れるのかを決めるのは、みなさんの選択次第なのです。

実行

◎改善活動は、開発業務の中核業務であり、暇な時についでにやるようなものではない。

仕事に対する認識

ルーチンワークをこなしているだけでは、品質の高い業務や製品を生み出すことはできません。ルーチンワークと合わせて、自他の過去の失敗に学ぶ改善活動の実行が、高品質な業務および製品を生み出すことはトヨタのカイゼン活動で実証済みであり、世界中に知られていることです。

改善活動を行っていない開発組織は、開発という創造的な仕事から、いつの間にかルーチンワークという単純労働集約型の仕事へと劣化していきます。良い仕事を継続的に実行していくためには、改善活動を中核にすえた仕事を日常的に実行する必要があります。

実行

◎**良い仕事＝改善活動＋ルーチンワーク**

開発業務＝創造的仕事＋ルーチンワーク

改善活動の基本コンセプト

改善活動の基本コンセプトは以下の通りです。

【改善活動の基本コンセプト】

① 個人戦[147]ではなく組織戦[148]（チームプレー）を行うこと。
② 一人だけが成長するのではなく、みな共に成長すること。
③ 能力に長けたものは、後進の者にその知恵（ノウハウ）を譲ること。
④ 後進の者もその成長に従って、さらに後に続く後進の者に、その知恵（ノウハウ）を譲ること。
⑤ 先に進んでいる者は、その持てるあらゆる価値あるもの、資産・資金・知恵（ノウハウ）の全てを後進のものに順に譲り渡し、永続的な繁栄の循環を実現させること。

実行

◎**改善活動の基本コンセプト**

共に成長し、持てる智・財を継承し、永続的な繁栄の循環を実現すること。

147 **個人戦** 組織に拠ることなく、個人単独で問題解決に当たること。

148 **組織戦** 複数の人間で組織を形成し、それぞれに役割分担を決め、協調・連携のもとに問題解決に当たること。

改善活動の事例

改善活動の具体的な例としては以下のものがあります。

【改善活動の事例】

１．コミュニケーションの活性化
２．開発プロセスの確立と励行
３．プロジェクトの数値目標設定
４．要求仕様書の精度向上
５．要求仕様の早期凍結化
６．要求仕様の変更管理
７．仕様変更影響度表の作成
８．見積り手法問題の改善
９．見積りリスクの排除
１０．開発リスクの排除
１１．設計書の精度向上
１２．評価仕様書の精度向上
１３．無駄の排除
１４．開発の効率化
１５．レビュー品質問題の改善
１６．プロジェクトの振り返り（ラップアップ）励行
１７．開発ノウハウの継承
１８．リーダーシップとチームプレー問題の改善
１９．モチベーションの喚起
２０．手抜き問題の撲滅
２１．組織風土文化の改善

いずれも身近にある問題ばかりです。これらの問題を全てとは言わないまでも１０%、いや３０%程度改善するだけで、どれ程の成果がプロジェクトのみならず会社にもたらされるのかは容易に想像できるでしょう。

◎改善活動の具体的なテーマは、
繰り返される失敗や、いつもリスク管理表に取り上げられる項目の中にある。

改善活動計画書

改善活動計画書の基本的な記述内容を下記に示します。

この計画書はいわゆるＤＭＡＩＣ[149]手法に準じたもので、ＤＭＡＩＣの意味は次の通りです。

Ｄ（Define：問題点記述）

Ｍ（Measure：現状の指標値データ）

Ａ（Analyze：現状のデータ分析）

Ｉ（Improve：解決策）

Ｃ（Control：成果の維持・定着方法）

【改善活動計画書フォーム】

１．プロジェクトテーマ名

２．プロジェクト体制　：リーダー名記述、メンバー名記述

３．プロジェクト期間　：開始日、終了予定日

４．問題点記述（Ｄｅｆｉｎｅ）：＊解決すべき課題を現在の状況と共に複数記述する。

５．顧客に提供される価値（顧客のＣＴＱ Critical To Quality）

＊この改善によって顧客に提供される価値（ＱＣＤ等）を記述する。

６．プロジェクトの成果目標

・目標の記述　＊何をどのようにするかを記述する。

・成果目標値の記述　＊改善活動によるＱＣＤ等の成果期待値・金額等を記述する。

７．現状の指標（値）データ（Ｍｅａｓｕｒｅ）

・収集した現状の問題等を整理・分類し、データ化する。改善目標について現状の数値を示す。

８．現状のデータ分析（Ａｎａｌｙｚｅ）

・フィッシュボーンによる分析；問題を構成している要因分析を行い、主要要因をあぶり出す。

・データに基づく分析；問題を構成している複数の要因を数値分析し、改善ターゲットを特定する。

９．解決策（Ｉｍｐｒｏｖｅ）

・複数の解決策を立案し、実行の優先順位をつける。

・実行の優先順位は、ＱＣＤ改善効果（費用対効果）＋実行難易度＋緊急度により決定する。

１０．成果の達成見込み（＊改善実行後には実績値に書き換える）

・成果物、改善金額、投資コスト、成果金額（＝改善金額－投資コスト）等。

１１．成果の維持・定着方法（Ｃｏｎｔｒｏｌ）

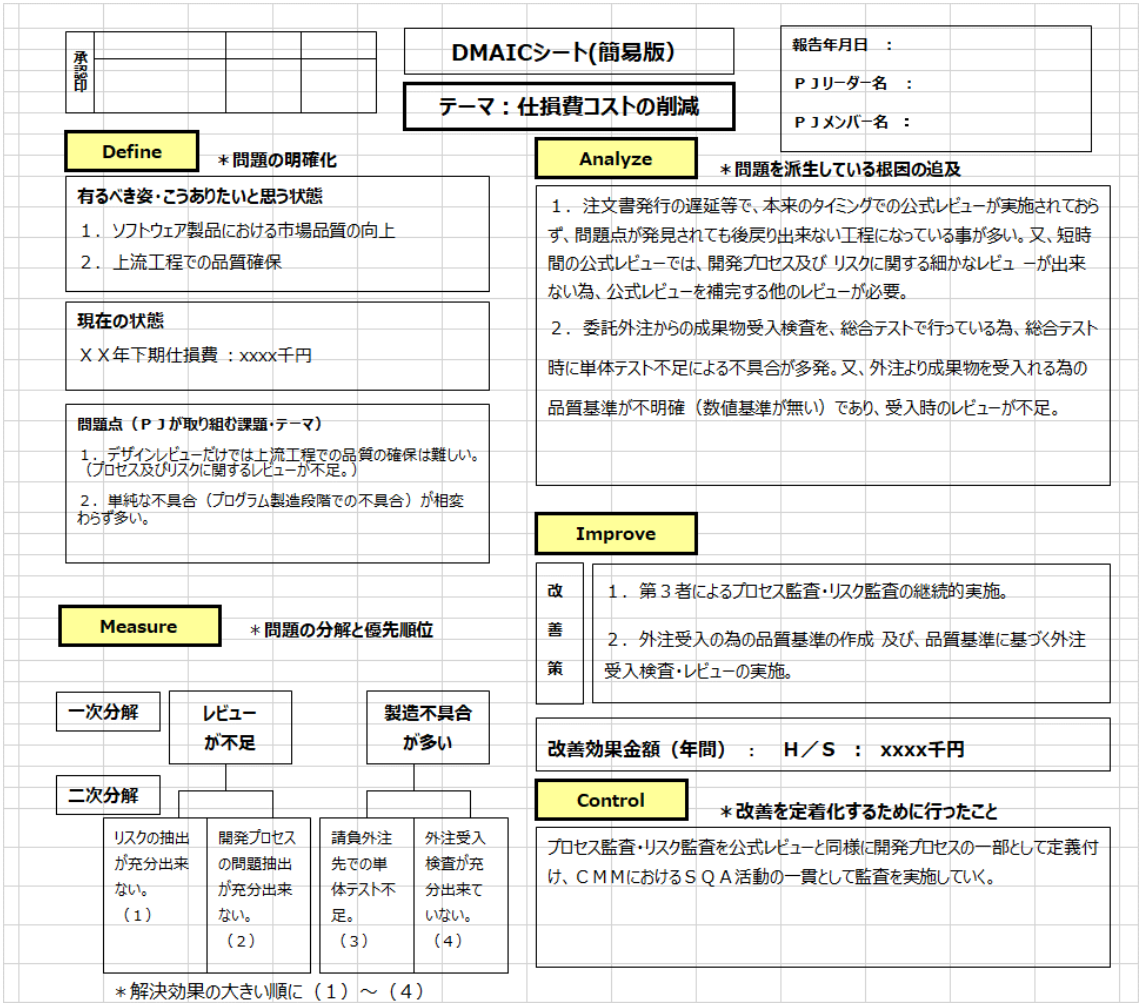
・実行された解決策が将来に渡って組織に定着する方法・手段を記述する。

---

[149] **ＤＭＡＩＣ** Define（定義）、Measure（測定）、Analyze（分析）、Improve（解決策）、Control（管理）のステップからなる経営変革手法であり、VOC（Voice of Customer、顧客の声）を基にして事業活動を分析し、データドリブンでプロセスの改善を進める。

下記は改善計画書簡易版のサンプルです。

【ＤＭＡＩＣシート簡易版】（図４－６－１）



ノウハウの継承の視点から見た改善活動の意義については、「７－２．改善活動のすすめ」を参照。

実行

改善活動を阻害するもの

改善活動を阻害する要因として、心理的なものと環境的なものの二つがあります。

心理的な要因

心理的な要因としては、たとえば改善活動によって将来大きな利益が得られると言われても、その失敗リスクを考えると現状のままの方がまだ安全だという心理が働き、たとえ、品質が悪い・コストが高い・生産性が悪いという状況であったとしても、何とかやっていくことが出来ているという、低レベルではあるが、それなりに安定している現状を失いたくないという心理が働きます。要するに、悪いなりに安定している現状を失いたくないために、将来利益を生み出すかも知れない改善活動を避けようとしているものと考えられます。また現状のＱＣＤのレベルはたとえ余り良くないとは思っていても、顧客からの苦情も一過性で済んでおり、何年もこのような状況に慣れてしまっているので、改善などの手間ひまがかかる活動を始める動機がわかない、という心理的な傾向の影響もあるのでしょう。

【改善活動の阻害要因】　１．現状を替えたくない心理　２．費用・時間の不足

環境的な要因

社内の改善活動にとっての最初の障害は費用および時間の不足にあります。出たとこ勝負的な開発を行っているような組織は、慢性的な赤字と時間不足に悩まされています。仕事に投入される時間は、妥当な品質を確保するために必要な時間の６０％以下なのかも知れません。現実的な問題として、このようなチームに自分たちの時間の５％を割いて改善活動を開始するように説いたところで、所詮そんな時間はどこにもないから、やはり改善活動はできないということになります。

このようなチームを復活させるためには、そのチームの予算外で動ける人材の投入が必要になるということを最初に理解しておく必要があります。チーム内に改善活動をリードする適任者がいたならば、その人材の予算を別途確保し、そのチームには背負わせないような手当てが必要となります。

経験から言えることは、改善活動によってある程度の実利が得られる見通しさえついていれば、このような初期投資の金額の数倍もの改善益はすぐにでも得られますが、ないない尽くしの中で改善活動ができないのは当り前のことであるとも言えます。

実行

◎それでも改善活動には、初期投資のヒト・カネ・時間が必要になる。

ノウハウ継承の視点から見た改善活動の意義については、「７－２．改善活動のすすめ」を参照のこと。

【Human Activity】

4－h 1 獲得時間の最大化と失う時間の最小化【タイムマネジメント】

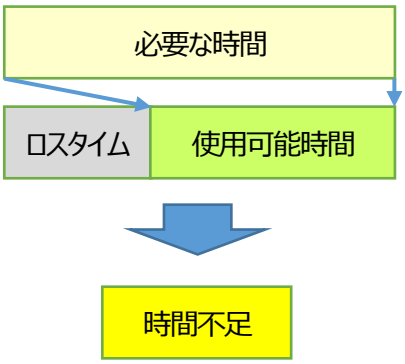
私たちが失っている時間

最初に私たちが失っている時間にはどのようなものがあるのか、プロジェクト活動の全体を通して見てみる必要があります。 私たちは下記に示すようなことが原因で多くの時間を失っています。昨日も・今日も・明日もこのようなことがあちこちで行われているのです。下記に示した事例に全く該当するものがない人など存在しないでしょう。

【ロスタイム】

① 要件定義遅れによる時間ロス
② 雑な要求仕様書による長時間の仕様調査・検討
③ 無理な納期短縮による工期不足
④ 見積りの失敗による工期不足
⑤ 他人まかせの仕事による進捗遅延
⑥ 繰り返される類似不具合
⑦ 段取りミスによる手戻り
⑧ 現物確認を怠ったためのミスによる手戻り
⑨ 口頭依存のコミュニケーションによる仕様の誤解
⑩ 情報共有不足によるミス・仕事の重複
⑪ 優先順位判断ミスによる時間ロス
⑫ 使える設計書がないことによる長時間のソースコード解析
⑬ 使える設計書がないことによる的外れな評価業務
⑭ 結論の出ない長時間・頻繁な会議
⑮ 時期外れの形式的なレビュー
⑯ 報告者の意思がない意味不明な報告書
⑰ 多数の無関係なｃｃ：メール
⑱ 仕事に関係のない長電話
⑲ 会議・会談への遅刻

【ロスタイムは取り戻せない】
（図４－ｈ１－１）



上記を見ただけでも、どれだけ多くの時間が失われているのか簡単にイメージできるでしょう。全開発時間の３～４割は軽く超えていることでしょう。

この問題はプロマネを中心としたチーム全員で取り組めば、大幅な改善が可能です。

実行

プロジェクトにおける時間の考え方

開発担当者たちの最大の悩みは時間不足です。個々の問題の表面的な原因は、要求仕様書や設計書のあいまいさ、コミュニケーションの不良や技術レベルの低さなどがあげられていますが、さらに突き詰めると「調査時間が足りませんでした」「検討時間が足りませんでした」とか「実行時間そのものが足りませんでした」というような答えが非常に多いことに気づかされます。

プロジェクトに必要なリソースとして、人・モノ・カネ・情報などが重要なものとして挙げられますが、最も重要なものは「時間」なのです。他のリソースは何らかの手段を講じればなんとか入手することは可能ですが、失われた時間を取り戻す方法などありません。実際にプロジェクトにおいて必要な時間が不足した場合に、納期を守るために行われることは、手抜きという不正行為なのです。

実行

◎プロジェクトにおける最大の悩みは、時間不足。

Ｐ．ドラッカー[150]は仕事と時間の関係について次のように語っています。

「私の観察によれば、成果をあげる者は仕事からスタートしない。時間からスタートする。計画からもスタートしない。何に時間がとられているかを明らかにすることからスタートする。（中略）成果をあげる者は時間こそが、真に普遍的な制約条件であることを知っている。」

まさにその通りで、プロジェクトはその実力に応じて開発に必要な時間が決まり、この必要な時間を獲得するための行動が見積り交渉であり、その獲得された時間を必要なタスクに割り当てる細分化作業がＷＢＳの本質なのです。もし見積りで獲得した時間がプロジェクトで必要とされる時間より短ければ、時間不足という事態に陥ることになり、プロジェクトはその出発点において失敗を運命づけられることになります。

実行

◎プロジェクトにおける普遍的な究極の制約条件は、時間である。

【時間不足が招くもの】

◎手抜きによる品質の悪化、さもなくば　**◎納期の遅延**

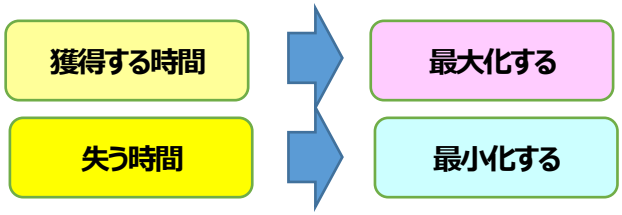
150 **Ｐ．ドラッカー**（Peter Ferdinand Drucker）　経営学者。『マネジメント』はその代表的な著書。

実行

獲得時間の最大化と失う時間の最小化

プロジェクトにおいての時間には二種類があります。一つはプロジェクトが獲得した時間、もう一つはプロジェクトで失う時間です。プロジェクトは、見積り交渉において開発期間という時間を獲得し、プロジェクトの進行中にさまざまな問題によって時間を失っています。プロジェクトを成功させるためには、そのプロジェクトに必要な時間の確保が必要であり、そのためには見積り交渉においての獲得時間の最大化と、開発行為の不手際によって失われる時間の最小化という二面性のアプローチが必要です。

【プロジェクトにおける二通りの時間】（図４－ｈ１－２）



獲得時間の最大化

獲得する時間の最大化は、主に見積り時における必要な開発期間の獲得によって達成されます。

プロマネを初めとしたマネージャは、見積り交渉において不条理な圧力に負けず、無理な短納期の要求に対しては知恵を出し条件闘争に持ち込むような、タフな交渉力および論理的な説得力が必要となります。

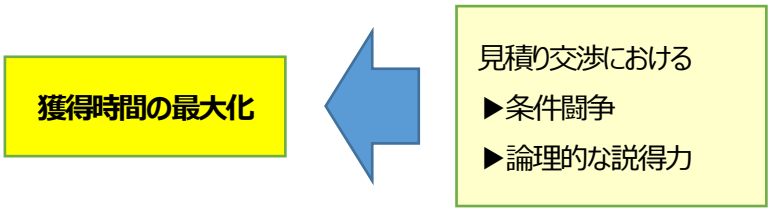
会社が戦略的な物件として不足分を補う約束をしていない限りは、コスト競争に勝つためという理由で、本当の生産性を上げる努力もしないまま、低価格・短納期の見積りを出すべきではありません。

他社との競争に勝てるような見積りが提示できるようになるには、失敗や無駄の排除および効率的な開発の実践による、生産性の向上などの継続的な改善活動を行う必要があります。改善活動の実践は見積り交渉時における論理的な説得力およびタフな交渉力の源泉ともなります。

実行

◎見積り交渉にて妥当な開発期間・費用を獲得するためには、
普段の改善活動の実践による、論理的な顧客説得力の強化が必要。

【獲得時間の最大化】（図４－ｈ１－３）



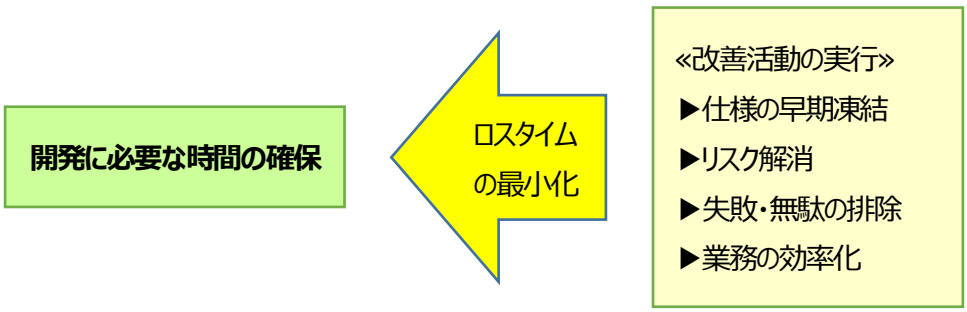
失う時間の最小化

失う時間の最小化に貢献する具体的な行動としては以下のものがあります。

① 要求仕様の早期凍結を実現すること。
② 開発初期の時点で大きなリスクを解消しておくこと。
③ 同じ失敗を繰り返さないこと。
④ 無駄を排除すること。
⑤ 仕事自体の効率化を行うこと。

いつまでも決まらない二転三転する要求仕様が、失われる時間の代表的なものです。顧客から要求仕様が出て来るのを待つのではなく、提案型のアプローチのもとに、集中的な要求仕様検討会を実行することで、早期の仕様凍結は可能になります。少なくとも基本設計着手前には、要求仕様の骨子の決定が必要であり、詳細設計着手前には、詳細レベルの要求仕様の決定が必要です。

【ロスタイムの最小化】（図４－ｈ１－４）



プロジェクトにおける主な時間効率化方法

プロジェクトにおいては、さまざまな効率化施策が必要となりますが、いずれにしても時間効率性を高めることに効果のある施策は、同時に品質やコストの改善にも有効に働きます。下記は実際のプロジェクトにおいて必要とされる重要な効率化方法です。

目標の明確化および集約化

1．顧客との密接なコミュニケーションを通して、要求仕様の明確化および開発項目の優先度付けを行なうこと。顧客価値のないあるいは低い開発を防止し、顧客価値のある開発に集中すること。

2．チーム内の短時間日次会議の繰り返しを通して、目標の明確化および全員による共有化、開発情報・問題情報の共有化、行動の方向性の統一化を図ること。

【目標の明確化および集約化】

- 要求仕様の明確化
- 開発項目の優先度付け
- ＱＣＤ目標の明確化・共有化
- 問題情報の共有化
- チーム行動の方向性の統一化

変化即応的な組織運営

1．科学的データ・思考に基づくプロジェクト運営を行なうこと。情緒的な人間関係に片寄らず、非人間的な運営に傾かず、合理的かつ妥当性のある組織運営を行なうこと。

2．多重請負構造を避け、役割りの丸投げを禁止し、参加組織数の削減に心がけることで、役割分担の明確化を図り、情報伝達性・情報正確性・組織統合性を向上させること。

3．プロジェクトを構成している複数の組織間の情報連携、活動連携を密接にすること。

【変化即応的組織】

- 合理的・妥当性のある組織行動
- 多重請負構造の回避
- 役割丸投げの禁止
- 役割分担の明確化

リスク管理・プロセス管理の実行

1．リスク管理により、予測される全ての失敗を予防すること。

2．あるべきプロセスの確実な実行により、手抜き・実行忘れなどの不要な失敗を防止すること。

【リスク管理・プロセス管理】

- ●リスクの解消
- ●手抜き・実行忘れの防止

ドキュメントベース開発の実行

1．基幹ドキュメントの質の向上およびメンテナンスを励行することで、ノウハウの蓄積および継承の最大化を図ると同時に、ドキュメントベースによる強力な情報共有化を通した、正確かつ無駄のない開発を実行すること。

2．使用技術用語の統一化による、顧客・開発間の意思疎通レベルの向上を図ること。

【ドキュメントベース開発】

- ● ドキュメントの質の向上
- ● ドキュメントによるノウハウの蓄積と継承の拡大
- ● 情報共有化による無駄のない開発
- ● 技術用語統一化によるコミュニケーションの質の向上

柔軟なシステム構造の考慮

スピードと柔軟性を実現する高メンテナンス性[151]を持ったシステム構造体、すなわち高メンテ領域と低メンテ領域の分離構造の実現等によって、仕様変更による他の領域への影響度を最小限に抑える。

【柔軟なシステム構造】

- ● 高メンテナンスソフトウェア構造による、仕様変更・追加に対する開発スピード性と柔軟性の向上

151 **メンテナンス性** 仕様変更の容易性が高いこと。いわゆるスパゲッティプログラムは、低メンテナンス性の代表例。

実行

ノウハウ循環サイクルの実現

1．振り返り会議を通して失敗の教訓に学び、同様な失敗を組織的に防ぐと同時に、ノウハウの蓄積および継承を行なうこと。

2．改善活動の常態化による、継続的な効率化の進展および部下の育成を図ること。

3．知識インフラとしての顧客情報・技術情報の収集、学習、教育を継続的に実行すること。

4．指示待ちではなく、自分の頭で考え・判断・行動する自律的人材を育成すること。

【ノウハウの循環サイクル】

- 失敗の教訓に学び、類似の失敗を予防
- ノウハウの蓄積と継承による、好循環開発サイクルの実現
- 改善活動による、効率化および自律的な人材の育成

実行

◎知恵と工夫と努力により、時間は手に入れることができる。

参照：第７章　７－２．改善活動のコンセプト●コンセプト＃１．時間の獲得

実行

【実行プロセスのまとめ】

実行のプロセスの最大の目的は、ＱＣＤ目標の達成でした。本文の順に沿って振り返ってみます。

【Job Activity】

実行

【リスクの事前解消】

◎リスクの三つの押さえ所、開発の入り口・中間地点・出口。

◎コンティンジェンシープランとは、予期しない失敗に備えるための計画のこと。

リスクの事前解消については、「リスクはヒトにあり」という視点で、本章で取り上げた多くの事例およびリスク管理表を参考に取り組んでいただきたいと思います。なお不測の事態に対応するコンティンジェンシープランの計画も最後の防波堤として用意しておく必要があります。

実行

【データドリブン開発】

◎基本的なデータは、ＱＣＤのデータ。

◎過去のデータ、目標とするデータ、結果のデータの三つでドライブする。

合理的な開発の基本は、ＱＣＤの数値に基づいた開発の実行にあります。モノの品質だけではなくヒトの品質にも注目し、開発開始時には必ずＱＣＤの目標値を設定し、開発終了時においてはＱＣＤ実績値との比較を行うことで、プロジェクトの弱点・長所の把握を行い、改善行動にドライブをかける必要があります。

実行

【プロフィットドリブン開発】

◎利益志向型のプロフィットドライブ組織、原価志向型のコストドライブ組織。

◎改善活動の継続的な実行が、プロフィットドライブを実現可能にする。

自律した開発組織になるためには、採算責任を持った組織であり続ける必要があります。会社依存型のコストセンター的な組織では改善意欲も湧かず、いつの間にか惰性で動くような低生産性な組織になってしまいます。

実行

【ドキュメントベース開発】

◎整備状況・精度レベルの確認が必要な三種類のドキュメント

①設計ドキュメント ②開発管理ドキュメント ③ビジネスドキュメント

◎妥当なレベルのドキュメントなしでは、ＱもＣもＤも達成することは不可能。

◎ドキュメントの誤記・抜け・追加は、気づいた時点でリアルタイムな修正が必要。

◎重要な三つの開発管理表：プロセス管理表、リスク管理表、課題管理表

精度の低い、メンテもされてないような設計書で、自分の記憶や想像に頼ってできる程、この仕事は単純な仕事ではありません。時間がないからメンテできないという人たちは、アイマスクをして車を運転しているようなものです。時間がないなら、時間を生み出す工夫が必要です。ソフトウェア開発の基本中の基本は、正確なドキュメントに基づいた開発を実行するということです。

実行

【優先順位志向開発】

◎無駄な開発の排除のために

① 顧客価値のない要件を排除すること。

② 顧客価値の低い要件の優先順位を下げること。

◎有効な開発に要するエネルギーの最小化を図るために

① 顧客の要求と開発行為を一致させること。

② 顧客価値の優先度の順・必要とされる時期に成果物を提供していくこと。

◎ＱＣＤに優先順位を設けてはいけない

最大の時間効率性を発揮するために、優先順位の判断と行動は必ず必要です。判断に迷った場合は知恵のある人に相談すべきでしょう。但しＱＣＤに優先順位は付けられないということを強く認識しておく必要があります。

実行

【改善活動の実行】

◎改善活動は、開発業務の中核業務であり、暇な時についでにやるようなものではない。

◎良い仕事＝改善活動＋ルーチンワーク

◎改善活動の基本コンセプト：共に成長し、持てる智・財を継承し、永続的な繁栄の循環を実現すること。

ＱＣＤ目標の達成および開発者のスキルアップを確実に実現するためには、開発行為そのものが改善行動である必要があります。いつもと同じようなルーチンワーク的な作業を続けることは、開発という名前にはふさわしくないでしょう。

【Human Activity】

実行

【獲得時間の最大化と失う時間の最小化】

◎プロジェクトにおける最大の悩みは、時間不足。

◎プロジェクトにおける普遍的な究極の制約条件は、時間である。

◎見積り交渉にて妥当な開発期間・費用を獲得するためには、普段の改善活動の実践による、論理的な顧客説得力の強化が必要。

◎知恵と工夫と努力により、時間は手に入れることができる。

プロジェクトの最大の問題は時間不足にあります。時間は究極の制約事項であり、開発者の行動もＱＣＤの到達レベルも限定されてしまいます。改善活動として、このテーマに取り組むことはプロジェクトおよび開発メンバーに多くの恩恵をもたらすでしょう。

以上で取り上げた実行のプロセスにおけるテーマについては、開発の規模に関係なく、すべてプロマネ主導で実行される必要があります。

第５章

管理のプロセス

概要

第５章　管理のプロセス

ＱＣＤ目標達成の状況推移を監視・管理し、目標との乖離を埋める活動についての解説です。

- 要求仕様の変更を管理することで、ＱＣＤの悪化を避ける（５－１．）
- 品質目標値の達成状況を監視し、目標との乖離を埋める活動を行う（５－２．）
- コスト・利益目標値の達成状況を監視し、目標との乖離を埋める活動を行う（５－３．）
- 開発進捗状況を監視し、目標との乖離を埋める活動を行う（５－４．）
- ＱＣＤを支える基本は、コミュニケーションの活性化にある（５－ｈ１．）
- チームプレーの実行による効果的な組織戦は、ＱＣＤ目標の達成の基本（５－ｈ２．）

事前準備を行い、計画を立て、その目標に従って実行を始めたとしても、計画通りに事が進むことはまずあり得ません。プロジェクトに対する要求の変化や開発能力の見誤りなどの、予期しなかったリスクの出現などが必ず有るものです。

本章の管理のプロセスは、設定した目標と実際の達成状況を実行のプロセスから監視し、目標との乖離を埋めるためのアクションを行うための活動です。プロジェクトにおける主要な管理項目である、要求仕様の変更管理、品質管理、コスト管理、進捗管理などについて順に解説を行い、最後にこれらのプロジェクト活動を下支えするコミュニケーションの活性化およびチームプレーについて触れることにします。

管理

【Job Activity】

5－1 要求仕様の変更管理【スコープマネジメント】

一旦、顧客と開発側において開発すべき仕様および開発条件（開発費・開発期間・品質条件）などが合意されたとしても、その後に顧客側ないしは開発側の都合によって変更や追加の要求が出された場合、最初に約束した内容を守ることが難しくなってしまいます。特に仕様の度重なる変更や追加は開発費・開発期間・品質に重大な悪影響を与えてしまいます。

ベースラインの意味

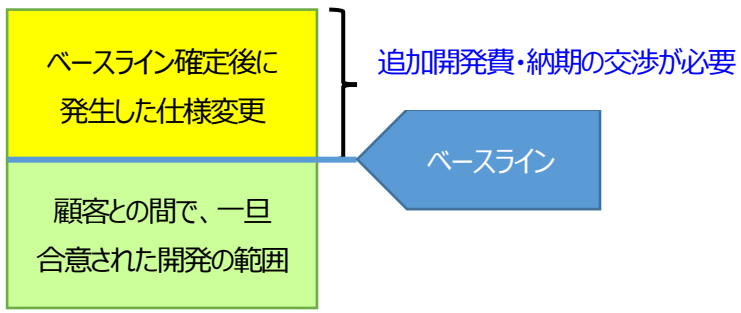
見積り回答によって顧客と約束した、開発費・開発スケジュール・品質条件および開発仕様の内容・範囲のことをベースライン[152]と呼び、このベースライン確定後の開発条件の変更に対する顧客との交渉を伴う処置を変更管理と呼んでいます。

開発側においては、大きな変更によるＱＣＤ等に対する重大な悪影響を避けるために、最初に約束したもの（ベースライン）とその後に発生したもの（変更・追加）を分けて処理する必要があります。追加・変更分は基本的に追加開発費・追加開発期間で処理される必要があります。

管理

◎ベースライン（基線）とは、
プロジェクトマネジメントにおいては、顧客と開発側で合意した成果物の範囲および基準の事を指す。両者間で約束した仕様とそのＱＣＤに関する基準は重要なベースラインとなる。

【ベースラインと仕様の変更管理】（図５－１－１）



152 **ベースライン** 見積り回答によって顧客と約束した開発費・開発スケジュール・品質条件および開発仕様の内容・範囲のこと。

変更管理が行われにくい理由

変更管理が行われにくい理由として最初に考えられることは、開発のスコープ・予算・期間を決定してしまうベースラインの確定そのものを顧客側が嫌うということがあります。

一般的に顧客側は限られた予算と期間の中で、できるだけ多くの仕様を盛り込みたいと思っていますが、ベースラインの確定はそれに対しての縛りとなってしまいます。

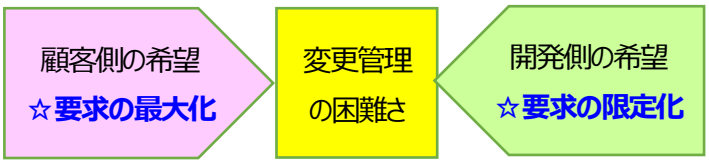
顧客側の考え方は、できるだけ良いものをできるだけ安く手に入れたいという所から出ているのでしょうが、そのように都合の良いことがいつでもあるわけがなく、相応の価値あるものに相応の対価を払うという健全なビジネスの基本からは逸脱する考え方と言わざるを得ません。

余談になりますが、優れたビジネスを行っているある顧客は「品質にはそれ相応の対価を支払う」と言っており、その顧客の値引き要求に対する回答に対しても削減の根拠の提示を求められました。根拠のない値引き提案など受け付けられないと言うことなのです。

管理

◎健全なプロジェクトには、ベースラインの確定が必要である。

【変更管理の困難さ】（図５－１－２）



要求仕様の変更管理不在がもたらす災い

要求仕様のベースラインの確定も変更管理も実行されないならば、顧客側は白紙委任状[153]を手にしたのも同然で、早期の仕様まとめの動機も失われ、顧客のあちこちの部署からさみだれ的に集まってくる要求をさみだれ的に開発会社に投げ続けることになります。これが二転三転する要求仕様の本当の原因なのです。要求仕様を考える能力がないのではなく、要求仕様を決められた時期までにまとめる意思がないだけのことなのかも知れません。

このように際限なく続く仕様の変更や追加は、ついにはプロジェクトの処理能力を超えてしまい、多くの手戻り作業や追加開発により、膨大な時間を失い、品質劣化、コスト高、生産性低下を招いてしまいます。

この結果は顧客側にもブーメランのように戻ってくることになり、顧客は納期遅延や障害多発に見舞われ、最悪の場合は顧客のビジネス自体が止まってしまうこともあります。

ベースラインの確定も変更管理の実行もなされないプロジェクトは、顧客側・開発側双方において致命的な結果を招くという認識が必要です。

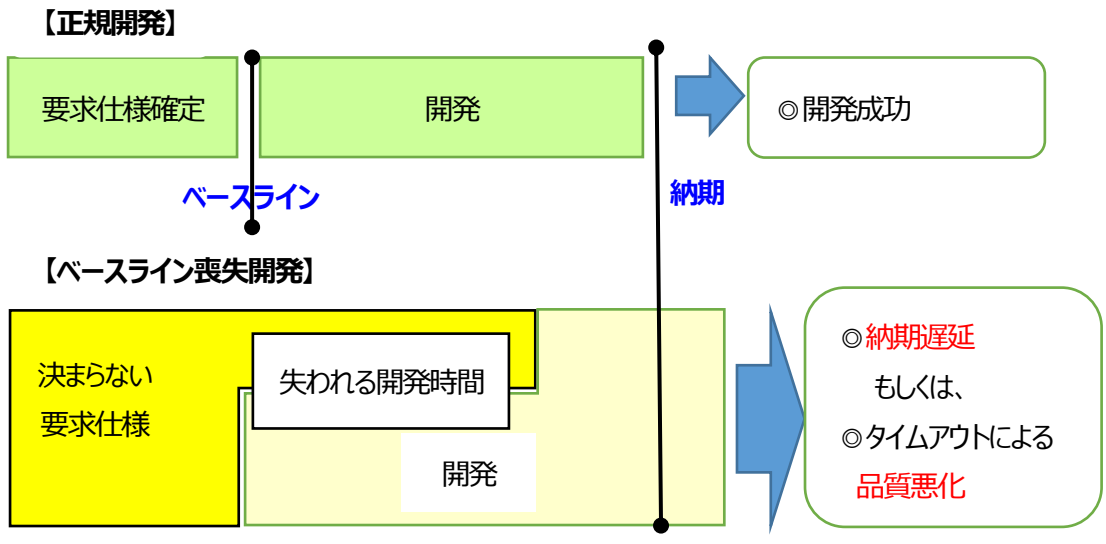
管理

◎ベースラインの確定・変更管理の不在がもたらすリスク

☆プロジェクトの失敗：ＱＣＤの大幅な未達

☆顧客ビジネスの停止：市場障害の多発

【正規開発とベースライン喪失開発】（図５－１－３）



153 **白紙委任状** ものごとの決定権を相手に委ねてしまうことの例え。

管理

仕様変更管理表がもたらす効果

安易な仕様変更にしっかりとした歯止めをかけるためには、仕様変更管理表の運用が不可欠です。

この管理表の主な目的は、仕様変更の影響を管理し、顧客側の無定見な仕様変更・追加を抑制することにありますが、特に顧客と開発側で同意した仕様凍結日以降に発生する仕様変更を厳重に管理することが重要であり、同意された仕様凍結日以降に発生した変更・追加仕様は基本的には別途見積り（追加コスト・追加開発期間）であることを顧客側が明確に認識するために使用されます。 仕様変更管理表の運用は下記のような効果をもたらします。

【仕様変更管理表の効果】

1．放縦・無定見な仕様変更の防止
2．顧客・開発側両者に対する仕様凍結期日の厳守化（開発仕様のベースラインの確定）
3．情緒的な要件定義工程のプロセス化
4．仕様変更の質、量および費用の厳格な管理

管理

◎ベースライン確定後の仕様変更・追加分は別途見積りとする交渉が必要。

仕様変更管理表には、管理番号、発生日、対応期限、対応日、対応状況、業務名、変更内容、変更区分、修正担当、影響度範囲、難易度、リリースバージョン、見込み対応工数、実績対応工数などを記録し、仕様の変更状況を明確にしておく必要があります。仕様変更管理表のフォーマットは次のようなものになります。

【仕様変更管理表】（表５－１－１）

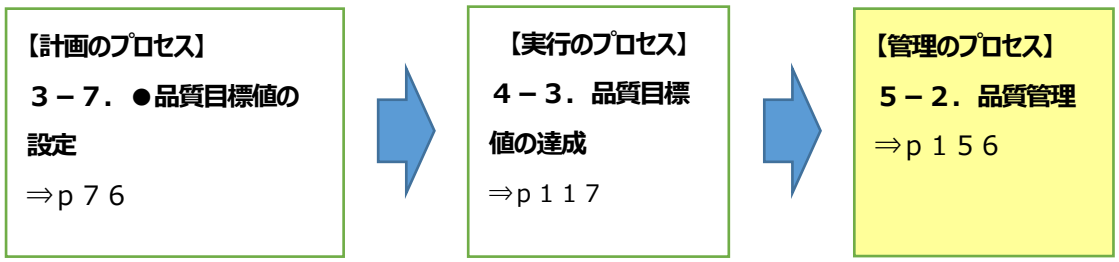
【仕様変更管理表】

No.	管理番号	発生日	対応期限（ベースライン）	対応日	対応状況	業務名	変更内容	変更区分	修正担当	影響度範囲	難易度	リリースバージョン	対応工数（見込）	対応工数（実績）
1	PJ-001										A			
2											B			
3											C			
4														
5														
6														
7														
8														
9														
10														
合計														

拡大図表は、付録図表１１．を参照のこと。

5－2 品質管理【品質マネジメント】

これまでの品質に関する解説の流れを図示すると下記のようになります。



第３章の３－７．においては品質目標値の設定方法について、第４章の４－３．においては品質目標値の達成活動について解説をしました。本章においてはこれらの設定された品質目標値に対するモニタリング[154]およびコントロール[155]について解説をします。

品質管理

品質管理の対象になる品質指標はすでに示した通りですが、これらの目標値を達成するためには当然のことにその裏付けとなる改善活動が必要になってきます。改善活動を行うためには、その指標の数値について、現在の数値の把握、改善目標数値の設定および改善結果数値の把握およびコントロールが必要になってきます。 例えば製品品質の代表的な指標である単体テスト・結合テスト・総合テストにおける不具合密度の例を次に示します。

【各テストの不具合密度の現在値・目標値・改善実績値】 不具合密度：（件数／kstep）

	現在値	目標値	実績値
単体テスト	5	３．５	3
結合テスト	3	2	２．５
総合テスト	2	１．４	１．６

各テストにおける不具合密度について現在値を把握した上で、何らかの改善施策により３０％の改善目標を立てた場合には、上記のような数値目標の設定となります。各テストの実行時に、実績数値を把握することで達成率を把握し、改善施策の不足部や欠点を特定し、次なる改善活動の目標とします。

改善施策としては、設計書の精度向上・評価仕様書の精度向上・仕様変更影響度表の作成などが有効でしょう。

---

154 **モニタリング** ＱＣＤ等の目標値の達成推移状態を記録・監視すること。

155 **コントロール** ＱＣＤ等の目標値のモニタリングにおいて目標未達が予測された場合、原因の特定および是正措置を行う。

品質管理に使用される管理表の例

品質状況管理表

品質状況管理表は、単体テスト・結合テスト・総合テストにおける不具合発生件数・原因・バグ密度[156]・深刻度レベルなどの把握に使用されます。拡大図は「付録図表１２．品質状況管理表」を参照のこと。

【品質状況管理表】（表５－２－１）

【品質状況管理表】

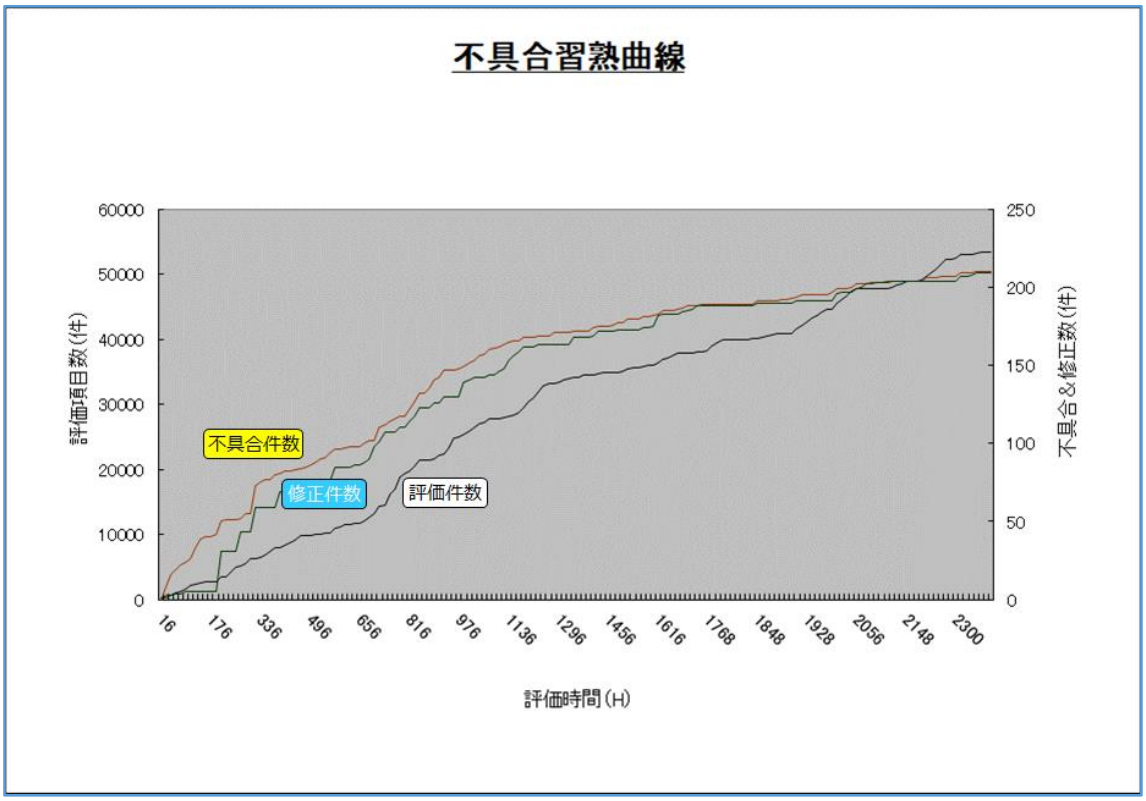
				不具合原因内訳						不具合深刻度レベル			
工程	業務名	プログラムステップ数	テスト項目数	仕様バグ	設計バグ	製造バグ	管理ミス	不具合件数合計	バグ密度（件数/kstep）	A（重度）	B（中度）	C（軽度）	テスト工数
単体テスト	A												
	B												
	C												
	D												
単体テスト合計													
結合テスト	A												
	B												
	C												
	D												
結合テスト合計													
総合テスト	A												
	B												
	C												
	D												
総合テスト合計													

156 **バグ密度** 作成されたソフトウェアの１ｋｓｔｅｐ当たりに発生したバグの数で表される。

不具合習熟曲線表

不具合習熟曲線表[157]は、信頼度成長曲線またはゴンペルツ曲線などと呼ばれるもので、横軸にテスト時間、縦軸に累積バグ発見数・修正件数・評価件数をとったグラフで表されます。S 字の成長習熟曲線を描くことが多く、決定論的モデルとして、現在の状況から今後の予想を立て、テスト進捗管理、バグ収束率の予測、残バグ数の予測などに用いられます。

【不具合習熟曲線表】（図５－２－１）



157 **不具合習熟曲線表** 信頼度成長曲線またはゴンペルツ曲線などと呼ばれるもので、横軸にテスト時間、縦軸に累積バグ発見数をとったグラフで表される。S 字の成長習熟曲線を描くことが多く、決定論的モデルとして、現在の状況から今後の予想を立て、テスト進捗管理、バグ収束率の予測、残バグ数の予測などに用いられる。

レビューの品質と実行

品質のモニタリングおよびコントロールにおいて、品質管理表と同時に大切なのが、主要な成果物に対するレビュー[158]です。レビューの目的はそれぞれの成果物の品質評価による誤りの排除にあります。

レビューのポイントおよび設計レビューにおける主なチェック内容を示します。

◆【全レビュー共通の効果的レビューのポイント】

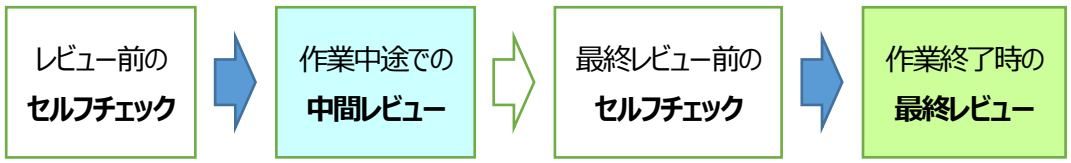
- □ あらかじめレビュー計画を立て、時間を確保しておき、レビューのポイントを考えておく。
- □ レビュー対象物が一目で分かるような一覧表を作成しモレをなくす。
- □ 共通レビュー項目の雛形を作成しておく。また既存のレビュー資料で流用できるものは流用する。
- □ 各担当者のスキルに合わせたレビューのチェックポイントおよびレビュー内容を考える。
- □ セルフチェックの時間を設け、レビューしきれない詳細部分については、事前にチェックを済ませておく。
- □ セルフチェックおよびレビューに要した工数実績データを記録・累積しておき、セルフチェックやレビューにかけるべき妥当な時間を割り出し、効果的なセルフチェック資料を作成する。
- □ 中間レビューおよび本レビューの二段階レビューを行う。中間レビューでは基本的な理解や方向性の間違いをチェックし、本レビューでは重要機能やリスク機能に絞ったレビューを行う。
  何から何まで全てをレビューすることは不可能であるということを認識しておく。
- □ 影響度の確認レビューを行う。
- □ ミスや不具合の傾向をデータに基づいて分析・把握し、関連する重度障害や再発の多い不具合を重点的にレビューする。

管理

◎効果的なレビューのポイント

事前の準備、事前のセルフチェック、二段階レビュー、影響度の確認、過去の失敗

【レビューの基本的な流れ】（図５－２－２）



158 **レビュー** 工程毎の成果物に対して、その成果物が意図された通りに作成されたかどうかを検証し誤りを発見する行為のこと。

◆【設計レビューのポイント】

□ 前提として、設計は重要な仕様順に実行すること。
□ 事前に説明する設計骨子を整理しておくこと。
□ 事前に自己レビューを行うこと。
　□ 詳細にわたるレビューは自己レビューの段階で済ませておき、単純なミスはこの段階で排除しておく。
　□ 頻度の高い設計ミスなどを網羅した、自己レビュー用の基本的なチェックシートを用意しておく。
□ 設計骨子の重要な順に、理解した要求仕様内容および実現方法としての設計内容を説明する。
□ 中間レビューと本レビューの二段階レビューを行う。
　□ 中間レビューにおいて、重要仕様の理解が過不足なく設計に反映されているかどうかをチェックする。
　□ 本レビューにおいて、重要仕様の最終チェックと同時に、時間の許す限り他の仕様のチェックを行う。
□ レビュー内容のポイント
　□ 不明仕様を自分の想定によって設計している部分はないか。想定部は速やかに確定させる。
　□ 要求仕様の意味を正しく理解しているか。
　□ 入力条件は正しいか、処理のロジックは正しいか、出力は正しいか。
　□ 異常処理は適切か。
　□ レスポンス・パフォーマンスの考慮は適切か。
□疑問点・不明点は都度レビュアーに判断を仰ぐこと。

効果的なレビューを実現するためのドキュメントの条件

レビューがうまく行かない原因として、ドキュメントの内容が貧弱であることおよび各ドキュメント間の仕様のひもづけがされていないことがあります。主要なドキュメントである、要求仕様書・基本設計書・詳細設計書・テスト設計書においては同じ仕様について要求仕様番号と同様な番号を付番することで一気通貫的にひもづけを行い、要求仕様書と同様な構造（ＷＢＳ）を持つような記述をする必要があります。

すべてのドキュメントが、同じ思想のもとで同じ構造化が行われていれば、どの工程の担当者においても仕様の理解が容易になると同時に、ドキュメントのメンテナンスも容易になり、当然のことに誤記・記入抜け・バラバラな記述・分かりにくい記述の発見も容易になり、それぞれのレビューもスムーズになります。

このように全てのドキュメントが共通の構造のもとに統一されていれば、作成されるプログラムも自ずと構造化されやすくなり、プログラム作成の精度が向上すると同時にメンテナンスも容易になるはずです。

管理

◎ドキュメントの条件

要求仕様書・基本設計書・詳細設計書・テスト設計書は、共通なＷＢＳ構造を持つこと。

【共通なＷＢＳ構造のイメージ】（表５－２－２）

	レベル１：システムＡ					
	レベル２		レベル３		レベル４	
	項番	成果物	項番	成果物	項番	成果物
要求仕様書	1	要求仕様書	Ｒ１－１	業務アプリ１	Ｒ１－１－１	業務１－１．要求仕様
					Ｒ１－１－２	業務１－２．要求仕様
					・・・	・・・
基本設計書	1	基本設計書	Ｂ１－１	業務アプリ１	Ｂ１－１－１	業務１－１．基本設計
					Ｂ１－１－２	業務１－２．基本設計
					・・・	・・・
詳細設計書	1	詳細設計書	Ｄ１－１	業務アプリ１	Ｄ１－１－１	業務１－１．詳細設計
					Ｄ１－１－２	業務１－２．詳細設計
					・・・	・・・
テスト設計書	1	テスト設計書	Ｔ１－１	業務アプリ１	Ｔ１－１－１	業務１－１．テスト設計
					Ｔ１－１－２	業務１－２．テスト設計
					・・・	・・・

要求仕様書からテスト設計書に至るまで、同じ仕様や機能に関するもの同士は、同様の項番を附番することによって、次のことが可能になる。

- ドキュメントの構造化
- ドキュメント相互の関係性を容易に追跡可能にする。
- 仕様の矛盾やモレなどの発見を容易にする。

外注納品物の品質管理

ソフトウェア開発の元請ベンダーにおいては、近年ますます外注ソフト開発会社に対する依存度が高まって来ており、それらの外注会社から納品されるソフトウェアの品質確保が重要な課題になっています。

外注化は、主に設計・製造・評価テストなどの工程において進んでいますが、本章では外注会社の請負範囲を設計・製造および結合テストまでとした、ソフトウェアの受入れ検査について解説をします。

外注化されるものに対しての品質基準は、社内における品質基準と同等レベルのものを設定する必要があり、それらは第３章の３－７．●品質目標値の設定において記載した品質指標と同等なものです。

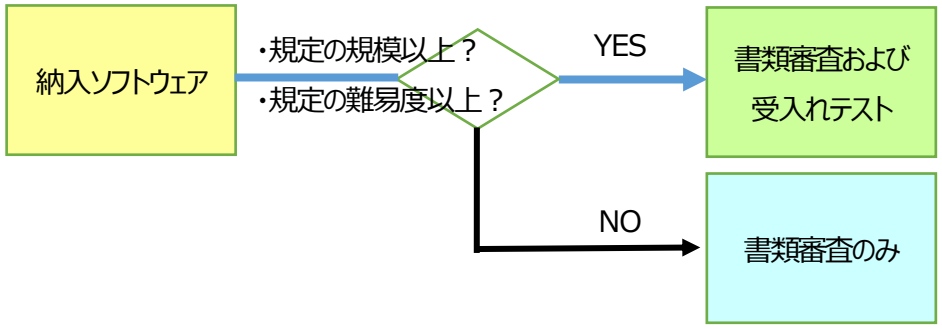
外注から納品されるソフトウェアは、「ソフトウェア受入検査基準管理表」などの書面により、納入されるソフトウェアの品質が、要求された基準を満たしていることが証明される必要があります。

納入ソフトウェアの受入判定は、書類審査と受入テストの二つの方法で行われますが、開発規模の小さなものおよび難易度が低いと判断されるものは、規定に従って書類審査のみの場合もあります。

管理

◎外注に委託した開発作業の品質基準は、社内基準と同等のものを適用すること。

【納入ソフトウェアの受入れ判定】（図５－２－３）



書類審査

外注会社は、以下の資料の提出が求められます。

1．ソフトウェア受入検査基準管理表

単体テストおよび結合テスト、それぞれのテスト件数、テスト時間、テスト密度、テスト不具合密度を記入し、受入基準を満たしていることを証明するもの。

2．テスト確認表（単体テスト・結合テスト）

3．テスト仕様書（テストチェックリスト、単体テスト・結合テスト）

4．テスト不具合一覧表（単体テスト・結合テスト）

5．テスト進捗表（単体テスト・結合テスト）

6．詳細設計書

7．ソースコード

受入検査基準管理表のサンプルは次の通りです。

【受入検査基準管理表】（表５－２－３）

テスト密度・不具合密度	合格基準値	単体テスト	結合テスト
テスト密度 （テスト件数／kstep）	○○件以上		
不具合密度 （不具合件数／kstep）	○○件以下		
テスト件数			
テスト時間			

付帯納入成果物の確認	受領日
テスト確認表	
テスト仕様書	
テスト不具合一覧表	
テスト進捗表	
詳細設計書	
ソースコード	

受入テスト審査

所定の開発規模以上ないしは難易度レベル以上のものは、書類審査に合格した上で、受入テストによる審査を必要とします。 受入テストの実施要領は次のようになります。

重要仕様部分の動作確認テストを行い、受入検査基準管理書に記載された不具合密度の妥当性を検証し、納入された付帯成果物における不具合の再発や類似不具合の発生がないか等の確認を行う。

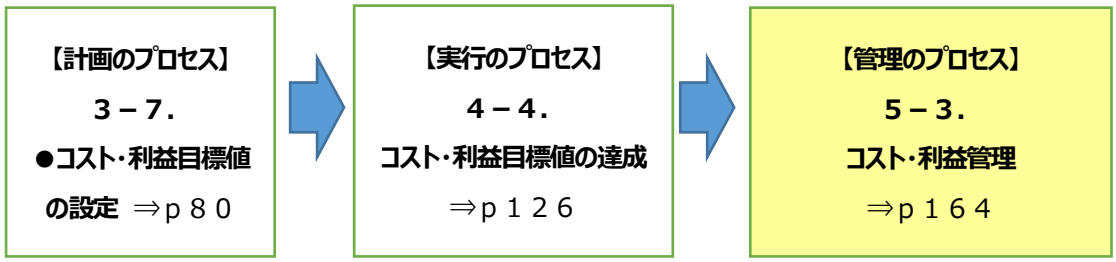
提出されたエビデンス[159]と相違ない結果で新たな不具合が発見されなければ合格とされます。

---

159 **エビデンス** 証拠または根拠を示すもの。

5－3 コスト・利益管理【コストマネジメント】

これまでのコスト・利益管理に関する解説の流れを図示すると下記のようになります。



第３章の３－７．においてはコスト・利益目標値の設定方法について、第４章の４－４．においてはコスト・利益目標の達成活動について解説をしました。本章においてはこれらの設定されたコスト・利益目標値に対するモニタリングおよびコントロールについて解説をします。

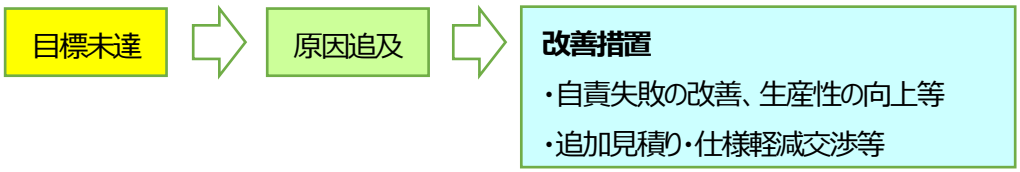
コスト・利益管理の役割

最初にお断りしておきますが、開発が開始された後に実行されるコスト・利益目標値のモニタリングおよびコントロールはコスト・利益目標を達成するための直接的な行為ではないということを認識しておく必要があります。いくら精密にコスト・利益の数値の追跡を行ったとしても、基本的にはコスト・利益の目標が達成されることはありません。 目標値を達成するための直接的な行為は、計画のプロセスにおける要求仕様の早期凍結や妥当な見積り交渉の実行および実行のプロセスにおける改善活動の実行などであり、これらの成果が確実に出ているかどうかを経理的な数値で追跡し、不十分な活動の是正を行うことが、本章のコスト・利益管理の役割です。

管理

◎コスト・利益管理表は目標の達成状況の追跡と是正措置のためのもの。
目標達成の直接的な行為は、要求仕様の早期凍結・見積り交渉・改善活動による。

【コスト・利益目標未達成に対する是正措置】（図５－３－１）



管理

原価管理表

コスト・利益を管理する損益管理表の構成については第３章の３－７．において説明した通りですが、コスト・利益目標値のモニタリングおよびコントロールについても損益管理表と同様の構成を持つ原価管理表[160]を用います。

プロジェクトの損益の目標値としては最初に粗利率の設定を行います。
粗利は、「粗利＝売上高―製造原価」と定義されている通りで、顧客から受注した金額が売上高となり、製造原価はプロジェクトが直接的に消費するすべての開発費用です。

管理

◎粗利＝売上高（受注金額）―原価（直接的な開発費用）

粗利率は、「粗利率＝粗利金額÷売上高」で表される通り、粗利の売上高に対する比率を表わすもので、粗利ベースでの利益率のことです。
粗利率の設定については会社の方針によってさまざまであり、各社において設定されている最低の粗利率以上の数値の設定が求められるでしょう。

売上高が決まり、粗利率が決められていれば、製造原価は自ずと決まってしまいます。
例えば、売上高が１００万円、粗利率が２０％（２０万円）とした場合、製造原価は８０万円以下にする必要があります。
結局プロジェクトが所定の目標利益を達成するためには、所定の粗利率を達成する必要があり、さらには所定の製造原価以内で損益管理をしなければならないと言うことです。

管理

◎コスト・利益の管理は、最初に粗利率の設定から始まる。

実際のプロジェクトにおいては多くのリスクが潜在しているため、会社が決めている最低限の粗利率で目標を設定していたのでは、一般的にその目標の達成は困難です。有能なプロマネにおいては、常にリスクを念頭において、会社設定の粗利率よりも大きな粗利率を設定していることでしょう。
しかし粗利率を大きくするためには売上高を大きくするか、原価を小さくするかという二つの方法しかありません。

---

160 **原価管理表** 原価低減の施策が適切に実行されているのかを判断し素早い是正措置を実行することでコスト・利益の目標の達成を確実にするために使用される管理表。

管理

売上高の拡大については、「３－５．●ソフトウェアの価格」で述べた通りですが、この方法を実行するためには、緻密な理論武装とタフな交渉力が必要となります。

普通のプロマネに残された手段は、製造原価の低減すなわち、要求仕様の早期凍結・妥当な見積り交渉の実行・改善活動の実行などを、開発の初期段階から確実に実行することです。

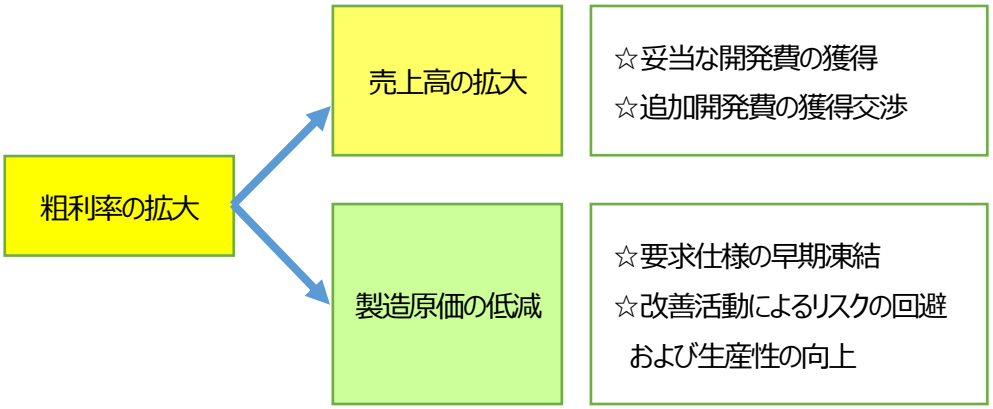
原価管理表によるコスト・利益の管理は、これらの原価低減の施策が適切に実行されているのかを判断し、素早い是正措置を実行することで、目標の達成を確実にするために使用されるものです。

管理

◎原価管理表の役割

① 予算と実績の差異とその理由を明確することで、
② 不適切な開発行為を是正し、
③ 目標原価を達成すること。

【粗利率の拡大】（図５－３－２）



原価管理表のサンプルは次に示した通りです。拡大図は「付録図表１４．原価管理表」を参照のこと。

【原価管理表】（表５－３－１）

【原価管理表】	売上高：xxxxx		＊%はすべて対売上高の数値						
プロジェクト名：			4月	5月	6月	7月	8月	9月	合計
予算 開発原価予算	直接材料費①	外注費							
		購入品等							
		小計							
	直接労務費②	開発人件費							
		経費							
		小計							
	間接費③	開発間接費							
	予算合計①＋②＋③	金額							
		累計金額							
		累計%							
実績 開発原価実績	直接材料費①	外注費							
		購入品等							
		小計							
	直接労務費②	開発人件費							
		経費							
		小計							
	間接費③	開発間接費							
	実績合計①＋②＋③	金額							
		累計金額							
		累計%							
（予算－実績）差額 開発原価差額	直接材料費①	外注費							
		購入品等							
		小計							
	直接労務費②	開発人件費							
		経費							
		小計							
	間接費③	開発間接費							
	差額合計①＋②＋③	金額							
		累計金額							
		累計%							
予算超過・消費不足の理由									

損益管理【予算合計】

	金額	%
売上高		
原価		
粗利		

損益管理【実績合計】

	金額	%
売上高		
原価		
粗利		

損益管理【差額合計（予算－実績）】

	金額	%
売上高		
原価		
粗利		

上記の原価管理表は、通常の原価管理に損益管理および毎月の累計管理までを含めたものです。単に毎月のコスト目標値の達成度を見るだけではなく、常に売上高に対する開発費の消費状況を監視することで、開発費消費実績の予算からの乖離および累計値の異常を早期に発見することができ、開発問題の早期発見・早期是正を可能にするものです。

上記の表に使用された経理用語の意味は次の通りです。

◎直接材料費

外注費　：開発委託を行った外注費は、調達部品扱いとし直接材料費の範疇に含まれています。

購入品費　：開発に直接要した製品・半製品などの材料費のことで、開発ツールや備品等の購入費等。

◎直接労務費

開発人件費　：開発のために消費される社内開発者の直接作業時間の人件費。

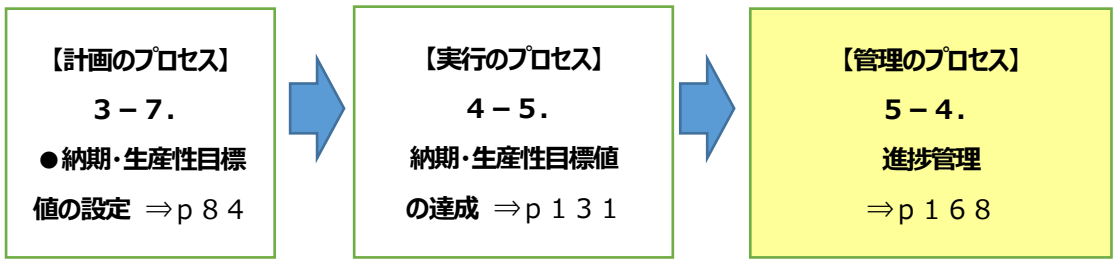
経費　：開発業務に付随して使用される旅費・交通費・通信費・消耗品等の費用。

◎間接費

開発を間接的にサポートする開発管理部門等の費用で、一般的には開発予算の規模に応じて一定の割合の負担が配分されている。

5－4 進捗管理【タイムマネジメント】

これまでの納期・生産性管理に関する解説の流れを図示すると下記のようになります。



第３章の３－７．においては納期・生産性目標値の設定方法について、第４章の４－５．においては納期・生産性目標値の達成活動について解説をしました。本章においてはこれらの設定された納期・生産性目標値の達成状況をモノ・カネ・ヒトの三つの視点による進捗管理で見て行くことにします。

第３章の計画のプロセス「３－６．スケジュールの作成」においてＷＢＳに基づいた開発スケジュールを作成しましたが、計画されたスケジュールを予定通りに進捗させるための方法について解説を行います。

プロジェクトの進捗管理といえば、モノという成果物を生み出す要件定義・設計・製造の進捗ばかりに目を向けがちになりますが、プロジェクトを成功裏に完了するためにはヒト・モノ・カネの三点すべてに対する進捗管理が必要になります。

開発の半ばにおいてプロダクト成果物の進捗がオンスケジュールであったとしても、予算の消費が８０％にもなっていればプロジェクトは大赤字になってしまうことは確実で、また開発者の平均残業時間が月１００時間を超えてしまっていればプロジェクトの遂行は困難になってしまいます。

納期・利益第一主義的風潮が強い状況下にあって、モノの進捗管理一辺倒では顧客が納得できるＱＣＤの達成は難しいでしょう。ヒト・モノ・カネのリソースが健全に使用されて初めて妥当なＱＣＤ目標が成立し、顧客の満足が得られる製品の開発が可能になります。

管理

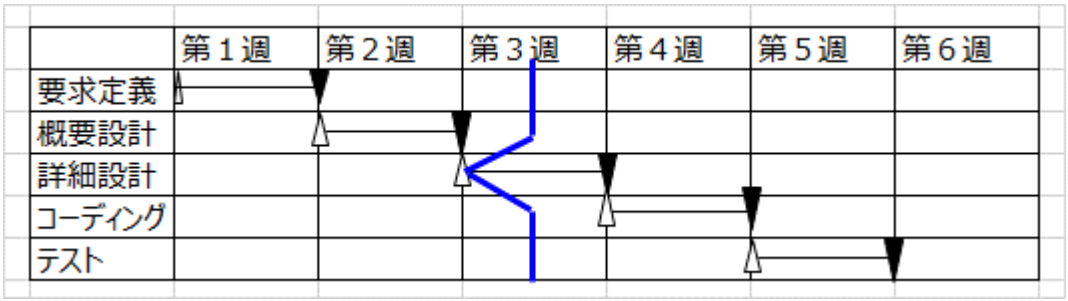
◎進捗管理の三点セット

１．モノの進捗管理　　：プロダクト成果物進捗管理
２．カネの進捗管理　　：予算消費進捗管理
３．ヒトの消耗度管理　　：残業時間管理

モノの進捗管理（プロダクト成果物進捗管理）

ソフトウェア開発の進捗管理には、一般的に下記に示したようなガントチャート[161]が用いられています。ガントチャートは視覚的に分かりやすく、大まかな進捗を管理するのには便利な管理表だと言えます。

【ガントチャート】（表５－４－１）



ガントチャートは日程表に進捗線を入れたもので管理されますが、この表ではタスクの正確な進捗を見ることができません。進捗線が１週間遅れの様に記入されていたとしても、本当に１週間なのかどうか記入した本人ですら分からないのが実状なのです。ひょっとしたら２週間遅れなのかも知れません。

「ソフトウエアの開発スケジュールは、納期の直前までは順調に進む（木下恂著、ソフトウエアの法則）」というような警句にもある通り、報告を受けたガントチャート上の最後の２週間遅れが、一か月を経過しても解消されないこともあります。

ガントチャートは大まかな進捗を見るのには適した表現方法ですが、おおまかさを補うために次に示した機能モジュール進捗管理表[162]の併用が有効です。機能モジュール進捗管理表は縦軸に開発タスクを最小レベルまでブレークダウンしたソフトウエアモジュール名を記載し、横軸にはそのプログラムの見込みサイズ、開発各工程の終了予定日・実績日を記入して予定日との差異を管理するものです。最小プログラムモジュール単位までブレークダウンされているため、主観の入る余地が少なく正確な進捗が見えやすいものとなっています。

機能モジュール進捗管理表は、いわばＷＢＳ的なモノの進捗管理ないしは開発物の単品管理[163]ともいえるものです。

機能モジュール進捗管理表のサンプルを次に示します。

---

161 **ガントチャート** ガントチャートは日程表に進捗線を入れたもので、視覚的に分かりやすく大まかな進捗を管理するのには便利な管理表だと言えます。

162 **機能モジュール進捗管理表** 縦軸に開発タスクを最小レベルまでブレークダウンしたソフトウエアモジュール名を記載し、横軸にはそのプログラムの見込みサイズ、開発各工程の終了予定日・実績日を記入して予定日との差異を管理するもので、いわばＷＢＳ的なモノの進捗管理ないしは開発物の単品管理ともいえる。

163 **単品管理** モノの無駄をなくすために、商品を一品ごと、販売・仕入・在庫の管理を行う手法。コンビニの単品管理は好事例。

管理

【機能モジュール進捗管理表（部分）】（表５－４－２）

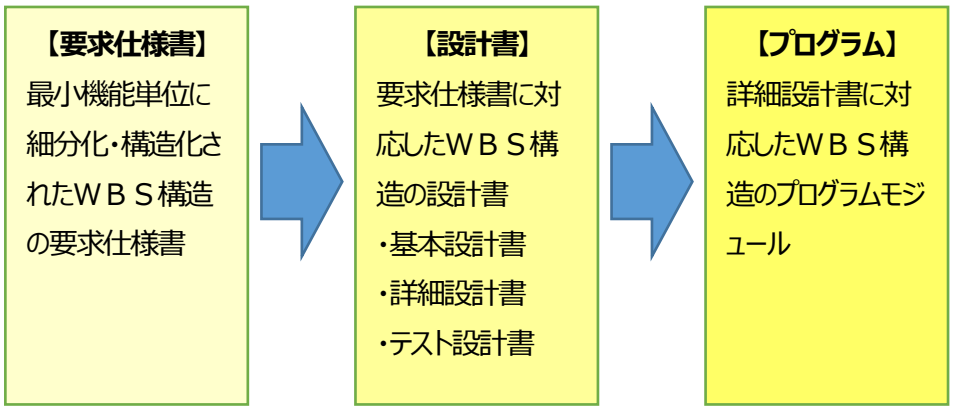
業務	プログラム名称	ＥＸＥ，ＤＬＬ、ＯＣＸ等名称	開発Ｖｏｌ（Kbyte）		担当者	開発状況	コーディング（Ｃ）		単体評価（Ｕ）	
			見込	実績			完了（予定日）	完了（実績日）	完了（予定日）	完了（実績日）
	ＸＸＸ受信プロセス	ｘｘｘｘｘ.exe	50	60	A、B	C	6/30	6/30	8/15	8/15
	ＸＸＸ送信プロセス	ｘｘｘｘｘ.exe	45	40	A、B	C	6/30	6/30	8/15	8/15
	通信プロセス	○○.exe	25	30	A	U	6/30	6/30	7/15	7/15
	リカバリー応答プロセス	○○RECOV.exe	35	30	A、B	U	6/30	6/30	7/15	7/15
	△△受送信プロセス	△△RECV.exe	25	20	A	K	5/30	5/30	6/15	6/15
	△△送受信プロセス	△△SEND.exe	20	30	A	K	5/30	5/30	6/15	6/15
	△△起動問合せ	△△INQ.exe	30	10	B	K	5/30	5/30	6/15	6/15
	ＦＴＰプロセス	○○FTP.exe	20	30	C	K	5/30	5/30	6/15	6/15
	最小モジュール単位までブレークダウンする		量的把握の見込み／実績を行う		担当者の明確化		工程別進捗を予定／実績日程にて行い。色彩にても直感しやすくする			

詳細拡大図は付録図表の「１５．機能モジュール進捗管理表」を参照のこと。

管理

◎モノの進捗管理は、単品管理の思想で行うこと。

【最小機能単位で作成管理される開発成果物】（図５－４－１）



参照：ｐ１６１　共通なＷＢＳ構造のイメージ（表５－２－２）

カネの進捗管理（予算消費進捗管理）

カネすなわち開発費の消費状況の管理は原価管理表だけでは不十分です。原価管理表を見て、今の時点において設計業務にいくら開発費を使っていくら残っているのかを即答できるプロマネはいないでしょう。

開発費の見積りは、業務の種類別すなわち開発工程別の作業に分解されたものの見積りの合計額であり、例えば要件定義工程・設計工程・製造工程・評価テスト工程それぞれに必要な費用が見積もられています。プロジェクトが健全に運営されるためには、各工程の業務がその見積り予算を目標に遂行されているか否かをモニターする必要があります。

工程別の費用配分・期間配分の例を次に示します。

【開発工程における工数比率・期間比率】（表５－４－３）

工程	要件定義	基本設計	詳細設計	製造	総合テスト	合計
工数比率	15.0%	13.0%	12.0%	40.0%	20.0%	100.0%
期間比率	25.0%	15.0%	12.0%	25.0%	23.0%	100.0%

出典：日経ＩＴプロフェッショナル２００２年１０月（ＩＢＭ：１４０頁）

調査対象：小規模プロジェクト（１００人以下のプロジェクト、１０数万ステップ規模）

注．筆者にて改変または推定した部分：

①原典において表記の外部設計、内部設計はそれぞれ基本設計、詳細設計の表記に置き換えました。

②原典における統合テストおよびシステムテストは、総合テストとしてまとめました。

③単体テスト・結合テストは製造に含まれているものと推定しました。

上記のデータはほぼ現実の経験値と一致しています。但し未経験分野の新規プロジェクトにおける総合テスト工数比率は３０％程度に膨らむ可能性があります。

読者において現在担当されているプロジェクトにおける数値と対比してみてはいかがでしょうか。何か別の気づきがあるかも知れません。

例えばあるプロジェクトの見積りが上記のデータのようであった場合で、要件定義工程の実績工数比率が２５％になろうとしていた場合は、要件定義工程が遅延していることになります。その他の工程についても同様であり、見積り時の数値から大きく乖離している状況が発生しそうな場合は、その工程に何らかの異常事態が起きている証拠であり早急な是正措置が必要になります。

開発業務の健康状態は開発費の消費状況にはっきりと表れてきます。

管理

各工程における開発予算と消費実績を管理するためには、通常の原価管理表とは別に次のような工程別原価管理表である予算進捗管理表[164]が必要になってきます。

【工程別予算進捗管理表】（表５－４－４）

	工程	要件定義	基本設計	詳細設計	製造	総合テスト	合計
予算	金額						
	%						
実績	金額						
	%						
差額	金額						
	%						

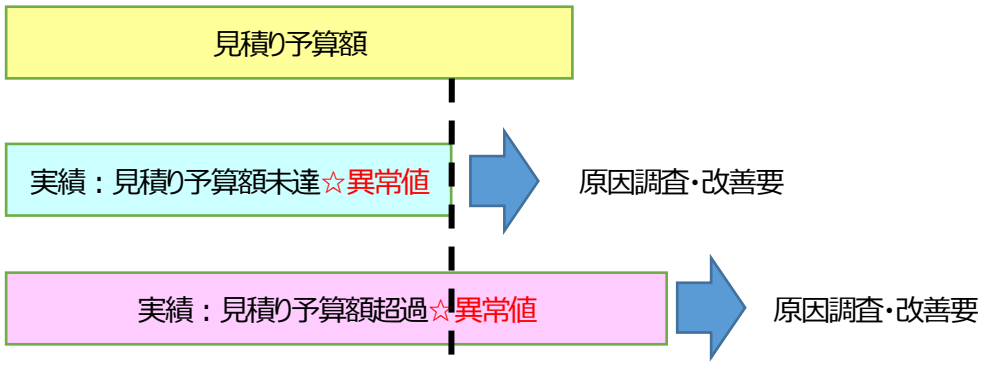
各工程の予算と実績に乖離が出そうな場合には、その工程に問題があるということで、予算オーバーのみならず予算未達の場合においても、その原因を調べ早急な対応が必要となります。

少しくらいの乖離ならばいいかと放置していると、詳細設計が終わるころまでに、本来は全予算の４０％程度の消費であるべきところが８０％も消費していたというような状況に陥る危険性もあります。

管理

◎開発費の消費状況からプロジェクトの健全性を見るためには、
工程別の予算進捗管理が必要になる。

【予算消費の健全性確認】（図５－４－２）



164 **予算進捗管理表** 各工程の予算と実績の乖離状況を把握することで、資金面におけるプロジェクトの健全性をチェックする管理表。

ヒトの消耗度管理（時間外労働時間管理）

開発を進める主体はあくまでも開発者であるヒトにあります。開発費の健全性を管理するために原価管理表や工程別予算進捗管理表を用いましたが、ヒトの健全性を確保するための管理も必要です。

進捗管理の横軸は時間や期間ですが、開発時間の経過と共に開発者たちの疲労度は間違いなく増加していき、過重労働を放置したままではプロジェクトは必ず行き詰ります。
ヒトの消耗度管理は進捗管理の中では、ある意味で最も重要な管理表だとも言えます。中核の開発者が倒れてしまえば、全ての管理業務は全く機能できなくなります。

ヒトの消耗度管理は、基本的に毎月の時間外労働時間の管理をするということですが、単に６０時間や８０時間を超えないように管理するだけではなく、個々人の体調を良く把握した上で制御する必要があります。プロマネにおいては自分自身の健康管理も含め、メンバー全員に対して以下のような消耗度の管理が必要です。

① 全員の健康状態について常に気を配ること。
② 危ない状況の人に対しては即刻休ませること。
③ 欠員が発生する前に他からの人材の補充を行うこと。
④ 法定・協定における時間外労働はしない、させないこと。
⑤ 超過残業や休日出勤に対しては、直後に必ず代休や半日休暇を設けること。
⑥ 特定の人間に負荷が集中しないように柔軟に人員体制を組み替えること。

上記のような施策を可能にするためには、本書の全てにおいて言及しているような、妥当性に基づいた合理的なプロジェクト運営を行う必要があります。プロジェクトから一人たりとも犠牲者を出さないという心構えが必要です。これはプロマネの人間としての最大の義務だと言えます。

管理

プロジェクトメンバーの労働負荷状況の管理表[165]の例を次に示します。

【労働負荷状況管理表】（表５－４－５）

プロジェクトメンバー氏名	先々月残業時間	先月残業時間	当月残業時間	３か月合計	１ヶ月平均残業時間	労働負荷レベル	労働負荷状況の理由	負荷軽減対策

労働負荷レベル

５：非常に過度　：残業８１時間／月以上
４：過度　：残業８０時間～４１時間／月
３：普通　：残業４０時間～２１時間／月
２：軽度　：残業２０時間～１１時間／月
１：非常に軽度　：残業１０時間～０時間／月

拡大図は「付録図表１６．労働負荷状況管理表」を参照のこと。

【負荷ブラック度判定表】（表５－４－６）

プロジェクトメンバー氏名	先々月残業時間	先月残業時間	当月残業時間	３か月平均
A	4	5	5	5
B	3	4	4	4
C	2	3	4	3
D	2	3	4	3
人員数	4	4	4	4
ブラック人数	1	2	4	2
ブラック度%	25%	50%	100%	50%
是正の要・不要	不要	要	要	要

165 **労働負荷状況管理表**　開発者の人的な消耗度を毎月の時間外労働時間によって管理するもの。

【Human Activity】

5－h 1 コミュニケーションの活性化【コミュニケーションマネジメント】

わたしたちの日常のコミュニケーションの状況を振り返ってみると、ＩＴエンジニアに限らず実はほとんどの人が、自分はコミュニケーションの問題を抱えていると思っているのではないでしょうか。実際、コミュニケーションにおいては、聞く・話す・読む・書くという四つの基本的なスキルに加えて、考える力が必要とされ、このような難しい技をスイスイとこなす人などそうそういるわけがありません。交渉事やプレゼンが難しいのは当たり前のことで、自分だけがコミュニケーション下手で全くダメだと思う必要はありません。

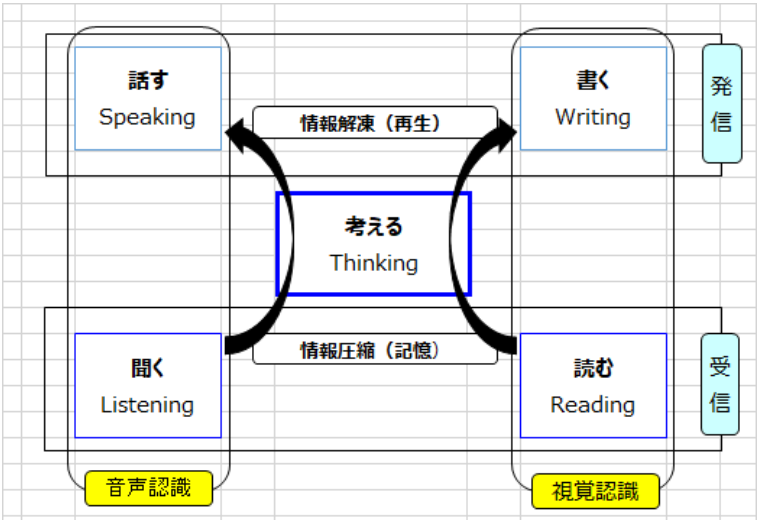
開発現場でよく聞かれるコミュニケーションの悩みとしては次のようなものがあります。

- 人と話をするのが苦手。
- 自分の考えや意見をうまく伝えられない。
- プレゼン能力を上げたい。
- 大勢の人がいる場では意見を言い出せない。
- 仕事の指示の仕方がうまくいかない。

コミュニケーション能力の構造

コミュニケーションの能力を図で表すと次のようになります。

【コミュニケーション能力の構造】（図５－ｈ１－１）



「聞く」情報は圧縮され脳内に記憶され、また脳によって解凍され「話す」情報として再生されます。同様に、文字や図は「読む（見る）」ことによって脳内に記憶され、また脳によって解凍され「書く（描く）」情報として再生されます。

このことでも分かるように「話す」前の段階に「聞く」ことがあり、「書く」前の段階に「読む」ことがあるのです。

多くの人たちは、早く上手に話せるようになりたい、上手に文章が書けるようになりたいと願って、話し方や書き方にばかり気を奪われていますが、本当に上手になりたいのなら、人の話を良く聞き、多くの書物を読むことに注力した方が良いと思われます。

コミュニケーション能力のレベル差は、量的には、どれだけ人の話をたくさん聞いたか、どれだけたくさんの書物を読んだかによって生じ、また質的には、聞いた情報・読んだ情報の圧縮率（記憶率）および解凍率（再生率）に依存しているのでしょう。

圧縮率や解凍率は、頭脳の良し悪しにも依存しているのでしょうが、人の興味の度合いや感動の度合いが最も強く影響していると思われます。われわれ凡人においても興味や感動は天才・秀才たちに劣らないものを持っているはずで、良く聞くこと、良く読むことから始めたいと思います。

ソフトウェア開発の現場におけるコミュニケーション

それではソフトウェア開発の現場におけるさまざまな問題の中でコミュニケーションの問題がどの程度の割合を占めているのでしょうか。

下記は、筆者において収集した５００の事例における、さまざまな問題の多さのワースト５を示したものです。

【プロジェクト問題のワースト５】

失敗の原因	件数／%
１．コミュニケーション不良	６６件　１３%
２．リスク回避の失敗	６５件　１２%
３．手抜き問題	４６件　９%
４．人材育成／ノウハウの継承問題	４２件　８%
５．ドキュメント・ベース開発の欠如	３８件　７%

管理

◎プロジェクト問題の原因№１は、コミュニケーションの不良にある。

ソフトウェア開発プロジェクトにおける問題の件数が最も多いのは、コミュニケーションの不良でした。これから見ても、コミュニケーションの不良がプロジェクト全体に及ぼす影響の大きさが分かっていただけたことでしょう。この事実は、コミュニケーションの不良が個々の開発者たちの業務品質、製品品質、生産性に大きな悪影響を及ぼしており、さらにはプロジェクト組織をコントロールするリーダー、プロマネおよびＳＥなどにおけるコミュニケーション能力が、その責任や役割を果たすためにどれほど重要なのかを物語っています。

管理

コミュニケーションの役割と影響力

コミュニケーションの役割は次の通りです。

- **人的な情報系の中枢を司る**
- **何をどのように実行すべきかを明らかにする**
- **共通の目的に向かって関係者全員のベクトルを揃える**

顧客とのコミュニケーションは、プロジェクトの起点である要求仕様の質に決定的な影響を及ぼします。さらにその影響は見積りに及び開発に許される時間と資金を決定的に規定してしまいます。続いてプロジェクトにおけるコミュニケーションは、開発者たちの思考や行動の質を支配することになり、それは製品のＱＣＤ全体に決定的な影響を及ぼすことになります。すなわちコミュニケーションは、全ての開発行為の質と量に大きな影響を及ぼしているということが分かります。

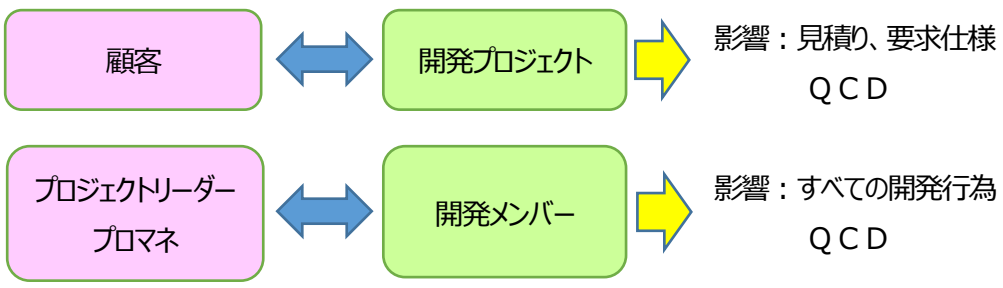
ソフトウェア開発のすべての工程で最も重要なことは、顧客との間、およびチーム内で良好なコミュニケーションが実行されなければいけないということです。潤沢な開発費、余裕のある開発期間、整備された開発環境がそろっていたとしても、孤立した人間関係しか持てないチームは必ず失敗を招くことになります。

良好な意思疎通は、共通の目的に向かって関係者全員のベクトル[166]を揃え、実行すべきことは何か（What）を明らかにし、それをどのように実現するのか（How）を明らかにします。開発は、人と人との関係性の中からしか生み出され得ないものだということを、もう一度強く認識する必要があります。

管理

◎コミュニケーションは、要件定義、見積りおよび全ての開発行為に大きな影響を及ぼす。

【コミュニケーションの質と量が及ぼす影響】（図５－ｈ１－２）



166 **ベクトル** 力がかかる方向のこと。

管理

良好なコミュニケーションを行うためのポイント

コミュニケーションに関する主なリスクは次の通りです。

① 顧客・開発チーム間の意思疎通・情報共有の不良

② 開発チーム内の意思疎通・情報共有の不良

プロジェクトにおけるコミュニケーションリスクの排除のためのチェックリストを次に示します。

◆**【コミュニケーション（基礎編）】** ▶Check Timing：会議・会話の前・中・後

- □ 顧客との会議に、予想される想定問答など十分な事前準備を行って臨んでいるか。
- □ 他人の評判だけに従って自分の行動を決めてはいないか。
- □ 発言すべきか否かを、必要性によらず感情によって決めてはいないか。
- □ 先に相手の話を聞くなど、丁寧なコミュニケーションを行っているか。
- □ 相手の話の途中に割り込むようなことをしてはいないか。
- □ 毎日、短時間でもコミュニケーションを継続することで、良好な人間関係を維持しているか。

他人からの評判だけに従って自分の行動を決めてはいけない

自分の所属する集団の反応によって自分の行動を決めるという姿勢は、日本人全般にみられる習慣の一つです。これは日本人の稲作農耕に由来する伝統的慣習である集団行動主義[167]に起因したものと考えられます。

その良い面としては、集団全員の思考・行動の方向性を一つにまとめ、困難な目標を達成する強力な手段となりますが、反面個々人の自主的な思考・行動を制限することにより、自律性を損ない集団を誤った方向に導きやすいという負の局面を持っています。いわゆる「空気を読む」という行動はこの延長線上にあり、過度に他人の思いを意識することで、個人の自由闊達な思考・行動を押さえ込んでしまいがちです。

他人の目を過度に意識することにエネルギーを消耗させるのではなく、”良い仕事をするためには何をすべきか”ということに気持ちを集中させれば、もっと自由闊達なコミュニケーションが行われることでしょう。その結果、報告・連絡・相談も自発的に頻繁に行われ、良いコミュニケーションが信用・信頼を醸成し、仕事のスキルも向上できるに違いありません。

管理

◎自由闊達なコミュニケーションは、
他人に意識を向けるよりも仕事に意識を向けることから始まる。

167 **集団行動主義** 同じ目的に向って、全員でそれぞれの役割分担に従い、協調した行動をとること。

管理

◆【コミュニケーション（中級編）】 ▶Check Timing：会議・会話の前・中・後

- □ 話をする前に、自分の中で伝えたい内容を整理しているか。
- □ 客観的なドキュメントや数値データに基づいてコミュニケーションを行っているか。
- □ 重要な取り決めは、相互の合意を文書にて残しているか。
- □ 相手の話を良く聞かずに、自分の意見ばかりを話していないか。
- □ 事実を伝えるよりも良く見せることに注意が傾いてはいないか。
- □ 恥や外聞を恐れ、不明点・疑問点をその場で確認することを避けてはいないか。
- □ 合理的かつ妥当な結論を導き出すことよりも、議論の勝ち負けにこだわってはいないか。
- □ 伝えるべきこと、伝えてはいけないことをわきまえているか。

話し言葉だけでは伝わりにくい

相手に話をうまく伝えるためのポイントは次の二つに絞られます。

① 客観的な資料や数値データを示しながら説明をすること。

② 相手の能力レベルに合わせた話し方をすること。

口頭でのコミュニケーションは、感情的ないしは情緒的な表現に傾きがちで、あいまいさが混入しやすいものです。これらのあいまいさを排除するためには、資料を使って数値やデータを示しながら話をすることが一番です。

また話す相手の知性や技術的能力のレベルに合った話し方をしなければ話は伝わりません。話の要所ごとに相手の理解度を確認することが必要です。理解度の確認は、相手に理解した内容を復唱させれば分かるでしょう。また抽象度の高い内容は、具象的な例え話をすることで、相手の理解を深めることができます。話の上手な人ほど例え話がうまいものです。

簡単な事柄でも、何度も同じ質問を受けるようなら文書での伝達を

相手に口頭で伝えても良い場合は、すでにその相手がその仕事の基本的な意味や背景について共通の理解を持っていると判断できる場合だけに限られます。それ以外の場合は簡単な事柄についても、最低限自分のメモに基づいた伝達が必要であり、その内容についての相手の理解が正しいかどうかをその場で確認する必要があります。また、ある事柄について何度も複数の人から尋ねられたり、自分から言う必要性を感じたりした場合は、そのものごとについて複数の相手に理解が行き届いていないことになり、文書化・ドキュメントによる伝達が必要な時期が来たと考える必要があります。

◎伝わるコミュニケーションは、数値やデータに基づくことによる。

管理

管理

◆【コミュニケーション（上級編）】 ▶Check Timing：会議・会話の前・中・後

- □ 相手が最も聞きたいと思われることを最初に伝えているか。
- □ 相手の能力レベルに合わせた話し方をしているか。
- □ 仕事の指示や依頼において、相手の理解度を確認しているか。
- □ 会話を深めるために、相手との共有体験を持つことや教養の幅を広げているか。
- □ 会社間の重要事項に関するコミュニケーションは、リーダー対リーダーで行っているか。

相手が最も聞きたいと思われることを最初に伝えること

質問に対しては、まずは相手が聞きたいことは何かを十分に把握し、それに対して直接関係のあることを答える必要があります。いろいろな条件や見解については相手の反応をみてから順次話をすればよいことです。これは報告書などのビジネス文書でも同様です。最初に相手が聞きたい結論を語ることが重要です。その結論の説明や条件などは、結論の後に書かれていればよいでしょう。

また口頭による会話では一度に多くのことを話したくなるものです。特に不意打ち的な質問を受けた場合、自分の中で正解を探す時間を稼ぐために、色々と他の話題を出してしまうこともあるでしょう。このような場面を少なくするには、普段から仕事の懸案事項や問題点について良く考えておき、頭の中で整理をしておく必要があります。目先の仕事に追われつつも、全体を見通すことを怠らないようにすることが大切です。会話や報告においては、まずは、相手が聞きたいことを先に伝えるように努力したいものです。

管理

◎コミュニケーションにおいては、先に相手が知りたいことを伝え自分の知りたいことはその後で。

会議における良好なコミュニケーション

開発中に行われる会議は、重要なコミュニケーションのシーンの一つです。

何が議題なのかよく分からない会議、何も結論が出ない会議、誰がいつまでに何を実行するのかを決めない会議、議事録が作成されない会議、主催者のリード不足な井戸端会議、議題に余り関係のない人たちまで招集している保険担保型の無責任会議、などさまざまなどうでもいいような低生産性の会議が多過ぎ、プロジェクトのキーマンたちの貴重な時間を盛大に浪費していることが多いとは思いませんか。

生産性の高い有効な会議を実行するためのチェックリストを示します。

◆**【顧客との会議のポイント】** ▶Check Timing：顧客会議前

- □ 開発仕様の全体像の把握、仕様の目的・意味・背景を理解しておくこと。
- □ 事前準備・事前検討を行い、顧客からの想定質問への答えを用意しておくこと。
- □ 顧客からの宿題事項は、状況を聞かれる前に答えること。
- □ 自分の言いたいことよりも顧客の話を良く聞くこと。
- □ 疑問点・不明点については臆することなく、その場で確認を取ること。
- □ リーダークラスにおいては、発表力やプレゼンテーション力を強化するために、普段の開発業務の中で発表やプレゼンの機会を自ら求めること。
- □ リーダークラスにおいては、議事進行や調整能力を磨くために、社内会議等にて議長役や書記役を自ら買って出ること。

上記の内容は、顧客との会議だけではなく社内における会議においても有効なコミュニケーションの基本的な姿勢であるともいえます。

また顧客との密接なコミュニケーションを行うためには公式の会議の場だけではなく、顧客と接するあらゆるシーンを生かしたコミュニケーションが重要であり、幅広い会話を続けることで顧客とのフランクな信頼関係を構築していく必要があります。

管理

◆【会議運営のポイント】 ▶Check Timing：会議前

□ 会議の目的・背景・討議内容を事前に出席者に伝えているか。
□ 何を討議するのか事前に決めているか。
□ 何を決めるのか事前に明確にしているか。
□ 誰が・何を・いつまでに実行するのかを決めるようにしているか。
□ 不必要な人を招集してはいないか。
□ ダラダラ雑談に終了宣言を行うことができるか。

会議を主催するにあたって、これらのことを実行するだけでもプロジェクトの生産性向上に大きな貢献をすることになります。

管理

◎無駄のない会議は、確実にプロジェクトの生産性を向上させる。

会議問題に関する実態を下記に示します。

【会議問題の実態調査アンケート】（コンピュータ・情報サービス業）

◎会議時間が全体業務に占める割合　１７．９％

◎会議問題のワースト３

１．会議時間が長い　　５０．７％
２．無駄な会議が多い　　４６．１％
３．会議の頻度が多い　　３６．２％

◎企業価値に対する会議の貢献

（質問内容：会議は、仕事の生産性向上やイノベーションの創出に貢献していると思いますか）

・ 非常にそう思う　　１．３％
・ ややそう思う　　３２．２％
・ あまりそう思わない　　５０．０％
・ 全くそう思わない　　１６．４％

出典：ＮＴＴデータ経営研究所、「会議の革新とワークスタイル」に関する調査、2012 年 10 月 5 日
上記のデータは、全回答者数 1,090 名の内、コンピュータ・情報サービス業の回答者 152 名によるもの。

短時間日次情報共有会議の実行

情報共有という言葉が頻繁に使われてはいますが、本当の意味での情報共有を実行しているプロジェクトは意外に少ないと思われます。共有サーバに情報を載せたとか、メールで通知したとか、だけで効果的な情報共有を図ることはできません。ＩＴにて提供された情報は一見リアルタイム的に見えますが、見る人がすぐに見なければ時間ラグ[168]がある、単なるバッチ情報[169]にしか過ぎません。さらにＩＴ情報は人間の喜怒哀楽の微妙な程度を伝えることはできません。ＩＴにおける情報は忘備録程度の役割にしか過ぎないという認識を持った上で、効果的な情報共有を行うためには、直接対面コミュニケーションを行う必要があります。

毎日短時間で実行する日次情報共有会議は、プロジェクトにおいて最も効果的な情報共有を実現できる簡単な方法です。

日次情報共有会議の励行を

例えば、毎日２０～３０分程度の短時間日次会議を実行して、その中でチームメンバー全員から、「①昨日何を実行したか。②今日何を実行する予定か。③今抱えている問題は何か。」の３点について報告・応答を続ければ、「内部レビューで指摘した内容が反映されていない、仕様確認を怠ったまま設計・コーディングを行ってしまった、コーディングミスが多い、見直しをしていない、デバッグをしていない」というような低レベルの問題は毎日ベースで発見でき、毎日ベースで部下の教育・しつけもできるということになります。一ヵ月の進捗遅れが突然発覚するというようなこともなくなるでしょう。

毎日ベースのこのショートミーティングを実行してみる価値は十分にあります。

◆【日次情報共有会議のポイント】

- □ 昨日（ｏｒ今日）何を実行したかの報告を行うこと。
- □ 今日（ｏｒ明日）何を実行するのかの予定報告を行うこと。
- □ 今抱えている問題や困っていることを報告すること。
- □ リーダーは、各報告に対して適時のアドバイスを行うこと。
- □ リーダーは、全員に共有すべき情報を伝えること。
- □ 長時間を要する問題は当事者とリーダーで別途打ち合わせをすること。
- □ メンバーは、事前に報告事項を日報に整理・記入しておく習慣を持つこと。

☆昨日、何を実行したか **☆今日、何を実行する予定か** **☆今、抱えている問題は何か**

---

168 **時間ラグ** 時間差のこと。

169 **バッチ情報** 後でまとめて処理される情報のこと。一方、即時的な処理はリアル処理と呼ばれる。

管理

この日次情報共有会議において注意しておくべきことは次の通りです。

１．単にリーダーがメンバーの状況を把握するだけのための会議になってはいけません。メンバーたちの状況や問題点の報告に対して、適切な助言や支援を必ず実行する必要があります。

２．この情報共有会議によってプロジェクトの進行に効果が出ていることを常に確認し、効果が出ていない場合にはその問題点を解決する必要があります。

マネージャがその上の上司に状況報告をするためだけの状況確認会議になってしまえば、メンバーにとっては毎日プレッシャーを受けるだけで何の役にも立たない無駄な会議となってしまいます。

管理

◎血の通った毎日のショートミーティングは、最高のリスクヘッジ策となる。

有効なコミュニケーションのポイントのまとめ

◆ 有効なコミュニケーションのポイントをまとめると次のようになります。

- □ 恥や失敗を恐れるより、行動しない自分を恥じること。
- □ 人の好き嫌いではなく必要か否かで行うこと。
- □ 礼儀知らずの相手には、ビジネスライクな対応を。
- □ 相手に対する思いやりから始めること。
- □ 思い込みの呪縛を解くこと。
- □ 言葉だけでも、文書だけでも伝わらないということを認識しておくこと。
- □ コミュニケーションのルール・礼儀を守ること。
- □ 情報共有とは密接な双方向性のコミュニケーションのことを指す。

コミュニケーションを円滑に進めるために、上記のことを実行すると同時に、コミュニケーションすなわち意思疎通の基本は、自分の都合よりも相手の都合を先に立てるということを忘れないようにしたいものです。

管理

◎コミュニケーションの極意は、自分の都合よりも相手の都合を優先させること。

管理

直接コミュニケーションの重要性

最後に直接コミュニケーションの重要性について触れておきます。直接コミュニケーションはダイレクトコミュニケーションとかフェイス・トゥ・フェイスのコミュニケーションとかとも呼ばれていますが、要するに実際に顔と顔を突き合わせた対面コミュニケーションのことです。 それに対して、電話・電子メール・文書・ＳＮＳ・インターネットなどの媒体を通して行われるのが間接コミュニケーションです。

現代の生活は、人間どうしの直接的な接触を極力避ける傾向が非常に強くなっています。一対一の対面接触は、心理的エネルギーを大きく消耗させるため、人は直接コミュニケーションを避けがちになってしまっているものと思われます。

ＩＴ技術による間接コミュニケーションは、その情報量・拡散力・スピードにおいて非常に強い力を発揮しますが、言葉のニュアンスや雰囲気などの伝達に弱いという欠点をもっています。

たとえば、ある情報をメールで発信したからといって、それで情報共有がされるとは限りません。つたない文章ならば誤解を生みやすいでしょうし、発信した時点で受信者全員がすぐに読む保証もありません。

一方、直接コミュニケーションはものごとの伝達や決定において、単なる言葉だけではなく、その場の雰囲気や相手の表情や声のトーンから何が重要なのか、何を伝えたいのか、何が理解されていないのか、などを伝える力をもっています。 重要な問題についてのコミュニケーション、例えば顧客との要求仕様の打ち合わせ・仕様調査時の質疑応答やチーム内の重要事項の伝達・情報の共有においては、当事者同士による対面直接コミュニケーションに拠った方が、大きな解釈違いや誤解を防止するでしょう。

アジャイルソフトウェア開発宣言の原則６．は、次のように述べています。「開発チームに対してあるいは開発チーム内において最も効率的かつ効果的な情報の伝達方法はフェイス・トゥ・フェイスの会話である」。

仕事においては、直接・間接コミュニケーションそれぞれの長所・短所を考慮したコミュニケーションの使い分けが必要となります。

管理

◎重要なコミュニケーションは、対面直接コミュニケーションに拠ること。

【直接コミュニケーションｖｓ．間接コミュニケーションの特徴比較】（図５－ｈ１－３）

□ **直接コミュニケーション**
感情の伝達性高、即時性高、相互確認性高

□ **間接コミュニケーション**
情報量大・拡散力大・スピード性高

5－h 2 チームプレーの実行【人的資源マネジメント】

チームプレー問題

チームプレー[170]とは、共通の目的を持った複数の人間集団が、その目的を達成するために相互に連携した活動を行うことだと言えます。連携という言葉は、連絡提携という言葉を略したもので、連絡を密に取り合って、一つの目的のために一緒に物事をするという意味を持っています。連絡とはコミュニケーションのことで、提携とは互いに助け合い協同して事をなすことを指します。 ところで、我々のプロジェクトでは果たして密なコミュニケーションが行われているでしょうか、またお互いに助け合い、力を合わせて仕事をしているでしょうか。

プロジェクトにおけるチームプレーに関する問題には次のようなものがあります。

- 他人と関わることが億劫で問題があっても見て見ぬふりをする。
- 対人関係に弱く他人との連携ができない。
- 自分のノルマ[171]の消化で精一杯で連携する余裕がない。
- 余裕がないと他人にきつくあたってしまう。
- 自分だけが助ける役ばかりでは何の得にもならない。
- 中途半端な支援になれば余計な責任を負わされる。
- 他人に恩を着せられるので支援などして欲しくない。

上記のような状態に追い込まれる原因としては次のようなことが考えられます。

① 仕事が忙し過ぎて時間的な余裕がない。 ② 他人との信頼関係がない。

①参照： ３－１．早期の仕様凍結、３－４．顧客要求の重要度順位の設定、
３－５．見積り●見積りにおけるリスク、３－６．スケジュールの作成、
３－７．プロジェクト目標の設定●納期・生産性目標値の設定、
４－ｈ１．獲得時間の最大化と失う時間の最小化
②参照： ５－ｈ１．コミュニケーションの活性化

管理

◎チームプレーとは、密接なコミュニケーションに基づき、
互いに助け合い協同して仕事をすること。

---

170 **チームプレー** 共通の目的を持った複数の人間集団が、その目的を達成するために相互に連携した活動を行うこと。

171 **ノルマ** 半強制的に与えられた時間制限付の労働の基準量。

管理

チームプレーの精神

チームプレーの精神は、ラグビーの代表的な標語である「All for one、One for all」（みんなは一人のために、一人はみんなのために）の言葉によく表れています。

チームプレーを阻害する最大の要因は、個人における“過剰”な自己防衛にあります。自己防衛は人間の生存本能の基本ですが、問題になるのはそれが“過剰”に発揮された場合です。他人へのたった１０分の説明を惜しみ、ちょっとしたノウハウの伝授さえ自分の損だと感じるような心持ちは一体どこからきたのでしょうか。このような貧しい心持ちのことは、一般的に「せこい」と言われています。この “せこさ”が、自分自身も含めて他人や組織を衰弱させることにつながる、と言うことに早く気づく必要があります。

自分が持っているものは、もともと親や先祖や誰か他人が譲ってくれたものであり、自分の譲りの大きさに比例したものがいつか必ず自分の手元に戻ってくるのです。そのことが信じられない人たちは過剰な自己中心的な行動を改められず、自己衰退への道を突き進むことになるでしょう。

プロジェクトにおけるチームプレー精神の発揮とは次のようなことを指します。

- 自分に可能な範囲から助け合いの行動を始めること。
- 行動できないことを自分の性格のせいにしないこと。
- まずは自分の意思で一歩を踏み出すこと。
- リーダーは、自分のノルマを超えた領域に踏み出すこと。
- まず自分から先に手を差し伸べること。
- 誰の責任かではなく、私にやらせてくださいの心で。
- 万策尽きる前に他人に支援を頼むこと。
- 失敗を個人の責任に帰さず、チームの力と知恵を結集して解決に取り組むこと。
- 余裕を生み出すあらゆる方策をチームとして実行すること。

このような考え方や行動は、かつて日本の伝統的な行動規範[172]であり義侠心[173]とかサムライスピリットとかと言われてきたもので、少なからず現代の日本においても存在しているものと信じています。

管理

◎チームプレーの精神は、
「All for one、One for all」（みんなは一人のために、一人はみんなのために）にある。

---

172 **行動規範** 人間社会集団におけるルールや慣習のこと。

173 **義侠心** 正義を重んじて、強い者をくじき、弱い者を助けるような心もちのこと。

チームプレーとは、そもそも何か

個人戦ではなく組織戦を

わたしたちのプロジェクト活動は、個人戦ではなく典型的な組織戦です。個人戦の勝った負けたにこだわらず、仲間との連携による組織戦に持ち込むことで、チームの５人力は１０人力ともなります。

先の大戦において、個人技に優れていたゼロ戦に対して、米軍がとった２対１の戦法いわゆるサッチ・ウィーブ戦法は、優秀なゼロ戦一機に対して二機のグラマンによる組織戦で対抗してきました。グラマンの性能向上と相まって二機による組織戦によって、ゼロ戦は連戦連敗に追い込まれたと言われています。極限の組織的戦闘である空中戦においても、味方同士の相互援助が勝利に不可欠なものだということが証明されているのです。

自分だけが良ければ他人のことはどうでも良いというような考え方では、仕事でも戦闘でも決して勝利を得ることはできません。「情けは人の為ならず」と言われている通り、助け・助けられの関係が自他を共に助け、組織を強化することになるのです。

次の一文は、チームプレーのことを端的に表しています。

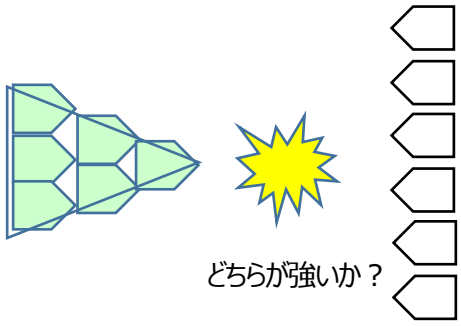
「例えば、ラグビーにおいては各々のポジションの役割は明確に決められているが、いったん体制が崩れた場合には、各メンバーはその役割を超えて臨機応変な対応が求められる。メンバーは自分の役割と組織を補い合う自律性が要求されると言われている。仲間が守るべきポジションが崩れた時には、巨漢のフォワード[174]もバックスに参加し、体の小さいスクラムハーフ[175]も、相手の巨漢に果敢にタックルを仕掛ける。」

（ＳＱＵＳＥ通信Ｖｏｌ80、チームプレーの魅力）

管理

◎チームプレーとは、個人戦ではなく組織戦で戦うこと。

【組織戦と個人戦、どちらが強いか】（図５－ｈ２－１）



174 **フォワード** ラグビーにおいてスクラムの最前列を守る巨漢・屈強な選手。

175 **スクラムハーフ** フォワードとバックスのつなぎ役で、俊敏性と判断力が求められ、小柄な選手でも活躍できる。

管理

チームプレーがもたらすもの

チームプレーすなわち連携プレーは間違いなく次のような効果をもたらします。

【チームプレーの効果】

- チーム内のミスをお互いに救い合い、学び合う。
- 連携プレーによる学習効果は人の成長を早める。
- 信頼関係を確かなものにする。
- モチベーションを高め、プロジェクトに前進する力を与える。
- 個々人の責任範囲を超えた活動を促し、プロジェクトの統合機能を高める

管理

◎チームプレーは、人を成長させ、信頼関係を高め、プロジェクトを前進させる。

チームプレーを可能にするリーダーシップ

チームプレーを可能にするものとして、リーダーシップを欠かすことはできません。個人戦を回避し、組織戦に持ち込むためには、リーダーのスキルが重要になってきます。

プロジェクトはある意味で軍隊と共通の機能を持っています。軍の目的は戦いに勝利することであり、プロジェクトの目的は開発を成功させることです。

軍隊が勝利する条件は次の二つと言われています。 ①目的・目標の単一化 ②兵力の集中

かの著名な『失敗の本質』には次のように記述されています。

「目的の単一化とそれに対する兵力の集中は作戦の基本であり、反対に目的が複数あり、そのため兵力が分散されるような状況は、それ自体で敗戦の条件になる。目的と手段は正しく適合していなければならない。」（『失敗の本質　日本軍の組織論的研究』）

同様に目標の設定は、プロジェクト活動の最初の最も重要な仕事です。達成すべきあるいは達成したい目標が明確でなければ目標は達成できません。明確な目標はチームの力を結集させ、無駄な活動を防止し、プロジェクトの効率的な活動の原点となります。

すなわちリーダーシップとは、リーダーがプロジェクトの目標を明確にし、その達成に向けて開発メンバーの知恵と力を結集するということに他なりません。

管理

◎リーダーシップは、プロジェクトの目標を明確にし、その達成に向けて
開発メンバーの知恵と力を結集することで、チームプレーを可能にする。

本書の全ての【Job Activity】において記述された内容は、リーダーの実務面における役割そのものですが、リーダーにおいて人間力として発揮される【Human Activity】におけるリーダーシップの具体的な内容は以下の通りです。

【リーダーの心得】

１．先頭に立って困難な問題に立ち向かうこと

チームが良い方向に向かう、あるいは悪い方向に向かうのは、メンバーたちの問題というよりもむしろ、これらを指導する側の誠実さ、あるいは不誠実さによるものだと考え、リーダーは自らメンバーに先立って仕事にいそしみ、思いやりのある行動をとり、チーム全体が安心してその道を進めるようにすること。

２．説得よりも納得を得ること

業務態度が悪いメンバーに対しては、丁寧にその弊害について説明し、改めることができるよう心から接しなければならない。説得できないと言って放置してはならず、何度も何度も繰り返し教え、納得を得るようにすること。

３．独断専行に走らないこと

色々なことでメンバーたちが苦情を言ってくることがあるが、一方の言い分だけを聞いて処理をすると事実誤認によって過ちをおかすこともあり、メンバーの信頼を失うことになってしまう。何事によらず両者の言い分を良く聞いて判断をする必要がある。

４．メンバーの公平・公正な扱い方をすること

メンバーに対するにあたって、気安く頻繁に接して来る者には自然と愛着がわいてき、疎遠な者には愛着もわかないのが人情だが、知らず知らずの間に日常の取扱い方がえこひいきになってしまいやすいものです。このことは十分注意をしなければならず、えこひいきと言われないようにすること。

５．後ろ指を指されるようなことをしないこと

リーダーは、自らの行いの正しさ・潔白を守り、誠意をもって身を慎み、メンバーたちからあざけりを受けないように心掛けること。

６．メンバーの不出来はリーダーの不出来に拠る

業務活動に対する違反行為やメンバーたちのやる気の増減は、すべてリーダーの責任であり、メンバーのできの悪さはメンバーのやり方によるものではなく、リーダーのやり方が至らないからだと思い、リーダーは日々節度のある行動をとり我欲を抑え、万事まごころをもってめんどうをみること。

７．率先垂範を行うこと

現場のリーダーの立場にあるものは、メンバーよりも前に現場に出て皆に指図し、終業時には明日の手配の指図をすませ、皆よりも遅く引き揚げるようにすること。

（参考資料：岩崎敏夫著「二宮尊徳仕法の研究」、二宮尊徳著「御仕法掛心得方大略」）

これがすべてできる神様のようなリーダーはいませんが、一つや二つくらいはできるようになりたいものです。

メンバーたちは常にリーダーの発言や行動に注目しており、リーダーの考え方や立ち居振る舞いを真似ようとするものです。

管理

チームプレーを促進する改善活動

仕事をうまく運ぶためには、時には自分たちの仕事を一歩離れた視点から見てみる必要があります。一歩離れて見るということは、冷静さを取り戻し、客観的に自分たちの仕事を評価してみるということです。仕事に追いまくられるだけの毎日では、自分たちの仕事の良し悪しを客観的に評価することは困難です。

改善活動は、冷静な目および合理的な客観性を身につけるのと同時に、チームプレーの訓練に最も適した活動だと言えます。

改善活動については「４－６．改善活動の実行」において述べた通りですが、もう一度その基本コンセプトを見てみることにします。

【改善活動の基本コンセプト】

① 個人戦ではなく組織戦（チームプレー）を。
② 一人だけが成長するのではなく、みな共に成長すること。
③ 能力に長けたものは、後進の者にその知恵（ノウハウ）を譲ること。
④ 後進の者もその成長に従って、さらに後に続く後進の者にその知恵（ノウハウ）を譲ること。
⑤ 先に進んでいる者は、その持てるあらゆる価値あるもの、資産・資金・知恵（ノウハウ）の全てを後進のものに順に譲り渡し、永続的な繁栄の循環を実現させること。

これらの基本コンセプトは、チームプレーを促進するコンセプトであるとも言えます。改善活動はＱＣＤの改善のみならず、その活動を通してメンバーの人的育成を図り、見事なチームプレーを実現することでしょう。

管理

◎改善活動は、ＱＣＤの改善・人材の育成・チームプレーの実現の場となる。

管理

相互扶助の実行

チームプレーは、何も開発者たちのチーム内に限ったことではありません。そのプロジェクトに関係する全てのステークホルダーたちも広義の意味で一つのプロジェクトチームだと言う認識を持つ必要があります。チームプレーはプロジェクトの中で実行される人間相互における連携プレーです。

それらの連携プレーにおいての重要なキーワードとして、相互扶助[176]と相互義務[177]を挙げることができます。

ＩＴプロジェクトにおける相互扶助欠落の具体例を次に示します。

【相互扶助欠落の具体例】

① 社員の悩みに耳を貸そうともしない上司や同僚
② 過重負荷に陥った部下を助けようともしない上司や同僚
③ 病気になった原因を社員の弱さに帰して平気で見捨てる上司
④ 不条理・不道徳な行為を平気で行う厚顔無恥な上司
⑤ 不条理・不道徳な行為を見て見ぬふりをする同僚
⑥ 緊急な救助・援助の求めを無視する人々

このような状況は個人の孤立分断化を進め、組織を無力化させてしまいます。

相互扶助の実行

相互扶助については先に述べた通り、お互いの不足部分をお互いに補うことで、各自の不完全な部分をチームとして補う行動です。ただしこの相互扶助の行動は義務として課されたものではなく、実行されなくても直接責任を問われることはありませんが、相互扶助の足りないチームの力は非常に弱く、多くの欠陥や失敗を防ぐことができません。

相互扶助はメンバーの間だけで実行されるものではなく、上司と部下の間、顧客との間、元請けとの間などさまざまな関係者の間で実行されることにより、人の間および組織の間の欠落部分を補いプロジェクトの統合管理の一翼を担う重要な働きをするものです。

相互扶助的な行動は援護射撃とも表現され、自分が危機に陥った時に、どこの誰が撃ったのか分からないが確実に自分を救ったその一撃に対する温情は、一生涯忘れることができないでしょう。

相互扶助は義務ではありませんが、人に必要とされる人材になりたいのならば、必ず実行されなければならない心の義務だと言えます。

管理

◎相互扶助は、自他を共に生かすための心の義務。

---

176 **相互扶助** プロジェクトの関係者がお互いの不足を補い助け合うこと。

177 **相互義務** プロジェクトの関係者がお互いに果たすべき義務のこと。

相互義務の履行

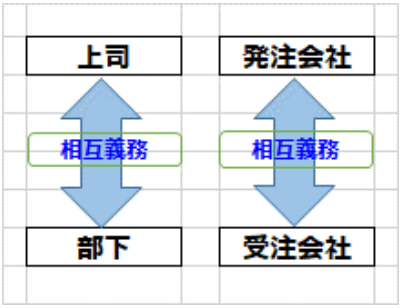
相互義務ということについて、多くの人はあまり意識してはいません。特に仕事における義務については、指示される側の一方的な履行義務としてしか理解されていない場合がほとんどです。利益だけが価値観のすべてであるかのような世の中においては、指示する側の義務はお金を払うことだけであるかのような歪んだ考え方が支配的になっています。お金を払う対価として労働の義務があると言う考え方は短絡的な合理性に傾いた考え方であり、大多数の人間において納得感を得ることはできません。それは人や集団を動かす、もう一つの原理である道徳的な妥当性の視点が欠落しているからです。

妥当性とは、その社会における人間感情を律する経験則であり、適正な行動は理と情のバランスがとれていなければ生まれてきません。

業務を指示する立場の者たちにおける道徳性の発揮とは、仕事を受ける立場のものがその業務を遂行しやすいような心理的および物的環境を提供するということで、例えば次のようなことになります。

① 仕事内容について、その意味や背景も含め、過不足のない説明をドキュメントに基づいて行うこと。
② 仕事を依頼するにあたって、強圧的態度によらず礼を尽すこと。
③ 仕事の遂行に必要かつ妥当な人材・物資・資金・時間などを過不足なく提供すること。

【相互義務の履行】（図５－ｈ２－２）



次に相互義務不履行の例として以下のようなものがあります。

【相互義務不履行の例】

① 上司から部下に対する不条理かつ一方的な業務命令
② 自分の責任を部下に押し付ける上司
③ 失敗の責任を取らないマネジメント
④ 妥当性および合理性に欠けたコストカットや納期短縮の要求
⑤ 請負契約ないしは業務終了後の値引き要求
⑥ 発注者における指示・説明責任を放棄した下請けに対する丸投げ的発注
⑦ 妥当性を超えた中間マージン中抜きの多重下請け発注

管理

これらの相互義務の不履行は、妥当性のみならず合理性にも欠けた恥ずべき行為であり、伝統的な日本人の行動規範を大きく逸脱したものであり、その実態が社会の明るみに出された場合は、法的あるいは社会的な制裁の対象となっています。多くの場合、弱者の立場にある者たちは、経済的な報復を恐れて抗議すらできない状況に置かれています。

相互義務の履行とは、会社間・組織間・個人間において取り交わされた約束に従って、各々が約束した義務を実行することです。しかしながらＩＴ開発関係者においては、相互義務の履行という認識は非常にあいまいなままになっています。

例えば要件定義書をまとめるべき者が、約束の期日までに提供しないことや、契約書の内容以外の業務を無理やり強要することや、契約済の開発費や開発期間の削減・短縮要求など、さまざまな相互義務違反の行為が横行しています。またこれらの契約違反行為の他にも、自分で行うべき仕事を他人に丸投げするような行為も少なからずあります。相互義務違反は多くの場合、優位の者や組織から従属の者および組織に対して行われています。

これらの違反行為が無理やり行われた結果は、障害の多発による品質の低下であり、スケジュールの遅延であり、手抜き仕事であり、赤字プロジェクトであり、その被害はブーメランのように違反行為を押しつけたものたちにも戻ってきているのです。

集団を維持・発展させる重要な要素は、相互義務の履行と相互扶助の実行の２つであり、すなわち相互に果たすべき義務を果たし、相互の弱点を補い助け合うということです。個人および組織が生き残り、更なる発展をするための原点が相互義務の履行と相互扶助を基盤としたチームプレーにあります。

管理

◎相互義務は、履行されなければならない義務。

相互義務の不履行は自他を共に滅ぼす。

管理

【管理プロセスのまとめ】

管理プロセスの役割は、開発状況のモニタリングおよびその正常化のためのコントロールにあります。本章では、開発の基本的なベースラインである顧客との約束に基づいた要求仕様の適切な変更管理およびその要求仕様に基づいた開発成功の三条件である品質・コスト・スケジュールに対するモニタリングおよび正常化コントロールについて述べてきました。

これらの活動を整理すると以下のようになります。

【Job Activity】

要求仕様の変更管理

管理

◎ベースライン（基線）とは、
プロジェクトマネジメントにおいては、顧客と開発側で合意した成果物の範囲および基準の事。両者間で約束した仕様とそのＱＣＤに関する基準は重要なベースラインとなる。

管理

◎ベースラインの確定・変更管理の不在がもたらすリスク
☆プロジェクトの失敗：ＱＣＤの大幅な未達
☆顧客ビジネスの停止：市場障害の多発

管理

◎ベースライン確定後の仕様変更・追加分は別途見積りとする交渉が必要。

品質管理

管理

◎製品品質の代表的な三つの指標、
単体テスト･結合テスト･総合テストの不具合密度（件数／kstep）

管理

◎代表的な品質管理表
品質状況管理表、不具合習熟曲線表

管理

◎効果的なレビューのポイント
事前の準備、事前のセルフチェック、二段階レビュー、影響度の確認、過去の失敗

管理

◎要求仕様書のチェックポイント
１．妥当であること　２．あいまいでないこと　３．完全であること　４．矛盾がないこと
５．重要度と安定度のランク付けがされていること　６．検証可能であること
７．変更が容易であること　８．追跡可能であること

管理

◎ドキュメントの条件
要求仕様書･基本設計書･詳細設計書･テスト設計書は共通なＷＢＳ構造を持つこと。

管理

◎外注の品質管理
外注に委託した開発作業の品質基準は、社内基準と同等のものを適用すること。

コスト･利益管理

管理

◎コスト･利益管理の管理は、最初に粗利率の設定から始まる。

管理

◎原価管理表の役割

① 予算と実績の差異とその理由を明確することで、
② 不適切な開発行為を是正し、
③ 目標原価を達成すること。

進捗管理

管理

◎進捗管理の三点セット

1．プロダクト成果物進捗管理（モノの進捗管理）
2．予算消費進捗管理（カネの進捗管理）
3．残業時間管理（ヒトの消耗度管理）

管理

◎モノの進捗管理は、単品管理の思想で行うこと。

管理

◎開発費の消費状況からプロジェクトの健全性を見るためには、工程別予算進捗管理が必要になる。

管理

【Human Activity】

コミュニケーション

管理
◎プロジェクト問題の原因№１は、コミュニケーション不良にある。

管理
◎コミュニケーションは、要件定義、見積りおよび全ての開発行為に大きな影響を及ぼす。

管理
◎自由闊達なコミュニケーションは、
他人に意識を向けるよりも仕事に意識を向けることから始まる。

管理
◎伝わるコミュニケーションは、数値やデータに基づくことによる。

管理
◎コミュニケーションにおいては、
先に相手が知りたいことを伝え自分の知りたいことはその後で。

管理
◎無駄のない会議は、確実にプロジェクトの生産性を向上させる。

管理
◎血の通った毎日のショートミーティングは、最高のリスクヘッジ策となる。

管理
◎コミュニケーションの極意は、自分の都合よりも相手の都合を優先させること。

管理
◎重要なコミュニケーションは、対面直接コミュニケーションに拠ること。

管理

チームプレー

管理

◎チームプレーとは、密接なコミュニケーションに基づき、
互いに助け合い協同して仕事をすること。

管理

◎チームプレーの精神は、「All for one,One for all」にある。

管理

◎チームプレーとは、個人戦ではなく組織戦で戦うこと。

管理

◎チームプレーは、人を成長させ、信頼関係を高め、プロジェクトを前進させる。

管理

◎リーダーシップは、プロジェクトの目標を明確にし、その達成に向けて
開発メンバーの知恵と力を結集することで、チームプレーを可能にする。

管理

◎改善活動は、ＱＣＤの改善･人材の育成･チームプレーの実現の場となる。

相互扶助と相互義務

管理

◎相互扶助は、自他を共に生かすための心の義務。

管理

◎相互義務は、履行されなければならない義務。
相互義務の不履行は自他を共に滅ぼす。

第６章

振り返りのプロセス

【統合マネジメント】

概要

第６章　振り返りのプロセス

プロジェクトのＱＣＤ結果の振り返りおよび総括の仕方についての解説です。

- 失敗の振り返りは、未来の失敗を予防する（６－１．）
- 失敗から読み取るべきは、未来への希望のメッセージ（６－１．）
- 工程ごとの振り返りは、連続する変化に対する柔軟な対応として機能する（６－１．）
- 振り返り会議は、犯人捜しの場ではなく情報共有の場とする（６－１．）
- 全ての管理データの最終結果をプロジェクト完了報告書として残す（６－２．）
- 失敗の原因を他者や物に求める人のリスクは高くなる（６－ｈ１．）
- やる気・モチベーションの喚起は、時間不足の解消によってもたらされる（６－ｈ２．）

【Job Activity】【統合マネジメント】

6－1 プロジェクト活動の振り返り

何のために振り返りが必要なのか

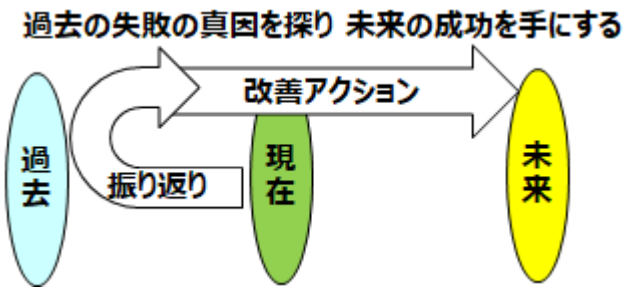
自分たちの行動や考え方を振り返ってみて、それが現実の要求にどう合わなくて、どういうふうに問題がおきたのかを確認し、将来同じ問題を引き起こさないようにするためには、今後どのような準備や行動をすればよいのかを発見する必要があります。

すなわち振り返りとは、過去の自分たちの不適切・非効率な考え方や行動パターンをはっきりと自覚し、それを改善する効果的な方法を考え、それを実行に移すところまでを指します。

過去の失敗には必ず未来の私たちに託された有益なメッセージが残されています。このメッセージを解読するためには、失敗の表面的な原因だけではなく、本当の真因を明らかにする必要があります。

振り返りのイメージを図で表すと次のようになります。

【振返りのイメージ】（図６－１－１）



プロジェクトにおける振り返りということは、個人だけではなく組織全体として、過去の失敗に学ぶということです。組織として失敗に学ばない限り、プロジェクトという組織戦で勝利することはできません。

振返

◎振り返りとは、過去の自分たちの不適切・非効率な考え方や行動パターンをはっきりと自覚し、それを改善する効果的な方法を考え、それを実行に移すこと。

なぜ振り返りができないのか

多くの組織においては、プロジェクト終了時に必ずその総括および反省を実行するように取り決められていると思いますが、実際にそれを実行しているプロジェクトは少ないようです。会社の規定でプロジェクト完了報告書の提出が義務づけられていたとしても、形だけで中身の薄い報告書も少なからずあります。

本当の振り返りが行われない理由は次のようです。

- 時間がない。めんどうくさい。
- 失敗の証拠を他人に見られたくない。
- 過去の失敗を思い出すことは苦痛で早く忘れたい。

素人か子供の言い訳にしか見えず、とてもプロの言葉だとは思えません。本当の振り返りを行わないまま何度も失敗を繰り返していると、いつのまにか負け癖がついてしまい、このようなネガティブ思考の罠にはまってしまいます。これは過去の失敗からみじめなメッセージしか読み取れていない結果が招いたもので、本当に読み取るべきなのは希望のメッセージなのです。失敗の本質が分かれば、二度と同じ失敗をする確率は激減することでしょう。

過去の失敗に学ぶべきことは、同じ失敗は繰り返さないということであり、すなわち成功の確率を高めるということに他なりません。みじめな傷をなめるのではなく、成功という希望に向って進むということです。

このことが本心で理解できれば、意味のある振り返り・反省・学習・総括を実行しなければどれだけ損をしているのかが理解できるでしょう。

ポジティブ思考な言葉は次のようになるでしょう。

- 時間がないなら時間を生み出す工夫や活動をする。
- めんどうでも後始末はちゃんとやり、けじめをつける。
- もう自転車操業[178]はやめにして、失敗で失う膨大な時間を減して楽になろう。
- 新たな改善を成功させ失敗の借りを返す。
- 毎日振り返りを行う。全ての工程で振り返りを行う。

振返

◎過去の失敗から読み取るべきは惨めなメッセージではなく
希望のメッセージを。

178 **自転車操業** 自転車はこぐのを止めれば倒れるのと同じで、効果がないことをいつまでも続けざるを得ないような状況のこと。

定期的な振り返りについて

振り返りは、開発終了時に実行されるプロジェクト完了会議（ラップアップミーティング）だけでは全く不十分でしょう。開発の諸条件や状況は常に流動的であり、この変化を見逃してしまうと、プロジェクトが窮地に立たされる場合もあります。

重要な開発条件や状況の変化を見逃さないためには、顧客や営業・ＳＥチームを初めとしたステークホルダーとの定期的なコミュニケーションの実行、および開発チーム内においては開発の各工程のレビューにおいての振り返りの実行が必要になります。

振返

◎振り返りは変化対応の出発点となる。

① 主要なステークホルダーたちとの定期的なコミュニケーションの実施。

② チームにおける定期的な振り返りの実行。

工程ごとの振り返りは、失敗の予防だけではなく、リスク排除および課題解決としても機能します。

定期的な振り返りにおいては、第５章「管理のプロセス」において紹介した全ての管理帳票類が示すデータによる振り返りが必要です。

振返

◎振り返りのポイントは、下記の五つ。

① 仕様変更管理

② 品質管理

③ コスト・利益管理

④ 進捗管理（モノ・カネ・ヒト）

⑤ 開発・評価工程および客先稼働等における重度障害対応

工程毎の振り返りは、一人の担当者が複数の物件を抱えていた場合においても、その開発規模の大きさにかかわらず実行が必要です。

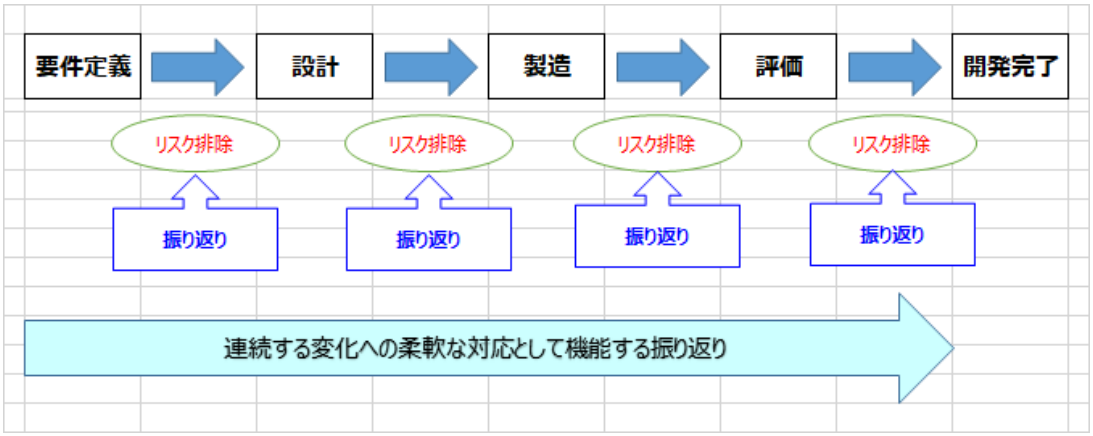
短納期に追われている開発者においては、工程ごとの振り返りのみならず工程ごとのレビューすら実行されない場合も珍しくなく、いくら値引き要求が厳しいからといっても、そのような状況を招くような、必要工数や期間をカットするような見積り回答をすべきではありません。

値引き要求に対しては改善活動などを通して得られた、生産性向上による利益の一部を充てるのが本来の正しい対応法だと言えます。

振り返り

工程ごとに実行される振り返りによってリスクが見えてくるようになり、そのリスクの解消行動によって次の工程における重大な問題の発生を予防することができます。この定期的な振り返りのイメージを次に示します。

【工程ごとの振り返りによるリスク排除】（図６－１－２）



アジャイルソフトウェア開発宣言の原則１２．においては、振り返りによる変化対応について、次のように述べています。

「チームは定期的に、より効果的な方法につき振り返りを行い、それに従ってその行動を変更し調節していく。」

振返

◎工程ごとの振り返りは、失敗予防・リスク排除・課題解決として機能する。

振り返り会議の進め方

振り返り会議の議事進行は下記の要領で進めるとよいでしょう。

（１）議長はプロマネが勤め、書記はサブリーダクラスから選ぶ。
（２）議事内容は下記についての問題点と対処状況の確認。
　① 要件定義の変更管理の結果
　② 品質管理の結果
　③ コスト・利益管理の結果
　④ 進捗管理（モノ・カネ・ヒト）の結果
（３）議事進行上の注意点
　① プロマネが議事をリードすること。
　② 犯人捜しをしないこと
　③ 真剣ではあるが穏やかな雰囲気を保つこと。
　④ 討議内容は全員に見えるよう白板に書き留め、ＩＴ掲示板等に記載し情報共有を図る。
　⑤ アクションの決定においては、データ・数値等の証拠に基づいて誰が・何を・どの様に・いつまでにという５Ｗ４Ｈ[179]を明確にしておくこと。

振り返り会議の目的は、不具合を発生させた犯人を捜すことではなく、不具合の原因および真因を特定し有効な対策を発見することです。誰のせいだとか誰が悪いとかの発言は避けるべきです。

振返

◎振り返り会議は、犯人捜しではなく原因・対策の情報共有の場にすること。

[179] **５Ｗ４Ｈ**「５Ｗ：Why、What、Who、When、Where　４Ｈ：How to、How many、How much、How long」の略で、５つのＷで始まる疑問詞は「誰が・いつ・どこで・何故・何を」したのか、つまり行為の当事者・時間・場所・動機（理由）・対象物を明確にすることでその行為の意味を明らかにする。また４つのＨは、その行為を実現するための手段・方法・程度などの科学的データを明らかにする。これらは人の行為の全体像を特定する良い方法である。

6－2 プロジェクト完了報告書

プロジェクト完了時に実行される振り返り会議（ラップアップ・ミーティング）においては、仕様変更管理、品質管理、コスト管理、進捗管理それぞれの目標と実績の差の理由および今後の課題を総括し、次に続く開発への申し送り情報としてプロジェクト完了報告書にまとめます。

プロジェクト完了報告書に記載される主な内容を下記に示します。

仕様変更管理の考察　（表６－２―１）

仕様変更管理の考察														
No.	管理番号	発生日	対応期限（ベースライン）	対応日	対応状況	業務名	変更内容	変更区分	修正担当	影響度範囲	難易度	リリースバージョン	対応工数（見込）	対応工数（実績）
1	PJ-001	xx.09.31	xx.10.31								A			
2											B			
合計														
考察	1．対応工数の見込みと実績の差異の理由および今後の改善課題 2．仕様変更管理全般についての反省													

顧客と約束した要求仕様のベースラインが確定した後の仕様変更は極力避けたいものですが、現実問題としては変更要求が発生してくることが少なからずあります。プロジェクトを終了するにあたって、仕様のベースライン確定後にどれくらいの仕様変更量があったのかを定量的に把握し、あまりにも大きい場合は次の開発においての顧客との仕様検討のやり方を改善する必要があります。

拡大表は「付録図表１７．プロジェクト完了報告書●仕様変更管理の考察」を参照のこと。

振り返り

品質管理の考察（表６－２－２）

品質管理の考察													
テストフェーズ			不具合原因内訳				実績値		目標値	不具合深刻度レベル			
工程	プログラムステップ数	テスト項目数	仕様バグ	設計バグ	製造バグ	管理ミス	不具合件数	バグ密度（件数/kstep）	バグ密度（件数/kstep）	A（重度）	B（中度）	C（軽度）	テスト工数
単体テスト													
結合テスト													
総合テスト													
市場障害													
合計													
考察	１．バグ密度の実績値と目標値の差異の理由および今後の改善課題 ２．不具合原因内訳・不具合深刻度レベルの結果についての反省 ３．品質管理全般についての反省												

品質管理においては、プロジェクトが目標としたバグ密度と実績の差異を把握し、目標未達の場合は何が不足していたのかを考察し、新たな改善施策を立て、次の開発において実行する必要があります。また不具合原因の内訳や不具合の深刻度レベルを分析し、チームの弱点をしっかりと認識し、これらについても改善の実行が必要になります。

拡大表は「付録図表１７．プロジェクト完了報告書●品質管理の考察」を参照のこと。

コスト管理の考察（表６－２－３）

コスト管理の考察		予算	実績	差異	考察
直接材料費①	外注費				１．予算超過・消費不足の理由 ２．今後の改善課題 ３．原価管理全般についての反省
	購入品等				
	小計				
直接労務費②	開発人件費				
	経費				
	小計				
間接費③	開発間接費				
原価合計 ①＋②＋③					

原価については、直接材料費の外注費、直接労務費の開発人件費が原価のほとんどを占めていますので、外注費および社内の開発人件費に着目する必要があります。

目標原価が未達成の場合、その多くの原因は品質問題・進捗問題に起因している場合が多く、外注費と社内開発人件費について仕様変更管理の考察、品質管理の考察および進捗管理の考察結果も参考にして、その原因を特定し改善施策を実行する必要があります。

拡大表は「付録図表１７．プロジェクト完了報告書●コスト管理の考察」を参照のこと。

振り返り

進捗管理の考察（表６－２－４）

進捗管理の考察	各工程における進捗の目標と実績の差異の理由およびに今後の改善課題
要求定義	
概要設計	
詳細設計	
コーディング	
テスト	
進捗全般に関する反省	

スケジュールはたとえ期日が形の上で守られたとしても、品質目標が達成されていなければ何の意味もなく、スケジュールが守られたとは言えません。

進捗管理の考察においては各工程における進捗遅延の本当の原因についての考察が必要であり、その改善対策を前記の３項目の考察から導かれた改善対策と合わせて次の開発に生かす必要があります。

振返

◎プロジェクト完了報告書は、そのプロジェクトのあるがままの結果を示すと同時に、
次に続くプロジェクトに活きる目標を提供する必要がある。

6 – h 1 失敗の原因を他に求めないこと

できにくい反省

失敗の結果に対する人の反応として３種類があります。1 つめは何も悔やまず平気な人。２つめは悔やむだけの人。３つめは悔やんだ後で、同じ失敗をしないために失敗の原因を探し対策を考える人。

失敗しても自分のせいにはしたくないと思えば真剣な反省もできません。反省したり謝罪したりすることは本当に難しいことです。言い訳から始まる謝罪は顧客や相手の怒りを倍増させます。

失敗の原因を追究しない限り、その失敗を生み出している状況や現実の変化を読み取ることはできません。現実直視力や柔軟性は失敗に学ぶことによって養成されていきます。

自分の失敗責任は最小化し、他人の失敗責任を最大化するような人は、自己の成長もリスクの回避もできないでしょう。失敗の現実を直視できない人のリスクは高くなります。失敗の原因を他者や物に求める人のリスクは更に高くなります。

振り返り

レビューが振り返り行為の一つだという認識のない人においては、忙しいという理由で、しばしばレビューがスキップされてしまうことが往々にしてあります。

レビューも振り返り会議も実行されなければ、失敗の経験に学ぶ機会が全くないということになります。そのようなプロジェクトは規模の大小を問わず、手抜きプロジェクトなのです。日常的にこのようなやり方を続けていれば品質も利益も悪化し、人材も全く成長できません。このような学習能力[180]のない組織に所属していたとしても、自分が気づいた時点でプロジェクトリーダーやマネージャに問題提起をする必要があります。開発の規模にかかわらず、すべてのプロジェクトにおいて、自己レビューや社内レビューおよび振り返り会議等を必ず実行する必要があります。

「振り返り」を通した「改善活動」を日常的に実行することで、技術者個人のスキル向上や組織力の向上が確実に実現され、バグに追い回されることもなく、確実な利益を確保できる開発組織を生み出すことができるでしょう。

---

180 **学習能力** 失敗に学ぶ能力ないしは成長したいと思う意欲のこと。

ラップアップ（開発完了会議）

ラップアップはプロジェクトを振り返る重要な最後のけじめであると同時に、次なる仕事へのノウハウの継承という重要な役割を担う行為であるともいえます。この仕事はプロマネの重要な責務の一つであり、メンバーの自主性に任せておくような性質のものではありません。プロマネ主導で必ず実行する必要があります。またラップアップは、業務遂行中に採取し分析を済ませておいたＱＣＤの指標数値に基づいた振り返りの実行、および次に続くプロジェクトへの申し渡し事項をまとめておく必要があります。単に、○○の失敗をしないように“がんばろー”と言うような情緒的な会議にすることだけは避けたいものです。

振返

◎振り返りは、失敗の経験に学び個人・組織ともに成長する有効な手段であり、
プロジェクトのけじめであると同時にノウハウ継承の役割をもっている。

振り返り

6－h 2　やる気を起こす～モチベーションの喚起

社員のモチベーションを上げるために多くの会社は何らかの褒章制度を持っていますが、それは社員たちのモチベーションの向上には余り役に立ってはいないようです。褒章をもらいたいために頑張る人などほとんどいないと思われます。また人は自分がすでに持っているようなものを褒美としてもらったとしても、大して喜びもしないでしょう。現代の世の中は物であふれており、余程貴重で高価なもの以外はみんなが既に持っているものばかりです。多くの人は物に飢えている訳ではなく、心を充足させてくれる事に飢えているのだと思われます。

人々が最も心的に価値をおいているものは、将来に対する安心安全や希望でしょう。組織の大小を問わずリーダーの重要な役割の一つは、人々が希望を持てるような施策を実行することです。この先良いことが起こりそうだという実感は間違いなく人にやる気を起こさせます。
人々のやる気によって良い仕事が行われ、良い製品が生み出され、同時に利益も生み出されます。

現代のＩＴプロジェクトにおいて最も不足しているものは「時間」であることに間違いはないでしょう。多くの開発者を最も苦しめているのは時間不足です。妥当な開発を実行するために必要な時間が全く不足しているのです。

開発者たちの時間を確保するために必要なことは次の二つだけです。
１．獲得する時間の最大化
２．失う時間の最小化
詳細は第４章「４－ｈ１．獲得する時間の最大化と失う時間の最小化」をご参照ください。

プロマネにおいては、この二つを満足させる施策を実行することで、深夜残業や休日出勤を撲滅し、開発者たちの心からのモチベーションを喚起することができるでしょう。

振返

◎やる気やモチベーションの喚起は、
開発者の時間不足の解消によってもたらされる。

【振り返りプロセスのまとめ】

振り返りのプロセスは、より良い明日を迎えるためには必須のものです。本章では、振り返りの必要性、振り返りの実行を妨げているもの、定期的な振り返りの効果、振り返り会議の進め方、プロジェクト完了報告書の内容について順に述べてきました。

これらの内容を整理すると以下のようになります。

【Job Activity】

振返

【振り返りの意味】

◎振り返りとは、過去の自分たちの不適切・非効率な考え方や行動パターンをはっきりと自覚し、それを改善する効果的な方法を考え、それを実行に移すこと。

振返

【振り返りのメッセージ】

◎過去の失敗から読み取るべきは惨めなメッセージではなく、希望のメッセージを。

振返

【振り返りのポイント】

◎振り返りのポイントは、下記の五つ。

① 仕様変更管理
② 品質管理
③ コスト・利益管理
④ 進捗管理（モノ・カネ・ヒト）
⑤ 開発・評価工程および客先稼働等における重度障害対応

振返

【定期的な振り返り】

◎工程ごとの振り返りは、失敗予防・リスク排除・課題解決として機能する。

振返

【振り返り会議】
◎振り返り会議は、犯人捜しではなく原因・対策の情報共有の場にすること。

振返

【プロジェクト完了報告】
◎プロジェクト完了報告書は、そのプロジェクトのあるがままの結果を示すと同時に、
次に続くプロジェクトに活きる目標を提供する必要がある。

【Human Activity】

振返

【失敗の原因を他に求めない】
◎振り返りは、失敗の経験に学び個人・組織ともに成長する有効な手段であり、
プロジェクトのけじめであると同時にノウハウ継承の役割をもっている。

振返

【モチベーションの喚起】
◎やる気やモチベーションの喚起は、
開発者の時間不足の解消によってもたらされる。

第７章

ノウハウ継承のプロセス

【統合マネジメント】

概要

第７章　概要：ノウハウ継承のプロセス

ノウハウの継承による、個人および組織の循環する成長についての解説です。

- ノウハウの継承は、個人および組織を成長させる（７－１．）
- ノウハウは、組織の誰もが利用可能な形で蓄積しておくこと（７－１．）
- ノウハウの継承は、開発行動の全ての場面で行われる（７－１．）
- ノウハウの蓄積や継承が行われる最も効果的な場は、改善活動の場である（７－２．）

ノウハウ継承のプロセスは、一つのプロジェクトの最後のプロセスであると同時に次に続くプロジェクトの最初のプロセスであるとも言えます。このプロセスは時系列的に最後にまとまって位置しているものではなく、開発の準備工程から振り返り工程に至る、すべてのプロセスの中で常時、組織的に人から人へと、プロジェクトにおける貴重なノウハウが受け渡される活動のことを意味しています。

本章では、ノウハウ継承の意義、継承の場面、蓄積すべきノウハウにつき、順に解説を行い、ノウハウ継承の最も強力的かつ効果的な活動である改善活動の採用について触れ、本書の主文の締めとします。

なおノウハウ継承のプロセスにおいては、Job　Activity および Human　Activity は混然一体となって実行されますので、二つの Activity をまとめた形で記述しました。

7－1 ノウハウの継承

ノウハウ継承の意義

近年あらゆる方面において組織が弱体化している原因の一つとして、個人の孤立化による連携・連帯[181]の喪失がノウハウの継承を著しく妨げているのではないかということが考えられます。

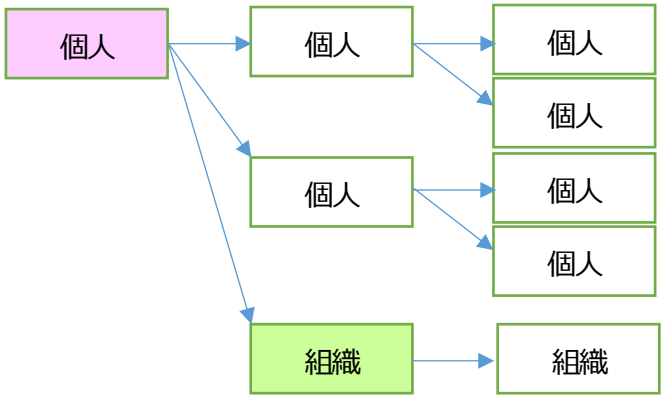
ノウハウの継承は、プロジェクトの有形・無形のノウハウを個人のみならず組織全体に伝える重要な活動です。ノウハウの継承を行わなければ、個人はもとより組織も確実な成長を手にすることはできません。個々人の連携が弱く孤立的なプロジェクトにおいては個人の成長も人によりけりで、それぞれが持つノウハウの結集も行われにくく組織的な成長は一向に進みません。

ノウハウ継承の重要性はみなさんもお分かりのことだと思いますが、本章では一体どうすればそれができるようになるのかということについて考えてみたいと思います。

継承

◎ノウハウの継承は、
プロジェクトの有形・無形のノウハウを個人のみならず組織全体に伝える。

【ノウハウの継承】（図７－１－１）



181 **連帯** 共通の目的に向って、共に責任を分担して行動すること。

継承

ノウハウの蓄積

多くのノウハウは失敗の経験の中から得られます。ノウハウの継承を促進するためには、個人の頭の中にあるノウハウをみんなの目に見える形で蓄積しておく必要があります。開発ノウハウは組織の財産であり、それを生きた財産として使うためには、個人が抱え込んでいるノウハウの組織的な共有化を進める必要があります。 どの開発プロジェクトの誰でも、必要な時にはいつでも、アクセス可能なノウハウデータベース[182]の構築が必要です。蓄積すべきノウハウとしては次のようなものがあります。

【蓄積すべき開発ノウハウ】

① 失敗事例における、失敗内容・原因・歯止め対策などの情報
② 成果を出した改善活動内容の情報
③ 開発基本ドキュメント
　　要求仕様書、見積り書・契約書、基本設計書、詳細設計書、評価設計書、レビュー記録、等
④ 開発管理表　：
　　プロジェクト計画書、プロセス管理表、リスク管理表、課題管理表、改善活動計画書
　　要求仕様変更管理表、品質管理表、損益管理表、進捗管理表、等
⑤ 顧客情報　：組織情報、顧客の志向・特徴情報、仕様情報、システム運用情報、等

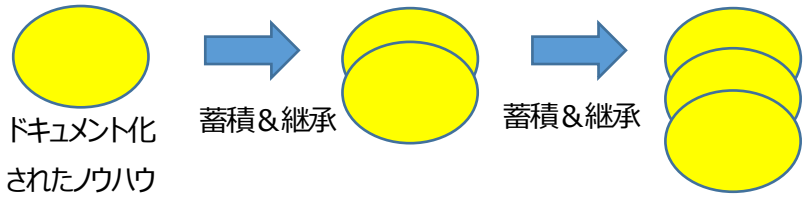
前記を見ても分かる通り、すべてのノウハウはドキュメントの形にしなければ組織全員の共有知識にすることはできません。さらに共有財産と呼ぶのにふさわしいドキュメントとするためには、正しい内容であり、不明点や疑問点のないものである必要があります。ドキュメントについては第４章　実行プロセスの「４－３．品質目標値の達成☆ドキュメントベース開発」を再度ご参照ください。

ドキュメントによるノウハウの組織的な共有化は、プロジェクトチームの組織戦を強力にバックアップします。

継承

◎蓄積すべきノウハウは、失敗情報・改善活動情報・開発ドキュメント・顧客情報の四つ。

【ノウハウは金のたまご】（図７－１－２）



182 **ノウハウデータベース**　ノウハウを集めたデータベースのことで、関係者なら誰でもいつでも容易に利用可能なシステム。

ノウハウの継承はいつどこで行われるのか

最後のプロセスとしてノウハウの継承を取り上げましたが、ノウハウの継承はプロジェクト活動の最後にだけ実行されるものではなく、開発の全ての工程において実行される必要があります。

たとえば要求仕様の調査・検討時において、不明点・疑問点に関して知識保有者に質問し回答を得ることや要求仕様書のレビューにおけるレビュワー[183]からの指摘・アドバイスなど、知識的に後進の人がより知識のある人からその知識を得ることなどはノウハウの継承そのものだと言えます。

同様なことは見積り・設計・製造・評価などすべての開発工程において必要とされます。

ノウハウの継承が行われる場面としては次のようなものがあります。

【ノウハウの継承が行われる開発シーン】

① 日常の業務活動中における疑問点・不明点についての質疑応答
② 日次情報共有会議におけるリーダーからの指導や助言
③ 改善活動における改善策の策定および実行
④ 各工程レビューにおける指摘や助言
⑤ 緊急不具合対策時における上級技術者からの指導
⑥ プロジェクト完了会議における失敗事例の総括
⑦ 講習会・勉強会におけるノウハウの学習

ノウハウの継承は、ここに示したように何らかの会話や会議の場において行われるもので、ノウハウの継承を促進させるためにはコミュニケーションを計画的に活発化し、その中身を意味のあるものにする必要があります。 そのためにはプロマネは開発のあり方を個人戦ではなく組織戦としてリードしていく必要があります。

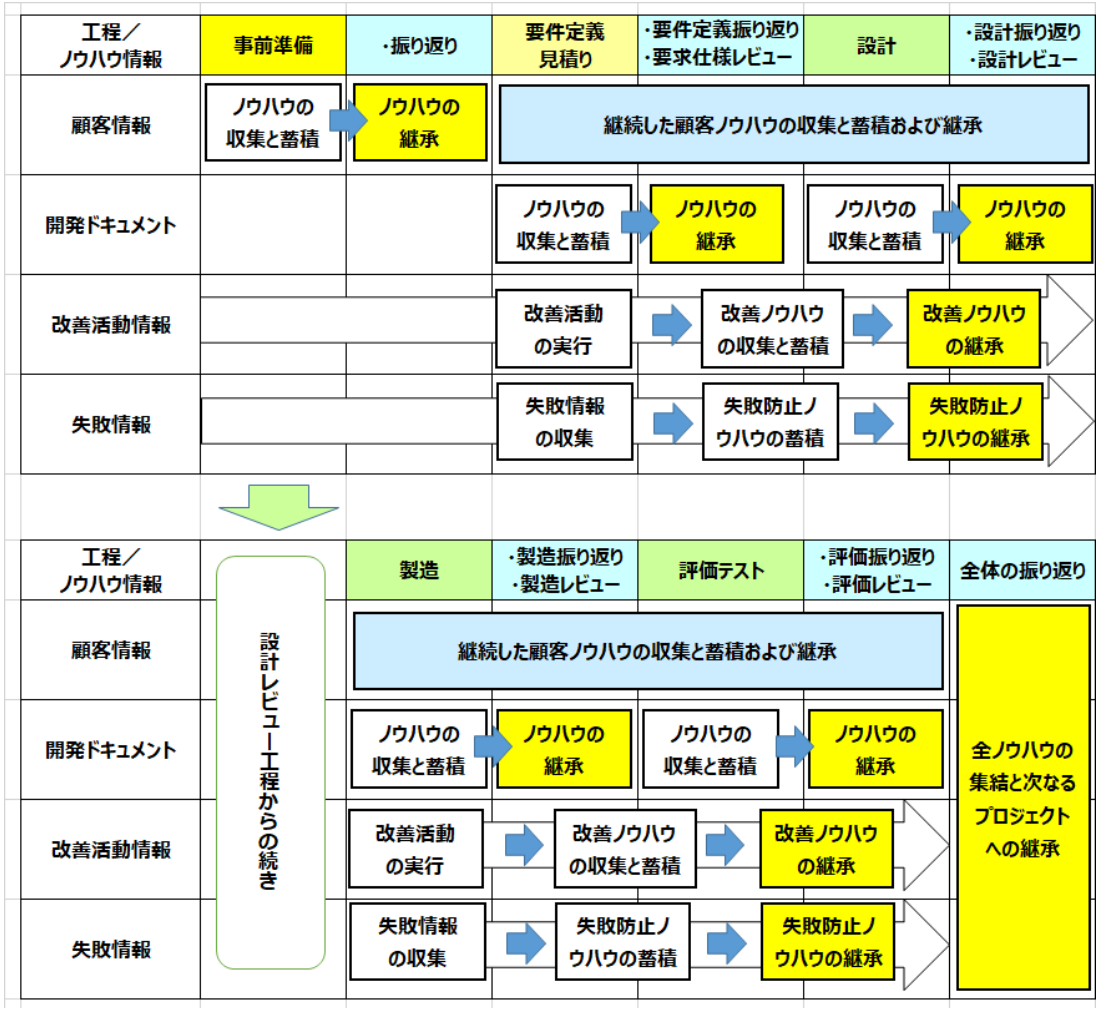
コミュニケーションについては第５章　管理プロセスの「５－ｈ１．コミュニケーションの活性化」を再度ご参照ください。

継承

◎ノウハウの継承は、開発の全工程におけるコミュニケーションの場で実行される。

183 **レビュワー** レビューをする人。反対にレビューをしてもらう人はレビューイと呼ばれる。

【開発工程内におけるノウハウの収集・蓄積と継承のイメージ】（図７－１－３）



上図は開発工程内におけるノウハウの収集・蓄積と継承を表わしたイメージ図です。

ノウハウ情報としては、新たに知り得た仕様情報や失敗情報をはじめとしたリスク情報や開発中に作成された諸設計書・諸管理表、改善活動で得られた新しい効率化手法などさまざまなものがありますが、これらの貴重な情報を間違いなく継承するためには、これらの情報が発生した時点で何らかのドキュメントに書き残しておき、工程毎の振り返りやレビューだけではなく日々の開発業務を通して後進の開発者たちに継承していく必要があります。

一見これらの活動は大変なようですが、プロジェクトの規模に応じて継承すべきノウハウの量もそれなりのものになりますので、規模が小さいプロジェクトにおいては短時間で可能な活動で済むはずです。

これらのノウハウ継承活動は、プロジェクトのＱＣＤを成功に導くだけではなく、開発者たちのスキル向上をもたらすでしょう。

7－2 改善活動のすすめ

改善活動の必要性については「４－６．改善活動の実行」において触れた通りですが、ノウハウ継承の観点から見た改善活動の意義について、本書における最後の重要なプロジェクト活動として取り上げたいと思います。

前節で取り上げたノウハウ継承を促進させるための施策である、ノウハウデータベースの構築・ドキュメントの整備・ドキュメントの品質向上は、みな改善活動という旗を立てなければなかなか組織的な取り組みにまでは至りません。本当に効果のある改善活動ならば、最初は小規模な成果かも知れませんが、プロジェクトの開発予算や期間内で一定の成果を出すことは可能であり、数年間も継続できれば大きな成果を手にすることができます。 組織的な改善活動によって、はじめて本格的なノウハウの継承が可能になります。

プロジェクトにおける「譲り」という行為

継承とは価値のあるものを、それを保有している人から保有していない人に対して譲り渡すという行為に他なりません。改善活動を通したチーム内におけるノウハウの譲り、および組織間におけるノウハウ譲りについて考えてみたいと思います。

チーム内におけるノウハウの譲り

プロジェクト活動における重要な目的の一つとして、人材の育成およびＱＣＤの確保があります。

人材の育成は組織における永続的な成功を支える基本要件となるものです。優れた人材はいわば金の卵を産む親鳥に相当します。

ＱＣＤの確保は顧客満足の基本要件の一つであり、これもまた組織の永続的な成功を支える基本要件となります。この人材育成とＱＣＤの確保を同時に達成する方法は、組織的な改善活動をおいて他には見当たりません。

改善活動は開発チームの弱点の克服運動であり、それは製品品質の向上や業務の効率化という具体的な改善行動を通して、上位技術者から下位技術者や同僚に対するノウハウの伝承という形で行われるものです。改善活動は、下位技術者のレベルアップと同時に上位技術者のさらなるスキルアップをも実現する活動だといえます。

開発チームのメンバーたちに対するノウハウの伝承や支援という譲りの行為は、間違いなくメンバーたちのスキル向上の結果を生み、チーム全体の開発力や信頼関係を強化します。

このような譲りというノウハウの継承が、習慣的に相互に実行されているようなチームは、顧客の期待に応える製品を生み出すと同時に、会社にとっても大きな利益をもたらすでしょう。

組織間におけるノウハウの譲り

またプロジェクトにおけるノウハウの継承については、先進的なチームから他の失敗プロジェクトに対するレスキュー活動という形でも行われます。失敗プロジェクトの再生に必要なものは、ノウハウであり時間であり資金です。プロジェクトの再生に必要なものの投入を通して、失敗チームのマネジメントおよび開発メンバーの再教育を効果的に行う必要があります。ただ一時の急場を助けるだけではなく、レスキューを受けたチームの全員が新たなノウハウを身に着け、将来の失敗を防止し、さらにまた別のプロジェクトを支援できるようにする必要があります。

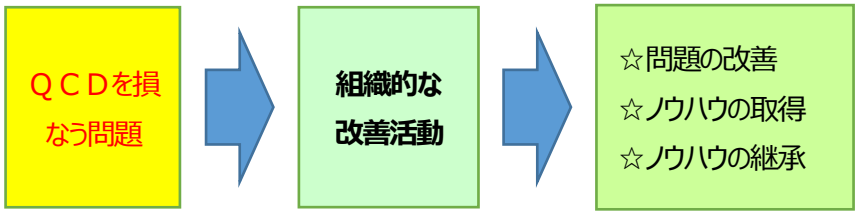
改善活動やレスキュー活動は、そのノウハウや資源を他に「譲る」という行為を通して、組織全体を永久循環成長のサイクルに乗せる重要な役割を担っているものです。

製品が売れて儲かった場合、営業は自分の営業の力だと言い、開発は自分の開発の力だと言い争うことがあります。それぞれの努力を否定はしませんが、本当の功績は過去の営業・開発の先輩たちの功績によって築かれたブランドの信用力に負うところが多いのです。私たちは駅伝における中継ランナーの一人であるという自覚が必要であり、他の人たちから受け取った良いものをさらに良いものに磨き上げ、次のランナーに受け渡す義務があるのです。この連綿と続く受け渡しのことを「継承」あるいは「譲り」と言います。

継承

◎ノウハウの継承とは、個人間および組織間における価値あるものの「譲り」である。

【改善活動を通して実現される組織的なノウハウの継承】（図７－２－１）



奪うと譲る

継承という行動は自分の時間と労力を使って、他人にノウハウを伝える重要な行動であり、他人のために自分の時間と労力を譲る行為そのものです。その「譲る」という行為の反対として「奪う」という行為を見てみることで我々のノウハウ継承を阻害するものについて考えて見たいと思います。

「譲る」の反対は「奪う」です。利を争うとは、他人から利を奪い取るということに他なりません。他人から奪い取れば他人は困窮し、奪った相手を怨むことになります。奪い過ぎれば相手は生きていけません。働く人たちを正規労働者と非正規労働者に分け、経営者や会社の利を増やすことも「奪う」ことに他ならないでしょう。その利の配分に過剰な格差があった場合、非正規労働者は健康に生きていくことができなくなります。戦前まであった自作農と小作人の関係の再現のようです。高い小作料に苦しんだ小作人たちは、一旦飢饉や天変地異に見舞われた場合、身売りをしても生きていけないほどの地獄に落とされてしまったのです。

このようなことを続けていけば民力は確実に落ちていき、ついには上も下も共々衰退と滅亡の憂き目に会うことは、つい七十数年前以前の歴史が示す通りです。

開発プロジェクトにおいても、それをリードする者が無能な余り、将来のある人間を酷使した上に病気や死に追い込むようなことは絶対にしてはいけません。プロジェクトの中で心身の病を多発させるような上司を放置しておいてはいけません。開発組織における最大の悪は、無能なプロマネだと強く認識しておく必要があります。彼らは人材を破壊しプロジェクトを破壊する者たちに他なりません。

ＩＴプロジェクトにおける不条理な行為には以下のようなものがあります。

【ＩＴプロジェクトにおける不条理さの事例】

- □ 実現不可能な短納期の要求
- □ 実現不可能な低コストの要求
- □ いつまでも提示されない要求仕様
- □ 開発の後半での大幅な仕様変更要求
- □ 仕事の丸投げ・責任および職務の放棄
- □ 契約に含まれていない開発や業務の強要
- □ 優位な立場を悪用した責任転嫁
- □ 根拠のない誹謗中傷行為や発言
- □ 能力の限界をこえた指示および要求

これらの奪う行為は、人や組織の健全性を損なうだけではなく、プロジェクトの目標であるＱＣＤさえも大きく損なってしまうものです。万全の注意と対策が必要です。

継承

◎不条理な要求は、要求された者を最初に損ない、続いて要求した者をも損なう。

継承

譲りがもたらした成果

ある改善プロジェクトは、貧乏暇なしの状態から脱出すべく無駄の撲滅を筆頭に全工程にわたる改善活動を始めました。改善益は、最初は小額でしたが数年後には大きな改善益を出せるようになりました。そこでプロマネが取った策は利益三分の計というものでした。一つは会社と約束した利益目標分に充て、二つは営業からの値引き要求の原資に充て、三つは次なる開発の準備金に充てました。これらの策は貯蓄ではなく投資と呼んだ方が良いのかも知れませんが、いずれも将来の自分たちの開発組織への贈り物として機能しました。

一つ目の策は会社の上部マネジメントを満足させ、二つ目の策は営業部に対する影響力を強化させ、三つ目の策は開発リスクの低下・品質向上および生産性の向上に大きな効果を現しました。

営業部に対する値引きの提供は従来の資金の流れの逆をいくもので、金のあるところに権力は集まるというように、それまでは営業部に隷従を強いられていた開発部を自律的な組織に変えることに成功し、営業部と対等なパートナーシップを築くことができました。

次なる開発に対する事前投資は、リスク物件における事前調査やプロトタイプの作成および種々の生産性向上のための開発ツール等の事前開発を可能にし、リスクの排除および生産性の向上に大きく貢献しました。

継承

◎譲り・ノウハウ継承の代表的な例が改善活動であり、
実行した分だけの実りが必ずもたらされる。

譲りは行われにくい

以上の譲りの行動はいずれも自他ともに大きな益をもたらすものであることは分かり切った話です。しかしながら、少なからず自分に負担を強いる必要があるために、人や組織によっては実行されないことも多いのです。 成長できる人や組織とそうでないものの差は、この譲りの多寡によって決まってきます。多く譲れば譲った分だけの恵みがもたらされ、譲りをしなかった者は没落の一途をたどる運命が待っているというだけのことです。

最初から、譲りはそもそも出来にくいということを強く認識しておく必要があります。

継承

◎人間の本性は譲ることを嫌う。
そのため譲り・ノウハウ継承・改善活動は実行されにくい。

改善活動のコンセプト

懸命に働くではなくて賢明に働くことや、切り詰める節約ではなくリソースを有効に使うことや、継承という譲りは、個人のみならず組織やプロジェクトに永続的な繁栄をもたらします。

継承・譲りという思想を中核に据えた、ある改善活動チームの活動コンセプトは次の３点でした。

【改善活動のコンセプト】

＃１．時間の獲得

＃２．自律型開発への転換

＃３．変化即応型開発の実施

活動の概要を以下に示します。

コンセプト＃１．時間の獲得

ものごとを遂行するにおいて時間の獲得は究極の必要条件です。ヒト・モノ・カネ・情報などのリソースはなんとかなる可能性がありますが、失われた時間を取り戻すことはできません。プロジェクトの究極の成功要件は、必要時間の確保にあることを深く認識し、必要時間の獲得および無駄な時間の削減に必要なあらゆる対策を講じる必要があります。そのアクションは以下の通りです。

アクション：

① 先行調査によりリスクを排除し、後工程で発生する不始末による膨大な時間ロスを防ぐ。
要求仕様の疑問点・不明点の払拭、未経験の仕様・機能の調査・理解、プロトタイプ作成による未知の新技術領域のリスクの排除などを実行する。

② 要件定義工程において、早期の仕様凍結を目指し、顧客との集中合宿検討会を含む密接かつ頻繁な仕様検討会を行い、顧客とのコミュニケーションを緊密にすることで、要求仕様の誤解やモレをなくし、後工程で発生する膨大な時間ロスを防ぐ。

③ リスクを考慮した妥当かつ正確な見積りを行うことで、開発に必要な絶対時間を確保する。
要求仕様および開発全体に関するリスクを把握し、開発に必要な時間と予算の確保を行う。

④ 開発プロジェクト内のコミュニケーションを緊密にすることで、指示の誤解やモレをなくし、後工程で発生する膨大な時間ロスを防ぐ。毎日ベースの日次情報共有会議の実行、週次・月次のまとめ会議の実行などの直接コミュニケーションによるリアルタイムな情報の共有および関係者間の信頼の醸成を図る。

継承

⑤ 重要な機能から先に開発を行い、完成の都度、顧客と現物にて動作確認を行うことで、やり直し・修正などの膨大な時間ロスを防ぐ。 顧客に対して７日～１０日ごとの重要仕様順のリリースを繰り返し、顧客および開発チームによる共同検証を行い、不具合修正・仕様変更等は顧客との直接対話による即断・即決による対応を行う。

⑥ 上記を、より可能にするために顧客および開発チームは場所および情報の共有を行い、開発を共同して遂行する。

効果：

① プロトタイプ作成による未知の新技術領域特有の不具合のあぶり出しによる、本番開発における致命的な不具合の予防。

② 顧客との密接直接コミュニケーションによる、要件定義における相互理解・相互信頼関係の醸成。

③ 要求仕様の早期凍結をベースにした妥当かつ正確な見積りは、プロジェクトの絶対的な成功要因である必要時間の確保と必要予算の確保を確かなものにする。

④ 開発チーム内の日次・週次・月次の頻繁なコミュニケーションは、メンバー全員による問題の共有および情報の整理整頓を促進し、開発者における仕様の誤解・設計ミス・段取りミス等を大幅に削減すると同時に、開発者における開発完遂の強いモチベーションおよび信頼関係を最後まで維持させる。

⑤ 短期間で逐次リリースされた動作する現物のソフトウェアにての顧客との直接的検証・対話は、開発における後戻りや不要な修正等の無駄な作業を大幅に削減し、同時に顧客における安心感・満足感の醸成に大きく寄与する。

継承

◎プロジェクトにおける究極の制約条件は時間にある。
必要な時間の獲得・無駄な時間の削減はプロジェクトを成功に導く。

参照：第４章「４－ｈ１．獲得時間の最大化と失う時間の最小化」

【時間獲得のための改善活動のテーマ】（図７－２－２）

- ●事前準備・事前調査の推進
- ●早期仕様凍結
- ●見積り精度向上
- ●コミュニケーション活性化
- ●顧客価値優先開発の推進
- ●情報共有化の推進

ＱＣＤを成功に導くために
必要な時間の獲得

継承

コンセプト＃２．自律型開発への転換

マンネリ化したウォーターフォール型開発[184]組織を自律性のある組織へ転換させる。

アクション：

① **毎日やる**　☆課題バラシ[185]およびその対策を毎日実行する。
☆開発担当者を毎日巡回しフォローする。
☆対策の指示を毎日行う。

② **毎日聞く**　☆開発担当者の不平・不満・グチを聞く、関連部署の状況を聞く。
☆お客様の話を良く聞く。

③ **毎日見る**　☆毎日、リスク管理表・課題管理表を見る、開発担当者たちの進捗状況を見る。
☆毎日、開発担当者たちの健康状況を見る。自分の健康状況も見る。

④ **毎日まとめる**　☆変化する課題・問題を毎日、整理更新し、まとめる。
☆明日以降の必要なアクションをまとめる。

⑤ **現場に居る**　☆プロマネは心理的・地理的にいつも開発担当者のそばに居る。
☆プロマネはいつでも担当者の相談にのる。
☆プロマネは事務所よりも開発ルームで仕事をする。

⑥ **顧客の声を聞く**　☆お客様を早い時点で開発パートナーとして巻き込む。

⑦ **早くする**　☆自部署の担当ではないと判断した仕事は即刻担当部署に渡す。

⑧ **見えるようにする**
☆リーダーは、担当者・外注の仕事の範囲・進捗状況を見えるようにする。
☆必要な情報は口頭だけに頼らず、まず手書きでも良いから書面として見えるようにする。
☆スケジュール進捗表・管理表は毎日更新しておく。

これらの行動は開発者における自律性・協調性を促し、顧客との密な連携を促し、日々刻々の変化に即応するための基本的な行動のポイントとなります。

継承

◎自律性とは、
●目の前の現実や事実を自分の目で直視し、
●それが意味するところを自分自身で解釈・判断し、
●それをどのような方法で解決するのかを自分で決定し、自分で行動を起こすこと。
◎自律的な開発は、毎日の自律性に基づいた行動によって可能になる。

---

184 **ウォーターフォール型開発**　開発プロジェクトを時系列に、「要求定義」「概要設計」「詳細設計」「プログラミング」「テスト」「運用」などの作業工程に分割し、前から順次後へと開発を進める方式のこと。原則として前工程が完了しないと次工程には進めない。

185 **課題バラシ**　問題の発生原因と思われるものを可能な限り書き出し、類似したものを同一グループにまとめ、グループごとに有効と思われる対策を立案することで、複雑な問題に対する解決法を見出そうとする問題解決手法。

継承

コンセプト＃３．変化即応型開発の実施

変化即応型開発のポイントは下記の二点にあります。

１．顧客要求に対して俊敏な対応をすること。

２．開発チーム全員による意識・価値観の共有による目標の達成を目指すこと。

アクション：

① 顧客とのデイリー＆ダイレクトなコミュニケーションの実施。

② インクリメント[186]な成果物を基にした顧客と開発チームによる共同評価・検証の実施。

③ 開発チーム内のデイリー＆ダイレクトなコミュニケーションの実施。

アクションの具体例：

プロジェクトの主要な問題は人と人の間に発生するものであり、それらの問題をすばやく解決することからしかプロジェクトの成功は導き出せないという認識に基づき、顧客と開発チーム間あるいは開発チーム内において発生する、課題・問題を、成果物である現物に基づいて直接的なコミュニケーションによって毎日ベースに俊敏に解決する。

継承

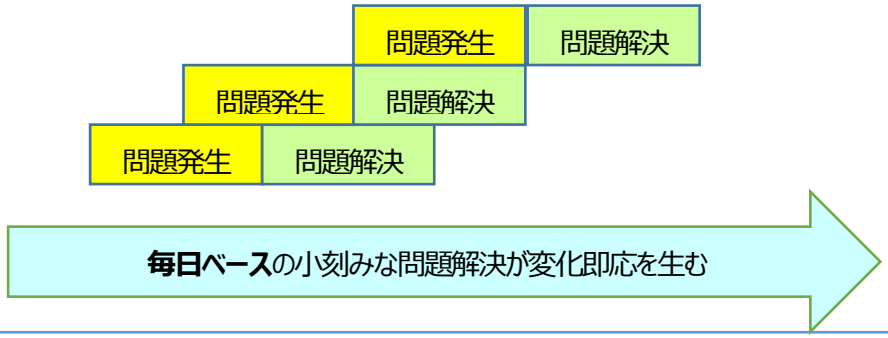
◎柔軟性に富んだ変化即応型のプロジェクトが持つ三つの行動特徴。

１．顧客とのデイリー＆ダイレクトなコミュニケーションの実施。

２．インクリメントな成果物を基にした顧客と開発チームによる共同評価・検証の実施。

３．開発チーム内のデイリー＆ダイレクトなコミュニケーションの実施。

【変化即応型開発】（図７－２－３）

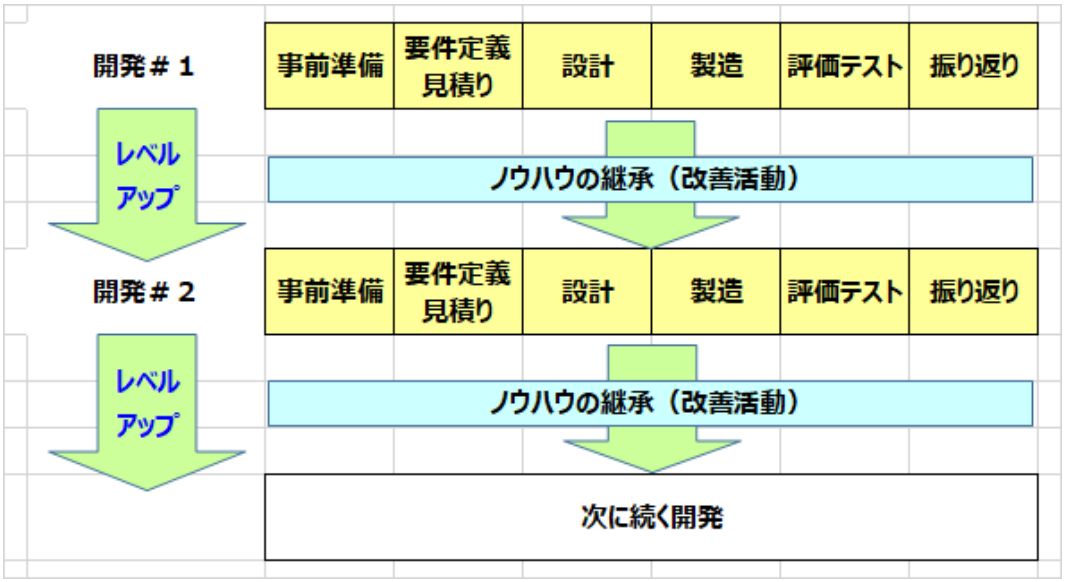


186 **インクリメント** アジャイル開発などで逐次分割してリリースされる成果物をインクリメントな成果物と呼んでいる。アジャイル開発では、従来のウォーターフォール開発で行われているような、たくさんの要求仕様をまとめて開発・検証・リリースするのではなく顧客が要求する順に従って幾つかの仕様群に分割して順次開発・評価・リリースが行われる。

継承

譲り・ノウハウの継承・改善活動の実行は、同じことを違った言葉で表したもので、これらのことを継続して実行するプロジェクトは永続する継承循環のサイクルに入ることができ、プロジェクトのＱＣＤの成功のみならず個人および組織のスキルアップ、開発力の永続的な発展を実現することができるでしょう。

【ノウハウの継承循環】（図７－２－４）



【ノウハウ継承プロセスのまとめ】

ノウハウの継承は個人および組織の成長の源であり、先行する者から後に続く者へ、先行プロジェクトから後に続くプロジェクトへのノウハウの継承は、単にプロジェクトのＱＣＤ目標の達成を確かなものにするだけではなく、個人および組織における成長の好循環サイクルを実現するでしょう。

継承

【ノウハウの継承先】
◎ノウハウの継承は、
プロジェクトの有形・無形のノウハウを個人のみならず組織全体に伝える。

継承

【蓄積すべきノウハウ】
◎蓄積すべきノウハウは、失敗情報・改善活動情報・開発ドキュメント・顧客情報の四つ。

継承

【ノウハウの継承はどの工程で実行されるのか】
◎ノウハウの継承は、開発の全工程におけるコミュニケーションの場で実行される。

継承

【ノウハウの継承は価値あるものの譲り】
◎ノウハウの継承とは、個人間および組織間における価値あるものの「譲り」である。

継承

【他から奪う行為の行き着くところ】
◎不条理な要求は、要求された者を最初に損ない、続いて要求した者をも損なう。

継承

【ノウハウ継承の代表的な例】
◎譲り・ノウハウ継承の代表的な例が改善活動であり、
実行した分だけの実りが必ずもたらされる。

継承

【継承は実行されにくい】

◎人間の本性は譲ることを嫌う。

そのため譲り・ノウハウ継承・改善活動は実行されにくい。

改善活動のコンセプト

継承

【時間の獲得】

◎プロジェクトにおける究極の制約条件は時間にある。

必要な時間の獲得・無駄な時間の削減はプロジェクトを成功に導く。

継承

【自律型開発への転換】

◎自律的な開発は、毎日の自律的な行動によって可能になる。

継承

【変化即応型開発の実施】

◎柔軟性に富んだ変化即応型のプロジェクトが持つ三つの行動特徴。

① 顧客とのデイリー＆ダイレクトなコミュニケーションの実施。

② インクリメントな成果物を基にした顧客と開発チームによる共同評価・検証の実施。

③ 開発チーム内のデイリー＆ダイレクトなコミュニケーションの実施。

おわりに

最後までお読みいただき有難うございました。

プロマネの基礎知識全般について、現場で実際に使用した管理帳票類、コンセプト図解、脚注キーワードなどを用いて、難解な内容を少しでも分かりやすく記述したつもりですが、いかがでしたでしょうか。

ソフトウェア開発はきつくて難しい仕事だとよく言われていますが、本来仕事というものは誰かが困っていることをその人に代わって解決することですから、世の中に簡単で楽な仕事はないと思います。そのように難しい仕事を自分たちの不手際でさらに困難なものにしないように、最初に用意周到に準備や段取りを整えることがプロとしての最初の仕事であるように思います。

また仕事を進めるにあたっては個人戦を避けチームの力を結集する組織戦の実行が成功のポイントになりますが、チームの力を結集するためにはプロマネという人間がメンバーから信頼される人間である必要があります。読者諸氏におかれましては、プロジェクトのやっかいな問題に先陣を切って取り組むようなプロマネやリーダーになられることを期待致します。 そのように他人先幸に努めれば何事もきっとうまく行くことでしょう。

なお本書を読んでプロマネの基本知識を大体理解されたとしても、それだけではすべてを実行することは難しいと思います。プロマネ業務のあるべき姿が分かったとしても、現実の仕事においては、さまざまな問題やリスクが発生してきます。そのために、全てを一度に実行することは難しいと思われますので、読者のそれぞれのレベルに合わせて、可能なものから取り組まれ、現実に発生してくる問題に逐次対応しながら経験を積み重ね、順次ステップアップを図って行かれることをお勧め致します。

本書に記述されたプラクティスはすべて実際の開発の中で得られた実証済の経験則であり、その考え方や実行方法もごく単純なものです。これらの中で現在の自分の課題と合致するものから取り組み、最初は小さな成果であっても決してあきらめることなく改善活動を継続し、納得の行く成果を手にされることを期待します。

つたない内容であったかもしれませんが、お読みいただいた方々の力添えに少しでもなれば幸いです。

皆様のプロマネ人生が大きく開けることを願っております。

最後に、本書の執筆にあたっては株式会社ドゥシステムのプロジェクトマネージャおよびソフトウェア開発技術者のみなさん方を中心として、東芝テック株式会社、テックインフォメーションシステムズ株式会社、東京システムズ株式会社の方々から多くの貴重なご意見をいただき厚く感謝申し上げます。

参考資料

・ ＩＥＥＥ８３０品質特性、ＵＳＤＭ （３－１．●要求仕様書の詳細チェックリスト）
・ アジャイルソフトウェア開発宣言 （３－８．プロジェクト体制の構築）
・ 日経ＩＴプロフェッショナル 2002 年 10 月 （５－４．●カネの進捗管理）
・ ＮＴＴデータ経営研究所、「会議の革新とワークスタイル」に関する調査、2012 年 10 月 5 日
（５－ｈ１．●会議における良好なコミュニケーション）
・ SQUSE 通信 Vol.80 チームプレーの魅力 （５－ｈ２．●チームプレーとはそもそも何か）
・「失敗の本質」日本軍の組織論的研究（５－ｈ２．●チームプレーを可能にするリーダーシップ）
・ 岩崎敏夫著「二宮尊徳仕法の研究」、二宮尊徳著「御仕法掛心得方大略」
（５－ｈ２．●チームプレーを可能にするリーダーシップ）
・ ７種類の虫のイラスト（表紙および各章の開始ページ） ©いらすとや

付録図表

１．プロマネ業務一覧：第１章１－２．●プロマネの基本的な役割
２．プロセス管理表：第１章１－４．プロセスをマネジメントする、表１－４－１
３．プロセスとマネジメントプラクティスの関連図：第１章 まとめ ３．、表１－まとめー１
４．事前準備工程のリスク管理表：第２章 ２－３．事前準備工程のリスク、表２－３－１
５．要求仕様書 採点表：第３章 ３－１．●要求仕様書の精度検証、表３－１－２
６．要件定義工程のリスク管理表：第３章 ３－２．要求仕様に関するリスク、表３－２－１
７．スケジュール表（大／中日程）：第３章 ３－６．スケジュールの作成
８．設計工程のリスク管理表：第４章 ４－１．●開発現場にころがるリスクの事例、表４－１－１
９．課題管理表：第４章 ４－３．●課題管理表、表４－３－１
１０．ＤＭＡＩＣシート簡易版：第４章 ４－６．●改善活動計画書、図４－６－１
１１．仕様変更管理表：第５章 ５－１．●仕様変更管理表がもたらす効果、表５－１－１
１２．品質状況管理表：第５章 ５－２．●品質状況管理表、表５－２－１
１３．不具合習熟曲線表：第５章 ５－２．●不具合習熟曲線表、図５－２－１
１４．原価管理表：第５章 ５－３．●原価管理表、表５－３－１
１５．機能モジュール進捗管理表：第５章 ５－４．●モノの進捗管理、表５－４－２
１６．労働負荷状況管理表：第５章 ５－４．●ヒトの消耗度管理、表５－４－５
１７．プロジェクト完了報告書：第６章 ６－２．プロジェクト完了報告書
●仕様変更管理の考察、表６－２－１
●品質管理の考察、表６－２－２
●コスト管理の考察、表６－２－３
●進捗管理の考察、表６－２－４
１８．開発工程内におけるノウハウの収集・蓄積と継承のイメージ
：第７章 ７－１．●ノウハウの継承はいつどこで行われるのか、図７－１－３

１．プロマネ業務一覧：第１章１－２．●プロマネの基本的な役割

【プロマネ業務一覧】

主要業務	細目
①適正な見積り交渉の実行	・適切な開発費と開発期間／評価期間の確保
	・リスク物件における分割見積り等（分割開発・分割リリースの交渉）
	・複数社によるプロジェクトでは自社責任範囲を見積り回答書の条件に記述する
	・過去の類似開発のＱＣＤ見積り・実績データベースを参考にした見積りの実行
②早期仕様凍結の実行	・顧客との直接コミュニケーションによる要求仕様の期限内凍結の実行
	・顧客の参加・協力の要請により要求内容／仕様凍結期限をあいまいなままにさせないこと
	・開発目的・範囲・内容の明確化、あいまいな仕様・開発範囲の撲滅
	・要求仕様の優先度順位の明確化および優先度順の仕様凍結・開発実行
	・仕様提案型仕様凍結の実行
	・仕様未凍結状態での先行着手の禁止
③プロジェクト計画書の作成	・プロジェクトの特徴・難易度に応じた、妥当性のある開発／評価体制の構築
	・開発技術情報の共有化によるメンバーの育成計画
	・採用ルールの規定に基づいた協力会社からの適正な人材の採用
	・開発環境の準備・整備
	・開発スケジュールの策定
④プロセス管理の実行	・プロセス管理表の作成と運用
	・仕様凍結遅延防止活動の実行
	・妥当な設計・製造・評価工程期間の確保
	・開発工程内の各工程のスケジュールの遵守および主要イベントの進捗管理
	・各工程の出入り口・中間におけるレビューの実行
	・各工程における成果物の妥当性の管理
⑤プロジェクト進捗管理の実施	・プロジェクト進捗表の作成と運用
⑥リスク管理の実施	・リスク管理表の作成と運用
	・プロジェクトの全工程におけるヒト・モノ・カネ・情報等に起因するリスクの発掘と解消の実行
⑦課題管理の実施	・課題管理表の作成と運用。プロジェクトの全ての課題の明確化と問題解決の実行
⑧損益管理の実施	・プロジェクト損益管理表の作成と運用
	・見積り工数と実績工数の適正な推移の管理と対策の実行
⑨プロジェクト完了報告	・プロジェクト活動における失敗の真因および再発防止策のまとめ等の振り返り
	・ＱＣＤの目標値／実績値の対比および原因分析およびデータベースへの登録
	・今後の課題のまとめと振り返り結果の他チームへの横展開
⑩日次情報共有会議の実行	・チーム内の情報共有
	・毎日の行動結果・行動予定の確認
	・毎日の課題／リスクの掘り起こしと対応状況の確認
	・開発目的・範囲・情報の徹底
	・開発仕様の情報共有
	・開発仕様の優先度順位情報の共有
⑪プロジェクト関係他社開発チームとの情報共有および連携	・複数会社で構成されるプロジェクトにおいては、自社担当領域に接する他社との情報共有、連携行動を実行する

2．プロセス管理表：第１章１－４．プロセスをマネジメントする、表１－４－１

【プロセス管理表】（サンプル）　　＊ＰＬ（ｏｒＰＭ）、ＳＰＬ（ｏｒＳＰＭ）

	NO	手 順	進捗	作 業	担当者	成果物	担当チェック	ＰＬチェック	実施予定日	実施日	レビュー結果
事前準備	1	顧客・ステークホルダーの情報収集		顧客の要求内容・予算・納期等の事前把握	開発部	・調査報告書					
	2	新ＯＳ・新言語・新システム対応の準備		新ソフトウェア・新ハードウェア等に対する技術知識のマスターおよびプロトタイプによる事前調査	開発部	・調査報告書					
計画プロセス	1	見積もり依頼 見積もり回答		仕様，スケジュール，要員検討、見積依頼	ＰＬ，ＳＰＬ	・見積回答書 ・リスク管理表					
	2	要求仕様の明確化		要求事項の明確化	ＰＬ	・要求事項の明確化チェックリスト作成					
	3	仕様検討、承認		SEとの仕様打ち合わせ	ＳＥ,ＰＬ、開発メンバー	・議事録／要求仕様書の承認					
	4	不具合歯止策の選定、決定		開発に対する有効な歯止め策を	ＰＬ、ＳＰＬ、開発メン	・歯止めチェックリスト					
	5	スケジュール		大、小日程の作成	ＰＬ、ＳＰＬ	・大日程表、小日程表					
	6	プロジェクト品質目標の作成		プロジェクト内品質目標の作成	ＰＬ、ＳＰＬ	・プロジェクト品質目標					
	7	プロジェクト計画書		プロジェクト計画書の作成 リスク管理表更新	ＰＬ，ＳＰＬ	・プロジェクト計画書					
				プロジェクト計画レビュー 開発開始会議	ＳＱＡグループメンバー、ＰＭ，ＰＬ	・レビュー記録					
	8	外注見積委託作成		外注見積委託ＤＢに登録	ＰＬ，ＳＰＬ	・見積書					
	9	開発管理		開発管理資料の作成	ＰＬ，ＳＰＬ	・プロセス管理表					
						・スケジュール表					
						・課題管理表					
						・規模進捗					
	10	デザインレビュー①		要求事項の明確化チェックリスト	ＳＥ→開発部	・要求仕様の採点確定					
実行・管理プロセス	1	仕様書		仕様書	担当者	・仕様書各種					
				仕様書レビュー	ＰＬ、ＳＰＬ、開発メン	・レビュー報告書					
	2	初期設計		初期設計	各担当	・変更モジュール一覧表					
				初期設計レビュー、スケジュール	ＰＬ，ＳＰＬ	・レビュー報告書／変更モジュール一覧表					
	3	デザインレビュー②		ＳＥ確認：要求事項の明確化チェックリスト確認	開発部→関係各位	・仕様書 ・設計書					
	4	詳細設計		詳細設計	各担当	・詳細設計書／リリース影響度表					
				詳細設計レビュー	ＳＰＬ、開発						
	5	部材の確認		部材の変更	ＳＰＬ	・新規部材					
	6	製造		製造	各担当	・ソースファイル／影響度表／変更モジュール一					
				ソースレビュー	ＳＰＬ、開発	・ステージ別チェックリスト／処理条件マトリクス					
	7	評価（単体）		チェックリスト作成	各担当	・チェックリスト					
				チェックリストレビュー	ＳＰＬ、開発	・レビュー報告書					
				デバッグ	各担当	・チェックリスト更新／試験結果					
				デバッグレビュー	各担当	・レビュー報告書					
	8	評価（結合）		チェックリスト作成	各担当	・テストチェックリスト					
				チェックリストレビュー	ＳＰＬ、開発	・レビュー報告書					
				開発チーム評価	各担当	・試験結果					
				評価レビュー	ＳＰＬ、開発	・レビュー報告書					
	9	ドキュメント		・仕様書フィードバック ・設計書フィードバック	各担当	・仕様書 ・設計書					
	10	総合テスト、第三者検証 品証検証 客先検証		チェックリスト作成	評価	・チェックリスト					
				チェックリストレビュー	ＳＰＬ、評価						
				評価	評価担当、品証担当	・チェックリスト更新／試験結果					
				評価レビュー	ＰＬ、ＳＰＬ、評価	・レビュー報告書					
				客先検証	開発部→ＳＥ、客先	・客先検証報告受領					
	11	デザインレビュー③		要求事項のチェックリスト確認	品質保証部→関係部署	・総合テスト報告書／第三者検証報告書					
	12	店舗リリース後確認 不適合の監視		実験店稼動確認チェック	ＳＰＬ、開発	・実験店稼動チェックリスト					
				実験店全ログチェック	ＳＰＬ、開発	・実験店チェックリスト					
				実験店全ログチェック結果レ	ＰＬ，ＳＰＬ	・レビュー報告書					
				全国展開		・ＰＧ交換障害報告書					
	13	ﾄﾞｷｭﾒﾝﾄ整理		ドキュメント整理、	開発部						
振返り・継承	1	開発完了作業		開発完了報告	開発部→業務部	・開発完了報告書					
				プロセスの測定、分析、改善、歯止めの成果確認	ＰＬ，ＳＰＬ	・プロセスの測定、分析、改善結果					
				開発予算との整合性チェック	ＰＬ，ＳＰＬ	・プロセス管理表に見積もり金額の記入					
				他プロジェクトへのノウハウの継承	開発部	・重要ノウハウの他部署への情報共有の実行 ・重要ノウハウのＤＢへの登録					

３．プロセスとマネジメントプラクティスの関連図：第１章　まとめ　３．、表１－まとめー１

		プロセス						
	マネジメントの役割	第１章 プロジェクトマネジメント	第２章 事前準備のプロセス （Preparation）	第３章 計画のプロセス （Plan）	第４章 実行のプロセス （Do）	第５章 管理のプロセス （Monitoring ＆ Controlling）	第６章 振り返りのプロセス （Retrospect）	第７章 ノウハウ継承のプロセス （Transfer）
マネジメント＆プラクティス	統合マネジメント（Integration Management） プロジェクトを統合的にマネジメントする	１－１．プロジェクトについて １－４．プロセスをマネジメントする		３－８．●統合的なプロジェクト体制の構築 ３－１０．プロジェクト計画書の作成 ３－ｈ１．プロジェクト統合管理			６－１．プロジェクト活動の振り返り ６－２．プロジェクト完了報告書	７－１．ノウハウの継承 ７－２．改善活動のすすめ
	ステークホルダーマネジメント（Stakeholder Management） ステークホルダーとの関係を良好に保つようにマネジメントする		２－１．顧客・ステークホルダー情報の収集					
	リスクマネジメント（Risk Management） プロジェクトへのリスクの影響が最小限になるようにマネジメントする	１－３．プロジェクトマネジメントにおける問題とリスク	２－２．未経験エリアへの対応準備 ２－３．事前準備工程のリスク	３－１．早期の仕様凍結 ３－５．●見積りにおけるリスク ３－８．●開発体制不備による失敗リスク	４－１．リスクの解消 ４－３．●開発管理表の概要			
	スコープマネジメント（Scope Management） プロジェクトへの要求事項を確実に達成するようにマネジメントする			３－１．早期の仕様凍結 ３－３．要求仕様の構造化（ＷＢＳ） ３－４．顧客要求の重要度順位の設定 ３－７．プロジェクト目標の設定		５－１．要求仕様の変更管理		
	品質マネジメント（Quality Management） 作業と成果物の品質目標を達成するようにマネジメントする			３－７．●品質目標値の設定	４－３．品質目標値の達成 ４－２．ＱＣＤ目標値の達成 ４－６．改善活動の実行	５－２．品質管理		
	コストマネジメント（Cost Management） コスト目標を達成するように予算とコストをマネジメントする			３－７．●コスト・利益目標値の設定 ３－５．見積り	４－４．コスト・利益目標値の達成 ４－２．ＱＣＤ目標値の達成 ４－６．改善活動の実行	５－３．コスト・利益管理		
	タイムマネジメント（Time Management） スケジュール目標を達成するように進捗をマネジメントする			３－７．●納期・生産性目標値の設定 ３－５．見積り ３－６．スケジュールの作成	４－５．納期・生産性目標値の達成 ４－２．ＱＣＤ目標値の達成 ４－５．納期・生産性目標値の達成 ４－６．改善活動の実行 ４－ｈ１．獲得時間の最大化と失う時間の最小化	５－４．進捗管理		７－４． ●改善活動のコンセプト コンセプト＃１．時間の獲得
	人的資源マネジメント（Human Resource Management） チームとしてパフォーマンスを良くするようにマネジメントする	１－２．プロマネの役割	２－ｈ１．開発者における自覚の喚起 ２－ｈ２．マネジメントの意識変革	３－８．●プロジェクト体制構築の要件		５－ｈ２．チームプレーの実行	６－ｈ１．失敗の原因を他に求めないこと ６－ｈ２．やる気を起こす～モチベーションの喚起	
	調達マネジメント（Procurement Management） 外部からの調達を円滑にするようにマネジメントする			３－８．●外注開発体制の構築 ３－９．開発環境の手配				
	コミュニケーションマネジメント（Communication Management） プロジェクトのコミュニケーションを円滑にするようにマネジメントする			３－８．●統合的なプロジェクト体制の構築◎統合管理および統合コミュニケーションの欠落 ３－ｈ１．●統合管理を阻害するもの		５－ｈ１．コミュニケーションの活性化		

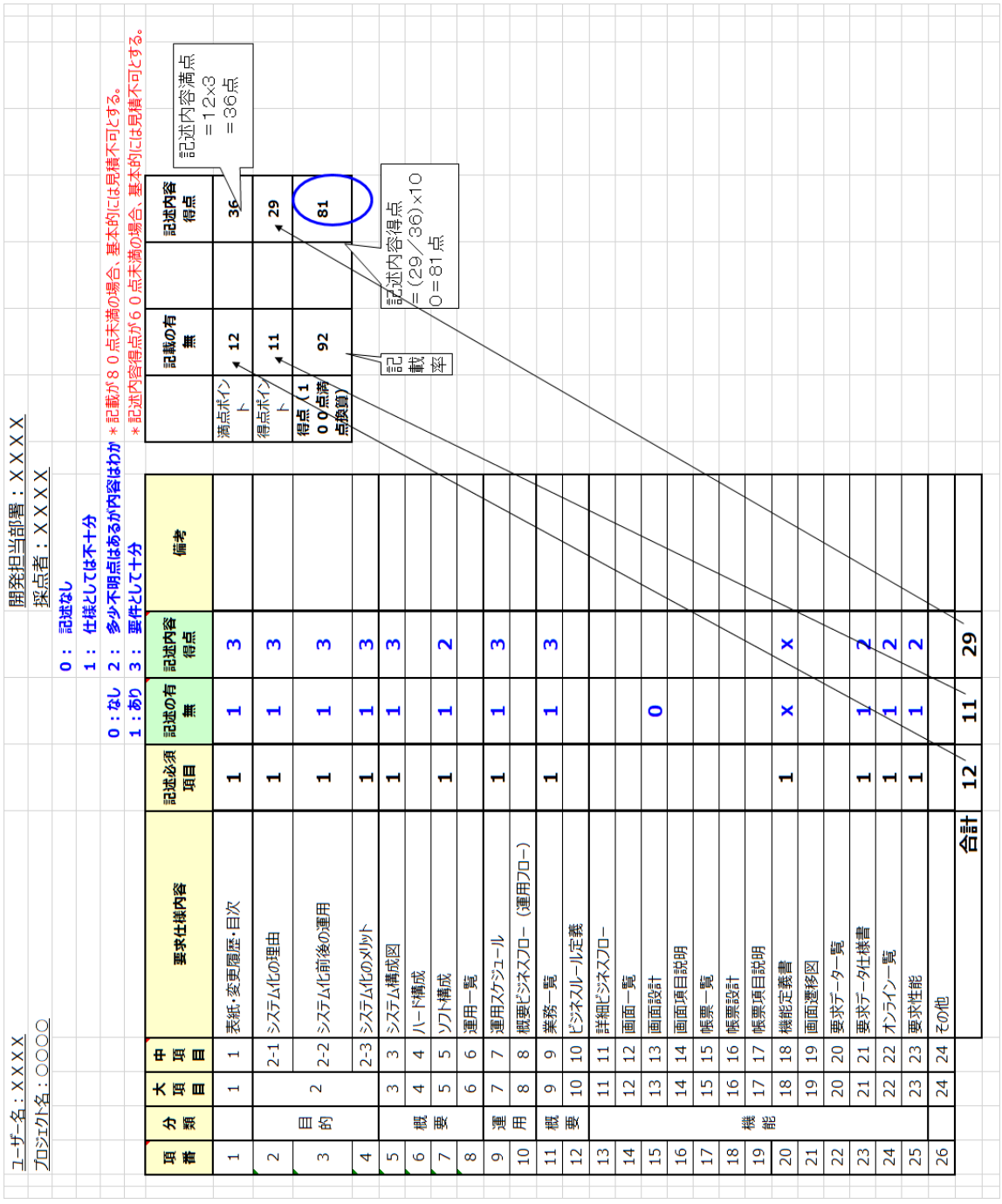
4．事前準備工程のリスク管理表：第２章　２－３．事前準備工程のリスク、表２－３－１

［リスク管理表（事前準備工程）］

プロジェクト名：　　初版発行日：　　更新日：　　文書番号：　　担当者名：

リスク要因分類	QCD影響 Q	C	D	事前準備工程リスク	✓ チェック 検出日	リスク管理項目 管理番号	リスク内容	対策アクション	対策日	振り返り
ヒトに関するリスク		○		①他者依存的姿勢（自律性の放棄）						
			○	・リーダーシップの欠如						
				・マルチベンダー下における分担責任のあいまいさ						
				・他社パッケージの未検証採用						
				・基幹技術の選定（なりゆきまかせ）						
				・開発組織の自律性不足						
		◎		②上位マネジメントの関与不足						
				・顧客交渉戦略・能力不足						
	◎	○	○	・不適切なプロマネの選任						
	◎	○	○	③ユーザーの参加・協力度不足						
				・顧客の参加・協力度が低い						
				・不適切な事前着手（発注前）						
			◎	④組織能力不足（未熟な組織文化、戦略の欠如）						
				・不適切な事前着手（開発量・時期の重複）						
			◎	・新旧システムの同時並行開発（開発量・時期の重複）						
				・顧客との各自だけの共同研究開発						
				・能力不足のプロマネの選任						
				・頻繁な変更による、スパゲッティ化						
	◎	◎	◎	⑤見積の能力						
				・安易な現行機能踏襲の保証						
	◎	○	○	・フィット＆ギャップ調査能力						
	◎	○	○	⑥要件定義能力						
				・要件定義能力不足						
				・顧客交渉戦略・能力不足						
				・不適切な事前着手						
				・新旧システムの同時並行開発						
				・企画能力不足						
				・適時の支援依頼発信能力不足						
				・問題の放置						
				・未経験言語採用の準備不足						
	◎		◎	・開発目的の誤り（受注優先）						
				・技術者のトレーニング不足						
				・開発体制の不備（経験者・能力・知識・人数）						
				・未経験の野ヂ業界参入の準備不足						
				・新技術採用の準備不足						
				・パッケージベース開発におけるフィット＆ギャップ調査不足						
				・タイムリミット管理能力不足						
				・技術方式選定対応における柔軟性						
				・オープンシステム知識の不足						
	◎		○	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）						
		○		・開発言語能力不足						
				・新技術知識不足						
				・顧客業務知識不足						
	◎	○	○	⑨プロセス管理の有無						
	◎	○	○	⑩コミュニケーション能力（阻害・ギャップ）						
				・顧客のパートナー化失敗						
				・顧客とのコミュニケーション能力						
	◎	○	○	⑪関連部署との連携不足						
				・組織間の協調性・コミュニケーション						
モノに関するリスク	◎	○	○	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）						
				・開発ガイドラインの不備（開発プロセス、設計手順書、コーディング規約、単体・総合評価テスト手順書等）						
	◎	○	○	⑬開発のベースの有無						
	◎	○	○	⑭開発環境の不備						
				・開発環境の不備						
カネに関するリスク	◎	◎	◎	⑮開発費不足						
				・赤字受注						
				・開発費不足（見積の失敗、開発の失敗）						
	◎	○	◎	⑯資源投入戦略の誤り（開発費投入時期ミス）						
				・資源投入戦略の誤り						
情報に関するリスク	◎	◎	◎	⑰あいまいな開発範囲（スコープ）						
	◎	◎	◎	⑱あいまいな要求仕様						
				・あいまいな要件						
	◎	○	◎	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）						
				・情報の不備・不足						
				・未経験の野ヂ業界の業務知識・技術情報不足						
				・新技術の技術情報不足						

5．要求仕様書　採点表：第３章　３－１．●要求仕様書の精度検証、表３－１－２



ユーザー名：ＸＸＸＸ
プロジェクト名：○○○○
開発担当部署：ＸＸＸＸ
採点者：ＸＸＸＸ

０：　記述なし
１：　仕様としては不十分
２：　多少不明点はあるが内容はわかる
３：　要件として十分

０：なし
１：あり

＊記述内容得点が６０点未満の場合、基本的には見積不可とする。
＊記述内容得点が６０点未満の場合、基本的には見積不可とする。

項番	分類	大項目	中項目	要求仕様内容	記述必須項目	記述の有無	記述内容得点	備考
1		1	1	表紙・変更履歴・目次	1	1	3	
2	目的	2	2-1	システム化の理由	1	1	3	
3	目的	2	2-2	システム化前後の運用	1	1	3	
4	目的	2	2-3	システム化のメリット	1	1	3	
5	概要	3	3	システム構成図	1	1	3	
6	概要	4	4	ハード構成				
7	概要	5	5	ソフト構成	1	1	2	
8	概要	6	6	運用一覧				
9	運用	7	7	運用スケジュール	1	1	3	
10	運用	8	8	概要ビジネスフロー（運用フロー）				
11	概要	9	9	業務一覧	1	1	3	
12	概要	10	10	ビジネスルール定義				
13	機能	11	11	詳細ビジネスフロー				
14	機能	12	12	画面一覧				
15	機能	13	13	画面設計		0		
16	機能	14	14	画面項目説明				
17	機能	15	15	帳票一覧				
18	機能	16	16	帳票設計				
19	機能	17	17	帳票項目説明				
20	機能	18	18	機能定義書	1	x	x	
21	機能	19	19	画面遷移図				
22	機能	20	20	要求データ一覧				
23	機能	21	21	要求データ仕様書	1	1	2	
24	機能	22	22	オンライン一覧	1	1	2	
25	機能	23	23	要求性能	1	1	2	
26		24	24	その他				
				合計	12	11	29	

	記載の有無	記述内容得点
満点ポイント	12	36
得点ポイント	11	29
得点（100点満点換算）	92	81

記載率

記述内容満点
＝12×3
＝36点

記述内容得点
＝（29／36）×10
０＝81点

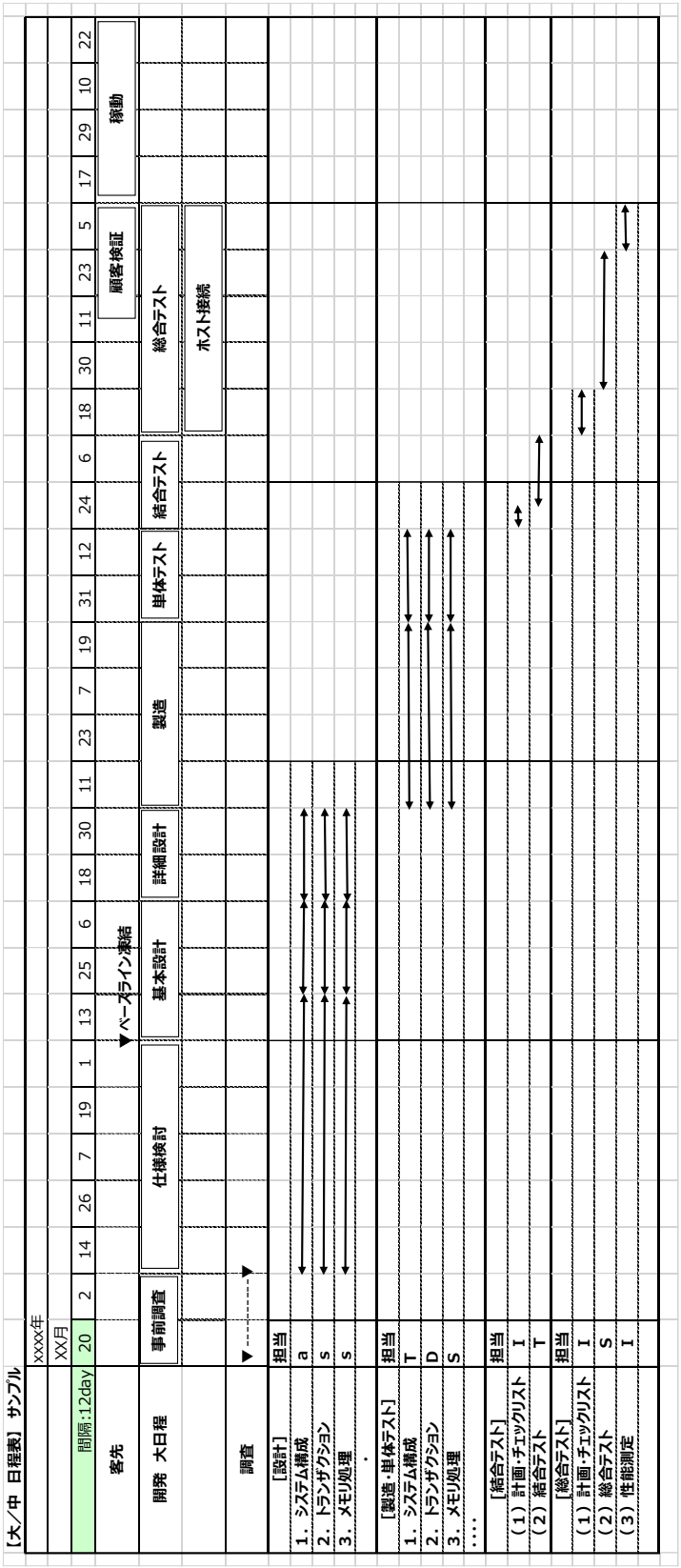
6．要件定義工程のリスク管理表：第３章　３－２．要求仕様に関するリスク、表３－２－１

【リスク管理表（要件定義工程）】

プロジェクト名：　　初版発行日：　　更新日：　　文書番号：　　担当者名：

要因分類		QCD影響			要件定義工程リスク	✓	チェック	リスク管理項目			
		Q	C	D			検出日	管理番号	リスク内容	対策アクション	対策日	振り返り
ヒトに関するリスク	①他者依存的姿勢（自律性の放棄）	○	○		・顧客の動き待ちの姿勢の結果、納期遅延							
	②上位マネジメントの関与不足			◎	・顧客交渉戦略、能力不足 ・大幅な仕様追加要求の丸呑み ・大幅な開発費削減要求の丸呑み ・大幅な納期短縮要求の丸呑み							
	③ユーザーの参加・協力度不足	◎	○	○	・顧客の参加・協力度が低い							
	④組織能力不足（未熟な組織文化、戦略の欠如）	◎	◎	◎	・見積りプロセスの手続き/ルール違反							
	⑤見積り能力	◎	◎	◎	・既存資産流用の可否判断 ・顧客要件の精査 ・フィット＆ギャップ調査能力							
	⑥要件定義能力	◎	○	○	・要件定義能力不足							
	⑦リーダーのプロジェクトマネジメント能力（外部交渉、タイムマネジメント、現場主義、見える化能力など）	◎	◎	◎	・顧客交渉戦略、能力不足 ・大幅な仕様追加要求 ・大幅な開発費削減要求 ・大幅な納期短縮要求 ・適時の支援依頼発信能力不足 ・変更管理未実施による開発費の増加 ・顧客に対する不都合な事実の隠蔽（故意による事故）							
	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）	◎	○	○	・要件定義体制不備（業務経験・知識不足の要員で要件定義を実施） ・要件定義遅延のため不十分なテストで客先リリース ・顧客要求の優先順位コントロール失敗による開発失敗 ・タイムリミット管理能力不足 ・新技術知識不足 ・顧客業務知識不足							
	⑨プロセス管理の有無	◎	○	○	・プロセス管理の不足 ・仕様凍結未確認のまま設計着手 ・基本設計の承認なしで詳細設計着手							
	⑩コミュニケーション能力（阻害・ギャップ）	◎	○	○	・顧客のパートナー化失敗 ・顧客とのコミュニケーション能力 ・プロマネと上司間のコミュニケーション ・営業と開発部門間のコミュニケーション能力 ・営業部との交渉力							
	⑪関連部署との連携不足	◎	○	○	・組織間の協調性・コミュニケーション							
モノ	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）	◎	○	○	・要求仕様書なし							
	⑬開発のベースの有無	◎	◎	◎								
	⑭開発環境の不備	◎	○	○								
カネ	⑮開発費不足	◎	◎	◎								
	⑯資源投入戦略の誤り（開発費投入時期ミス）	◎	○	◎								
情報	⑰あいまいな開発範囲（スコープ）	◎	◎	◎	・あいまいなスコープ							
	⑱あいまいな要求仕様	◎	◎	◎	・あいまいな要件 ・あいまいな仕様							
	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）	◎	○	○	・情報の不備・不足							

7．スケジュール表（大／中日程）：第3章　3－6．スケジュールの作成



8．設計工程のリスク管理表

：第４章　４－１．●開発現場にころがるリスクの事例、表４－１－１

【リスク管理表（設計工程）】

プロジェクト名：　　初版発行日：　　更新日：　　文書番号：　　担当者名：

	要因分類	QCD診断 Q	C	D	設計工程リスク	✓	リスク管理項目 チェック 検出日	管理番号	リスク内容	対策アクション	対策日	振り返り
ヒトに関するリスク	①他者依存的姿勢（自律性の放棄）	○	○	○	・協力会社への丸投げ（開発統合責任・製品最終責任の放棄）							
	②上位マネジメントの関与不足		◎									
	③ユーザーの参加・協力度不足	◎	○	○								
	④組織能力不足（未熟な組織文化、硬直の欠如）	◎	◎	◎	・オフショア開発							
	⑤見積能力	◎	◎	◎								
	⑥要件定義能力	◎	○	○								
	⑦リーダーのプロジェクトマネジメント能力（外部交渉、タイムマネジメント、現場主義、見える化能力など）	◎			・未経験技術者の大量投入							
		◎		◎	・基本設計レビューの未実施							
					・安易なフレームワークの改修							
	⑧メンバーの技術能力、ヒューマンエラー（うっかりミス）				・フレームワーク構築能力							
					・膨大系技術力の不足							
					・タイミング設計能力の不足							
					・排他処理設計能力の不足							
				◎	・手間排制御設計能力の不足							
		◎	○		・リトライ設計能力の不足							
					・UI技術力+認識不足							
					・オープンソフト対応技術力不足							
					・メモリーリーク対応技術力不足							
					・性能（パフォーマンス・レスポンス）技術力不足							
	⑨プロセス管理の有無				・プロセス管理不足							
		◎	○	○	・基本設計レビューの未実施							
					・性能要件（パフォーマンス・レスポンス）レビューの未実施							
	⑩コミュニケーション能力（用書・ギャップ）	◎	○	○	・組織間の協調性・コミュニケーション							
	⑪関連部署との連携不足	◎	○	○	・協力会社・オフショア先などとのコミュニケーション能力							
モノ	⑫ドキュメントの不備（要件定義書・設計書・チェックリスト・手順書など）				・要求仕様書なしの設計							
		◎	○	○	・資産なドキュメントやガイドライン							
	⑬開発のベースの有無	◎	◎	◎								
	⑭開発環境の不備	◎	○	○								
カネ	⑮開発費不足	◎	◎	◎								
	⑯資源投入範囲の誤り（開発費投入時期ミス）	◎	○	◎								
情報	⑰あいまいな開発範囲（スコープ）	◎	◎	◎	・あいまいなスコープ							
	⑱あいまいな要求仕様	◎	◎	◎	・あいまいな要件							
	⑲情報の不備・不足（マネジメント情報、技術情報、過去の失敗情報）	◎	○	○	・情報の不備・不足							

9．課題管理表：第4章　4－3．●課題管理表、表4－3－1

【課題管理表】

No	優先度	警告	発生日	期限	完了日	記入者	発生工程	課題項目	課題内容	課題対応策	担当	発生作業／進捗状況／結果	備考
1													
2													
3													
4													
5													
6													
7													

１０．ＤＭＡＩＣシート簡易版：第４章　４－６．●改善活動計画書、図４－６－１

【ＤＭＡＩＣシート簡易版】

承認印			

DMAICシート(簡易版)

テーマ：仕損費コストの削減

報告年月日　：

ＰＪリーダー名　：

ＰＪメンバー名　：

Define　＊問題の明確化

有るべき姿・こうありたいと思う状態

１．ソフトウェア製品における市場品質の向上

２．上流工程での品質確保

現在の状態

ＸＸ年下期仕損費　：xxxx千円

問題点（ＰＪが取り組む課題・テーマ）

１．デザインレビューだけでは上流工程での品質の確保は難しい。（プロセス及びリスクに関するレビューが不足。）

２．単純な不具合（プログラム製造段階での不具合）が相変わらず多い。

Measure　＊問題の分解と優先順位

一次分解	レビューが不足		製造不具合が多い	
二次分解	リスクの抽出が充分出来ない。（１）	開発プロセスの問題抽出が充分出来ない。（２）	請負外注先での単体テスト不足。（３）	外注受入検査が充分出来ていない。（４）

＊解決効果の大きい順に（１）～（４）

Analyze　＊問題を派生している根因の追及

１．注文書発行の遅延等で、本来のタイミングでの公式レビューが実施されておらず、問題点が発見されても後戻り出来ない工程になっている事が多い。又、短時間の公式レビューでは、開発プロセス及び リスクに関する細かなレビュ ーが出来ない為、公式レビューを補完する他のレビューが必要。

２．委託外注からの成果物受入検査を、総合テストで行っている為、総合テスト時に単体テスト不足による不具合が多発。又、外注より成果物を受入れる為の品質基準が不明確（数値基準が無い）であり、受入時のレビューが不足。

Improve

改善策

１．第３者によるプロセス監査・リスク監査の継続的実施。

２．外注受入の為の品質基準の作成 及び、品質基準に基づく外注受入検査・レビューの実施。

改善効果金額（年間）　：　Ｈ／Ｓ　：　xxxx千円

Control　＊改善を定着化するために行ったこと

プロセス監査・リスク監査を公式レビューと同様に開発プロセスの一部として定義付け、ＣＭＭにおけるＳＱＡ活動の一貫として監査を実施していく。

１１．仕様変更管理表：第５章　５－１．●仕様変更管理表がもたらす効果、表５－１－１

【仕様変更管理表】

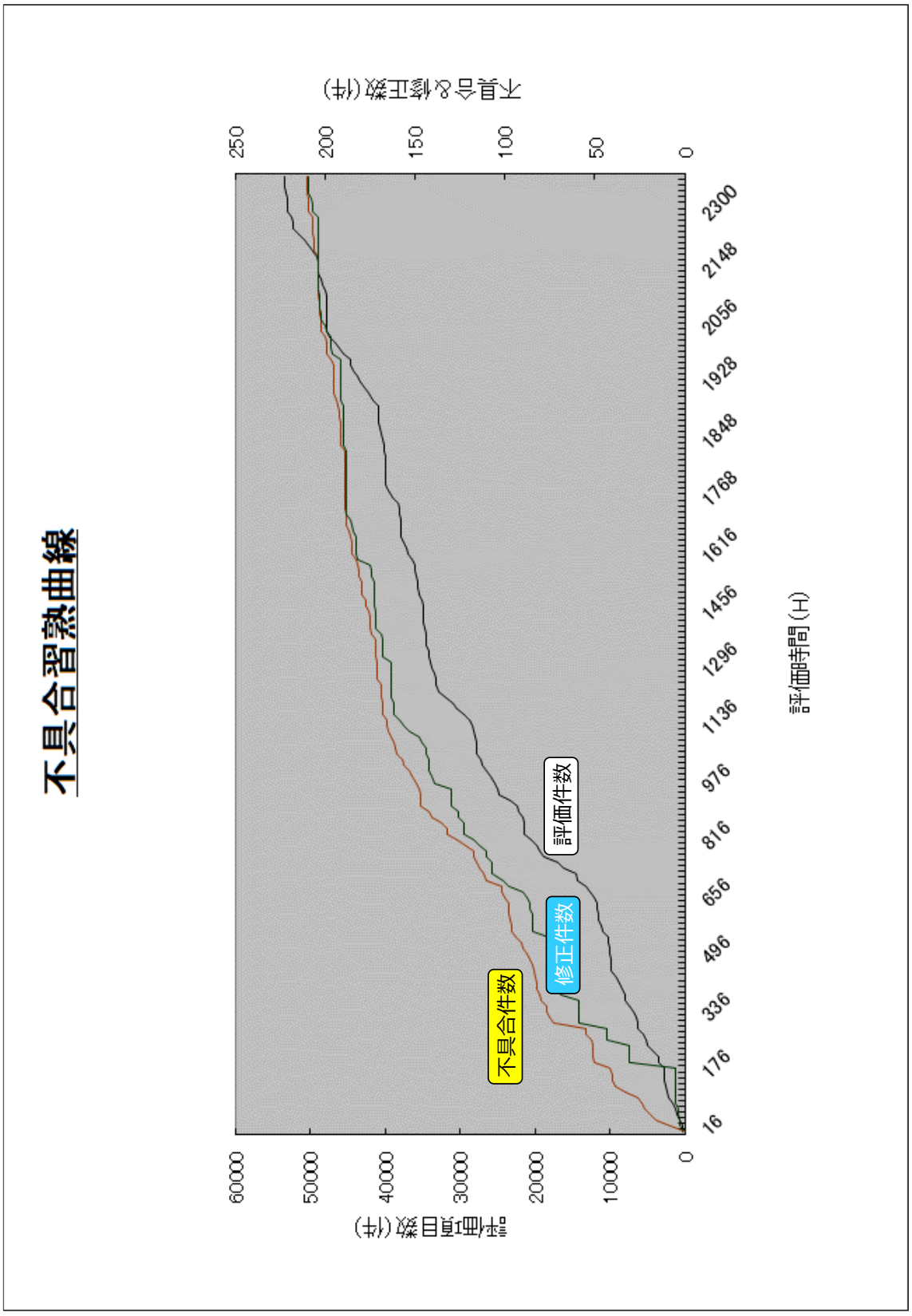
No.	管理番号	発生日	対応期限（ベースライン）	対応日	対応状況	業務名	変更内容	変更区分	修正担当	影響度範囲	難易度	リリースバージョン	対応工数（見込）	対応工数（実績）
1	PJ-001										A			
2											B			
3											C			
4														
5														
6														
7														
8														
9														
10														
合計														

１２．品質状況管理表：第５章　５－２．●品質状況管理表、表５－２－１

【品質状況管理表】

工程	業務名	プログラムステップ数	テスト項目数	不具合原因内訳				不具合件数合計	バグ密度（件数/kstep）	不具合深刻度レベル			テスト工数
				仕様バグ	設計バグ	製造バグ	管理ミス			A（重度）	B（中度）	C（軽度）	
単体テスト	A												
	B												
	C												
	D												
単体テスト合計													
結合テスト	A												
	B												
	C												
	D												
結合テスト合計													
総合テスト	A												
	B												
	C												
	D												
総合テスト合計													

１３．不具合習熟曲線表：第５章　５－２．●不具合習熟曲線表、図５－２－１



１４．**原価管理表**：第５章　５－３．●原価管理表、表５－３－１

【原価管理表】　売上高：xxxxx　＊%はすべて対売上高の数値

	プロジェクト名：		4月	5月	6月	7月	8月	9月	合計
予算	開発原価 予算	直接材料費 ①	外注費						
			購入品等						
			小計						
		直接労務費 ②	開発人件費						
			経費						
			小計						
		間接費③	開発間接費						
		予算合計 ①＋②＋③	金額						
			累計金額						
			累計%						
実績	開発原価 実績	直接材料費 ①	外注費						
			購入品等						
			小計						
		直接労務費 ②	開発人件費						
			経費						
			小計						
		間接費③	開発間接費						
		実績合計 ①＋②＋③	金額						
			累計金額						
			累計%						
（予算－実績）差額	開発原価 差額	直接材料費 ①	外注費						
			購入品等						
			小計						
		直接労務費 ②	開発人件費						
			経費						
			小計						
		間接費③	開発間接費						
		差額合計 ①＋②＋③	金額						
			累計金額						
			累計%						
	予算超過・消費不足の理由								

損益管理【予算合計】

	金額	%
売上高		
原価		
粗利		

損益管理【実績合計】

	金額	%
売上高		
原価		
粗利		

損益管理【差額合計（予算－実績）】

	金額	%
売上高		
原価		
粗利		

１５．機能モジュール進捗管理表：第５章　５－４．●モノの進捗管理、表５－４－２

業務	プログラム名称	ＥＸＥ，ＤＬＬ、ＯＣＸ等名称	開発Ｖｏｌ（Kbyte）		担当者	開発状況	コーディング（Ｃ）		単体評価（Ｕ）		結合評価（Ｋ）		不具合残件数	開発残件項目・処理日程			
			見込	実績			完了（予定日）	完了（実績日）	完了（予定日）	完了（実績日）	完了（予定日）	完了（実績日）		9月30日	10月10日	10月30日	備　考
	ＸＸＸ受信プロセス	ｘｘｘｘｘ.exe	50	60	A、B	C	6/30	6/30	8/15	8/15	8/30	8/30		△△△機能			△△△後の継続処理
	ＸＸＸ送信プロセス	ｘｘｘｘｘ.exe	45	40	A、B	C	6/30	6/30	8/15	8/15	8/30	8/30		△△△機能			○○○のパフォーマンス向上
	通信プロセス	○○.exe	25	30	A	U	6/30	6/30	7/15	7/15	7/30	7/30					
	リカバリー応答プロセス	○○RECOV.exe	35	30	A、B	U	6/30	6/30	7/15	7/15	7/30	7/30			○○○処理		スケジュールのステップ分け検討
	△△受送信プロセス	△△RECV.exe	25	20	A	K	5/30	5/30	6/15	6/15	6/30	6/30					
	△△送受信プロセス	△△SEND.exe	20	30	A	K	5/30	5/30	6/15	6/15	6/30	6/30					
	△△起動問合せ	△△INQ.exe	30	10	B	K	5/30	5/30	6/15	6/15	6/30	6/30					
	ＦＴＰプロセス	○○FTP.exe	20	30	C	K	5/30	5/30	6/15	6/15	6/30	6/30					

最小モジュール単位までブレークダウンする

量的把握の見込み／実績を行う

担当者の明確化

工程別進捗を予定／実績日程にて行い。色彩にても直感しやすくする

品質状況の把握を行う

残件処理日程を明確にする

備考記述による問題点のフォロー

１６．労働負荷状況管理表：第５章　５－４．●ヒトの消耗度管理、表５－４－５

【労働負荷状況管理表】

プジェクトメンバー氏名	先々月残業時間	先月残業時間	当月残業時間	３か月合計	１ヶ月平均残業時間	労働負荷レベル	労働負荷状況の理由	負荷軽減対策

労働負荷レベル

５：非常に過度：残業８１時間／月以上

４：過度：残業８０時間～４１時間／月

３：普通：残業４０時間～２１時間／月

２：軽度：残業２０時間～１１時間／月

１：非常に軽度：残業１０時間～０時間／月

１７．プロジェクト完了報告書：第６章　６－２．プロジェクト完了報告書

●仕様変更管理の考察、表６－２－１

仕様変更管理の考察														
No.	管理番号	発生日	対応期限（ベースライン）	対応日	対応状況	業務名	変更内容	変更区分	修正担当	影響度範囲	難易度	リリースバージョン	対応工数（見込）	対応工数（実績）
1	PJ-001	xx.09.31	xx.10.31								A			
2											B			
合計														
考察	1．対応工数の見込みと実績の差異の理由および今後の改善課題 2．仕様変更管理全般についての反省													

１７．プロジェクト完了報告書：第６章　６－２．プロジェクト完了報告書

●品質管理の考察、表６－２－２

品質管理の考察															
	テストフェーズ		不具合原因内訳				実績値		目標値	不具合深刻度レベル					
工程	プログラムステップ数	テスト項目数	仕様バグ	設計バグ	製造バグ	管理ミス	不具合件数	バグ密度（件数/kstep）	バグ密度（件数/kstep）	A（重度）	B（中度）	C（軽度）	テスト工数
単体テスト													
結合テスト													
総合テスト													
市場障害													
合計													
考察	1．バグ密度の実績値と目標値の差異の理由および今後の改善課題 2．不具合原因内訳・不具合深刻度レベルの結果についての反省 3．品質管理全般についての反省												

１７．プロジェクト完了報告書：第６章　６－２．プロジェクト完了報告書

●コスト管理の考察、表６－２－３

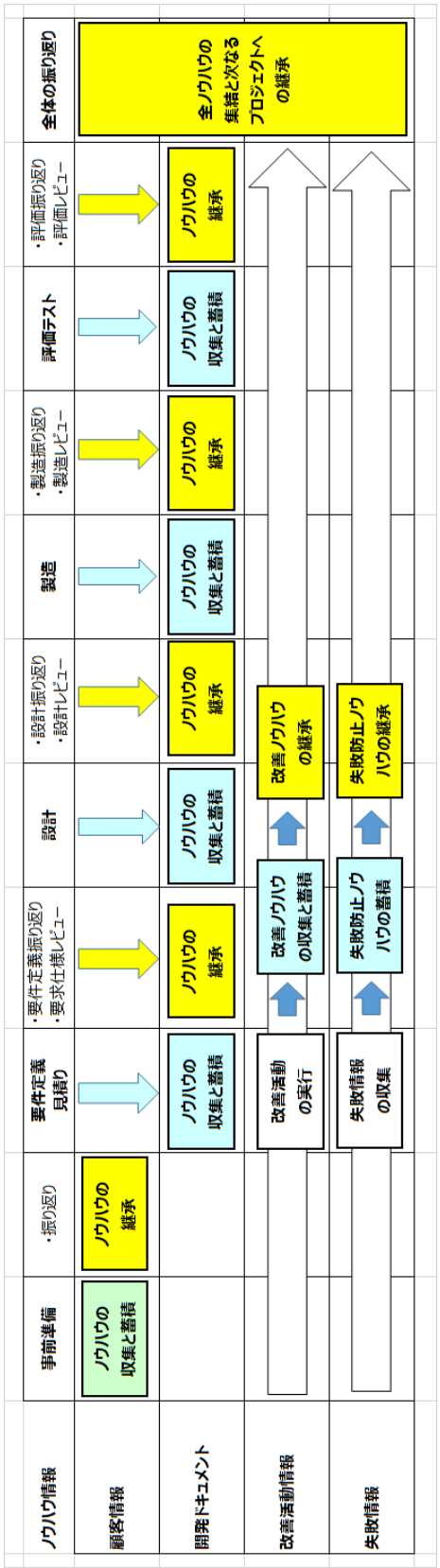
コスト管理の考察		予算	実績	差異	考察
直接材料費①	外注費				１．予算超過・消費不足の理由 ２．今後の改善課題 ３．原価管理全般についての反省
	購入品等				
	小計				
直接労務費②	開発人件費				
	経費				
	小計				
間接費③	開発間接費				
原価合計 ①＋②＋③					

●進捗管理の考察、表６－２－４

進捗管理の考察	各工程における進捗の目標と実績の差異の理由およびに今後の改善課題
要求定義	
概要設計	
詳細設計	
コーディング	
テスト	
進捗全般に関する反省	

１８．開発工程内におけるノウハウの収集･蓄積と継承のイメージ

：第７章　７－１．●ノウハウの継承はいつどこで行われるのか、図７－１－３



用語集

【あ】

アクティビティ　業務遂行に必要な活動のこと。（脚注１０、はじめにｐ４）

アジャイルソフトウェア開発宣言　アジャイルソフトウェア開発宣言は、アジャイルソフトウェア開発とその諸原則を公式に定義した文書である。この諸原則はウォーターフォール開発にも有意義なヒントを与えている。（脚注１１７、ｐ９０）

アプリケーション　特定の企業の業務にあわせて作成されるソフトウェアのこと。応用ソフトウェアともいわれる。（脚注３０、ｐ１４）

アプローチ　問題などへの取り組み方や方法のこと。（脚注９、はじめにｐ３）

粗利　売上高から製造原価を差し引いたもの。売上総利益とも言う。（脚注１１０、ｐ８１）

安定度　仕様変更の可能性の高さ低さのことを仕様の安定度と言う。（脚注８５、ｐ５４）

【い】

イベント　出来事、行事のこと。開発工程の節目ごとにおいて行われる成果物などの受け渡しのことを指す。（脚注２５、ｐ１１）

インクリメント　アジャイル開発などで逐次分割してリリースされる成果物をインクリメントな成果物と呼んでいる。アジャイル開発では、従来のウォーターフォール開発で行われているような、たくさんの要求仕様をまとめて開発・検証・リリースするのではなく顧客が要求する順に従って幾つかの仕様群に分割して順次開発・評価・リリースが行われる。（脚注１８６、ｐ２２７）

【う】

ウォーターフォール型開発　開発プロジェクトを時系列に、「要求定義」「概要設計」「詳細設計」「プログラミング」「テスト」「運用」などの作業工程に分割し、前から順次後へと開発を進める方式のこと。原則として前工程が完了しないと次工程には進めない。（脚注１８４、ｐ２２６）

運用フロー　客先におけるシステムの業務運用の流れを示したドキュメント。（脚注７８、ｐ５１）

【え】

影響度　仕様変更の影響が間接的に及ぶ他のソフトウェア部分を特定すること。仕様変更影響度表により管理される。（脚注７４、ｐ５０）

営業利益　売上高からすべてのコスト（製造原価・販売管理費）を差し引いたもの。（脚注１１２、ｐ８１）

エビデンス　証拠または根拠を示すもの。（脚注１５９、ｐ１６３）

【お】

オフショア（Offshore）　ソフトウェア開発に関する業務を海外の会社に委託する形態のこと。（脚注１１８、ｐ９２）

【か】

改善活動　ＱＣＤに関する問題の改善を図りプロジェクトを成功に導くと同時に、人材の育成を図る活動のこと。特に失敗に学ぶ活動は有効な結果を生む。（脚注２０、ｐ５）

ガイドライン　あるものごとにおいて実行すべき指針を示したドキュメント。（脚注７６、ｐ５１）

学習能力 失敗に学ぶ能力ないしは成長したいと思う意欲のこと。（脚注１８０、ｐ２１０）

カスタムソフト 顧客からの特別な要求仕様に基づいて開発されるソフトウェア。対語は市販品であるパッケージソフトウェア。（脚注９２、ｐ６２）

課題 現時点ですでに問題化してしまっているものごと。（脚注１２７、ｐ１０５）

課題管理表 すでに表面化している問題をその期限までに解決するための管理表。（脚注１２９、ｐ１０５）

課題バラシ 問題の発生原因と思われるものを可能な限り書き出し、類似したものを同一グループにまとめ、グループごとに有効と思われる対策を立案することで、複雑な問題に対する解決法を見出そうとする問題解決手法。（脚注１８５、ｐ２２６）

間接費 開発を間接的にサポートする開発管理部門等の費用。（脚注１０９、ｐ８１）

ガントチャート ガントチャートは日程表に進捗線を入れたもので、視覚的に分かりやすく大まかな進捗を管理するのには便利な管理表だと言える。（脚注１６１、ｐ１６９）

【き】

義侠心 正義を重んじて、強い者をくじき、弱い者を助けるような心もちのこと。（脚注１７３、ｐ１８７）

期待成果 達成したいと考えている目標数値。（脚注１３８、ｐ１１４）

機能モジュール進捗管理表 縦軸に開発タスクを最小レベルまでブレークダウンしたソフトウエアモジュール名を記載し、横軸にはそのプログラムの見込みサイズ、開発各工程の終了予定日・実績日を記入して予定日との差異を管理するもので、いわばＷＢＳ的なモノの進捗管理ないしは開発物の単品管理ともいえる。（脚注１６２、ｐ１６９）

キーマン プロジェクトの成否のかなめとなる中心人物。 キーパーソンとも言う。（脚注１３７、ｐ１１２）

許容時間 ある仕事を実行するために許される一定の時間。（脚注１１４、ｐ８４）

業務品質 人間が行う業務能力の品質。例：作業ミス率等。（脚注１０１、ｐ７６）

【け】

原価管理表 原価低減の施策が適切に実行されているのかを判断し素早い是正措置を実行することでコスト・利益の目標の達成を確実にするために使用される管理表。（脚注１６０、ｐ１６５）

原価志向型 コスト目標の達成を中心とする考え方。売上げ拡大の志向が弱くなる。（脚注１４４、ｐ１２６）

【こ】

構成管理 ソフトウェア開発におけるソースコードや設計書などの成果物に対して変更履歴を管理し、必要に応じたバージョンの成果物を取り出すことを可能にする。一般的に構成管理ツールが用いられ、ビルド管理、リリース管理、バージョン管理などを通して成果物の一貫性を保つようにしている。（脚注１０２、ｐ７６）

工程別分業方式 ソフトウェア開発の各工程を別々の会社で分業して担当している方式。（脚注１１９、ｐ９２）

行動規範 人間社会集団におけるルールや慣習のこと。（脚注１７２、ｐ１８７）

コード体系 JAN・EAN・UPC などの商品コード。（脚注７７、ｐ５１）

顧客価値 顧客がビジネス上重要視している機能やシステムのこと。（脚注６８、ｐ４７）

個人戦 組織に拠ることなく、個人単独で問題解決に当たること。（脚注１４７、ｐ１３６）

コストセンター あらかじめ用意された予算により運営される組織のこと。利益確保の直接的な責任は負わされない。（脚注５４、ｐ３５） ⇒参照「４－４．コスト・利益目標値の達成●プロフィットセンターとコストセンター」

コストマネジメント 達成すべき開発コストおよび利益の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。（脚注４０、ｐ２２）

コミュニケーション 聞く・話す・読む・書くなどの能力を使って、お互いの理解・合意・納得を得る双方向性の行為のこと。（脚注４６、ｐ２６）

コミュニケーション・ギャップ（Communication Gap） 組織間・工程間で発生するコミュニケーションの断絶のこと。（脚注１２４、ｐ９９）

コミュニケーションマネジメント １．人的な情報系の中枢を司ること。２．何をどうすべきかを明らかにし開発者全員で共有すること。３．プロジェクトの共通の目標に向かって開発者全員のベクトルを揃えること。（脚注４５、ｐ２６）

コンセプト ものごとの基本的な概念を言い表したもの。ものごとの骨格を表わす考え方。（脚注１３９、ｐ１１５）

コンティンジェンシープラン(Contingency Plan) 予期せぬ事態に備えて、予め定めておく緊急時対応計画・不測事態対応計画。（脚注１３３、ｐ１１２）

コントロール ＱＣＤ等の目標値のモニタリングにおいて目標未達が予測された場合、原因の特定および是正措置を行う。（脚注１５５、ｐ１５６）

【し】

時間ラグ 時間差のこと。（脚注１６８、ｐ１８３）

思考・行動様式 人や組織における、その成長過程において習得された、特有なものの考え方および行動。（脚注４８、ｐ３１）

自転車操業 自転車はこぐのを止めれば倒れるのと同じで、効果がないことをいつまでも続けざるを得ないような状況のこと。（脚注１７８、ｐ２０３）

システムインテグレータ 情報システムの開発において、コンサルティングから設計、開発、運用・保守・管理までを一括請負する情報通信企業。略して SIer（エスアイヤー）とも呼ばれる。（脚注１４６、ｐ１２６）

集団行動主義 同じ目的に向って、全員でそれぞれの役割分担に従い、協調した行動をとること。（脚注１６７、ｐ１７８）

仕様骨子 要求仕様の内、その中心となる重要な仕様。（脚注６４、ｐ４３）

仕様調査 要求仕様の疑問点や不明点を調査すること。この作業の質が見積りの精度に大きく影響する。（脚注７３、ｐ５０）

仕様凍結 顧客の要求内容を仕様化し顧客の同意を取り付けること。基本的な仕様を設計着手前に凍結することがプロジェクトの成功要因の一つとなる。（脚注２１、ｐ６）

自律性 直面する問題に対して自分自身の目で確かめ、状況を把握・判断し、実行に移す能力のこと。反対語は他人依存性。（脚注２４、ｐ９）

自律的思考 自分自身でものごとを直接観察し、その意味するところを判断すること。（脚注１１６、ｐ９０）

深層リスク ヒト・組織文化に由来する行動や仕事のやり方に起因するリスクのこと。（脚注１３２、ｐ１１１）

人的資源マネジメント １．プロジェクトのＱＣＤ目標達成に必要な人材を社内・社外に求め、プロジェクト体制を構築すること。２．開発者およびマネジメントの自覚を喚起し、モチベーションを維持すること。３．相互義務の履行、相互扶助の実行を通してチームプレーを行い、プロジェクトのパフォーマンスの最大化を図ること。（脚注４２、ｐ２４）

【す】

スクラムハーフ フォワードとバックスのつなぎ役で、俊敏性と判断力が求められ、小柄な選手でも活躍できる。（脚注１７５、ｐ１８８）

スコープ プロジェクトが開発すべき要求内容およびその範囲のこと。（脚注３７、ｐ２０）

スコープマネジメント 要求仕様の内容および開発範囲を明らかにし、顧客と同意しておくこと。 スコープの明確化は見積りの妥当性確保、ＱＣＤ目標値の達成に必須の要件となる。（脚注３８、ｐ２０）

ステークホルダー プロジェクトを取り巻く利害関係者のこと。（脚注３４、ｐ１８）

ステークホルダーマネジメント 顧客をはじめとしたステークホルダーからＱＣＤ等の情報を収集し、早期の仕様凍結・妥当な見積り・開発リスクの解消を通して、顧客の信頼を獲得すること。（脚注３５、ｐ１８）

スローガン（slogan） 目的や理念を表わしたイメージしやすい標語やキャッチコピーなど。（脚注９９、ｐ７４）

【せ】

生産性 時間当りに生み出される成果物の量および質を表わす指標。（脚注１９、ｐ４）

製造原価 直接材料費、直接労務費、間接費など開発チームが開発に要した費用（コスト）。（脚注１０６、ｐ８１）

製品品質 物自体の品質。例：ソフトウェアの不具合発生率等。（脚注１００、ｐ７６）

先行着手 仕様未凍結状態で、開発側の想定仕様に基づいて開発を始めること。（脚注６９、ｐ４８）

戦略的思考 計画を成功させるための方策を考えること。（脚注１３４、ｐ１１２）

【そ】

相互義務 プロジェクト関係者がお互いに果たすべき義務のこと。（脚注１７７、ｐ１９２）

相互扶助 プロジェクト関係者がお互いの不足を補い助け合うこと。（脚注１７６、ｐ１９２）

想定開発　確定仕様に基づかず、自分の想定仕様による開発。多くの場合、やり直しが発生する。（脚注７０、ｐ４８）

組織戦　複数の人間で組織を形成し、それぞれに役割分担を決め、協調・連携のもとに問題解決に当たること。（脚注１４８、ｐ１３６）

損益管理表　プロジェクトにおける売上高と費用（コスト）を対比して、その差額として利益を示すもの。一般的には損益計算書と呼ばれている。（脚注１０５、ｐ８０）

ソフトウェアハウス　ソフトウェアの受託開発やソフトウェアパッケージの開発を主要業務としている企業。（脚注１４５、ｐ１２６）

ソフト性能　ソフトウェアの動作速度、特に応答速度（レスポンス）および業務処理速度（パフォーマンス）は重要なソフトウェアの性能要件とされる。（脚注９３、ｐ６５）

【た】

ターゲットマシン　ファームウェアを搭載するハードウェア機器。（脚注１２２、ｐ９４）

タイムマネジメント　達成すべき納期および生産性の目標値を設定し、その目標を達成するための改善活動を伴った開発業務を遂行すること。（脚注４１、ｐ２３）

多重請負　一つのプロジェクトを元請・下請け・孫請けというような多段階の請負構造で形成していること。（脚注１２５、ｐ９９）

タスク　アプリケーションソフトにおける最小の業務単位。コンピュータが処理する仕事の最小単位。（脚注８８、ｐ５８）

単品管理　モノの無駄をなくすために、商品を一品ごと、販売・仕入・在庫の管理を行う手法。コンビニの単品管理は好事例。（脚注１６３、ｐ１６９）

【ち】

チームプレー　共通の目的を持った複数の人間集団がその目的を達成するために相互に連携した活動を行うこと。（脚注１７０、ｐ１８６）

チェックリスト　致命的な手順抜けの防止を目的として、失敗事例を手短にまとめたリスト。（脚注１３、はじめにｐ５）

調達マネジメント　１．社内人材における不足を補うために適切な人材を外部から採用し、外注開発体制を構築すること。２．開発の遂行に必要な機器・機材を適切な時期に適量を購入するための手配を行うこと。（脚注４４、ｐ２５）

直接コミュニケーション　実際に顔と顔を突き合わせた対面コミュニケーションのこと。ダイレクトコミュニケーション、フェイス トゥ フェイス コミュニケーションとも言われる。（脚注４９、ｐ３１）⇒参照「５－ｈ１．コミュニケーションの活性化●直接コミュニケーションの重要性」

直接材料費　外注費および開発ツールや備品等の購入品に要した費用。（脚注１０７、ｐ８１）

直接労務費　開発人件費および旅費・交通費・通信費・消耗品等経費に要した費用。（脚注１０８、ｐ８１）

【て】

データドリブン開発 データを根拠にした開発活動のこと。（脚注１１５、ｐ８６）⇒参照「４－２．ＱＣＤ目標値の達成●データドリブン開発とは何か」

データ移行 旧システムで使用していたデータを新システムに乗せ換えること。（脚注８３、ｐ５１）

デッドロック 行き詰ってしまった状態。いわゆる、にっちもさっちも行かない状態のこと。（脚注１３６、ｐ１１２）

【と】

当事者意識 他人に寄りかかることなく自分の役割をまっとうするという心構えのこと。（脚注６１、ｐ４０）

統合コミュニケーション 開発工程ごとに担当会社や担当者が異なる場合に発生するコミュニケーションの溝を埋めるための計画的な情報共有会議の実施等。（脚注１２０、ｐ９２） 参照⇒「３－ｈ１．プロジェクト統合管理」

統合プロジェクトマネジメント 工程別・機能別に分離・分業化された開発業務における情報分断の隙間を埋め、開発のプロセスを統合的にまとめあげる業務のこと。（脚注３２、ｐ１４）

統合マネジメント プロジェクトマネジメントの基本的な９つのマネジメントプラクティスである、ステークホルダーマネジメント、リスクマネジメント、スコープマネジメント、品質マネジメント、コストマネジメント、タイムマネジメント、人的資源マネジメント、調達マネジメント、コミュニケーションマネジメントを相互に結び付け一つの統合されたプラクティスとして働くようにすること。（脚注３３、ｐ１６）

ドキュメント 仕様書や設計書などに限らず管理用文書、ビジネス書類、メモなども含む全ての文書のこと。（脚注７２、ｐ４９）

ドキュメントベース開発 口頭だけに依存せず、書類・書面に基づいて実行される開発。（脚注１４０、ｐ１１５）

【に】

ニーズ 要求ないしは要望のこと。顧客のニーズが漏れなく要求仕様書に記述されていることを確認する手段としては、顧客との仕様打ち合わせ議事録や顧客から提供される提案依頼書（ＲＦＰ：Request For Proposal）および顧客への提案書などがある。（脚注８４、ｐ５４）

日次情報共有会議 プロジェクトメンバーにて毎日行われる短時間の情報共有会議。（脚注６２、ｐ４１）

認定仕様 その顧客において過去に十分な稼働実績がある、いわゆる枯れた仕様のこと。（脚注８６、ｐ５５）

【ね】

ネゴシエーション 交渉力のこと。特に見積り交渉においては、強い交渉力が必要とされる。（脚注６５． ｐ４３）

【の】

ノウハウ 業務の遂行において必要な知識や経験などの中でとりわけ重要なものの総称。（脚注１２、はじめにｐ４）

ノウハウデータベース ノウハウを集めたデータベースのことで、関係者なら誰でもいつでも容易に利用可能なシステム。（脚注１８２、ｐ２１７）

ノルマ 半強制的に与えられた時間制限付の労働の基準量。（脚注１７１、ｐ１８６）

【は】

バグ密度 作成されたソフトウェアの１ｋｓｔｅｐ当たりに発生したバグの数で表される。（脚注１５６、ｐ１５７）

派生開発 新規開発後のいわゆるバージョンアップ開発のこと。（脚注４、はじめにｐ１）

白紙委任状 ものごとの決定権を相手に委ねてしまうことの例え。（脚注１５３、ｐ１５４）

販売管理費 開発チーム以外の管理・販売・研究部門等に要する費用（コスト）。（脚注１１１、ｐ８１）

バッチ情報 後でまとめて処理される情報のこと。一方、即時的な処理はリアル処理と呼ばれる。（脚注１６９、ｐ１８３）

パフォーマンス 組織における開発処理能力のこと。またはソフトウェアの処理速度のことを指す。 あるソフトウェアに起動をかけてから全ての処理が完了するまでにかかる時間。処理速度とも言う。（脚注４３、ｐ２４）、（脚注８２、ｐ５１）

【ひ】

ビジネスロジック アプリケーション機能における処理方法の論理のこと。（脚注７９、ｐ５１）

非機能 非機能要件とは、性能や信頼性・拡張性・セキュリティなど、機能要件以外のもの全般を指す。（脚注８０、ｐ５１）

必要時間 ある仕事を実行するために必要な時間。実行する人や組織の能力により長短の差が出て来る。（脚注１１３、ｐ８４）

表層リスク 要件定義・設計書・開発費などの不備や不足などのモノやカネに関するリスクで表面化しやすいリスクのこと。（脚注１３１、ｐ１１１）

品質マネジメント 業務品質および製品品質を数値化し、達成すべき品質の目標値を設定し、品質目標を達成するための改善活動を伴った開発業務を遂行すること。（脚注３９、ｐ２１）

【ふ】

ファクター ある結果を生み出す要因。（脚注５０、ｐ３１）

ファームウェア ファームウェア(firmware)とは、電子機器に組み込まれたコンピュータシステム（ハードウェア）を制御するためのソフトウェアで、ソフトウェアを ROM 等の集積回路にあらかじめ書き込まれた状態で、機器に組み込んだもの。（脚注１２１、ｐ９４）

フィールドサービス部門 市場にて稼働するシステムなどの保守・整備部門。（脚注４７、ｐ３０）

フォーカス 焦点を当てること、ないしは明確に表現すること。（脚注２９、ｐ１３）

フォーマット 書式および構成があらかじめ設定されている文書のこと。（脚注７５、ｐ５１）

フォワード ラブビーにおいてスクラムの最前列を守る巨漢・屈強な選手。（脚注１７４、ｐ１８８）

不具合習熟曲線表 信頼度成長曲線またはゴンペルツ曲線などと呼ばれるもので、横軸にテスト時間、縦軸に累積バグ発見数をとったグラフで表される。Ｓ字の成長習熟曲線を描くことが多く、決定論的モデルとして、現在の状況から今後の予想を立て、テスト進捗管理、バグ収束率の予測、残バグ数の予測などに用いられる。（脚注１５７、ｐ１５８）

プラクティス 業務活動（アクティビティ）を構成する具体的な実践内容のこと。（脚注１１、はじめにｐ４）

プラットフォーム アプリケーションソフトウェアを支える、ＯＳおよびミドルウェアなどの基幹ソフトウェア。（脚注５６、ｐ３６）

プロジェクト 特別な目的を達成するための期間限定の臨時的な組織のこと。（脚注３、はじめにｐ１）

プロジェクトマネジメント 獲得したリソース（人・モノ・カネ・情報・時間）の制限内で、プロジェクトのＱＣＤ目標をすべて達成し、プロジェクトを成功に導くこと。（脚注２、はじめにｐ１）

プロジェクトリーダー プロジェクトを統率する筆頭責任者のこと。プロマネや開発リーダーなどを束ねる統括マネージャのこと。筆頭プロマネの別称。（脚注２２、ｐ６）

プロジェクト計画書 開発実行の行動計画書となるもので、プロジェクトをスムーズに運営し、ＱＣＤ目標を達成するために必要な活動内容を計画書という形で明文化したもの。プロジェクト計画書には、見積り内容、プロジェクトのスコープ、スケジュール、コスト、リスク、ステークホルダーとのコミュニケーション方法などの記載が必要。（脚注１２３、ｐ９５）

プロジェクト統合会議 ステークホルダーとの定期的な情報共有会議。（脚注５９、ｐ３７）

プロセス ある目的に従った結果を得るために、入力情報を一定の論理に従って加工する処理手順のこと。（脚注５、はじめにｐ１）

プロセス管理表 開発における各工程の流れに沿って、開発の主要なイベントおよび成果物を時系列順に並べた管理表で、各工程における実行手順忘れ・手抜き行為などを防止するためのもの。（脚注２７、ｐ１２）

プロセスサイクル 開発の始まりから終了までの一連の流れを、複数のプロセスに分割して表したもの。（脚注２８、ｐ１３）

プロトタイプ（prototype） ソフトウェア開発において、本番プログラムの作成に先立って新技術・新機能の検証および問題点の洗い出しのために仮に作成されたプログラムのこと。（脚注６０、ｐ３８）

プロトタイピング プロトタイプを作成すること。（脚注６６、ｐ４４） 参照⇒「プロトタイプ」

プロフィットセンター 独立採算により運営される組織のこと。利益確保の責任を持つ。（脚注５５、ｐ３５） ⇒参照「４－４．コスト・利益目標値の達成●プロフィットセンターとコストセンター」

プロフィットドリブン開発 利益目標の達成を主眼として、同時にコスト目標の達成を実現する開発。（脚注１４１、ｐ１１５）

プロマネ（プロジェクトマネージャ） プロジェクトマネジメントの手法を用いてプロジェクトを運営する責任者のこと。（脚注１、はじめにｐ１）

【へ】

ベクトル　力がかかる方向のこと。（脚注１６６、ｐ１７７）

ベースライン　見積り回答によって顧客と約束した開発費・開発スケジュール・品質条件および開発仕様の内容・範囲のこと。（脚注１５２、ｐ１５２）

ベンダー（vendor）　ソフトウェアないしはハードウェア製品のメーカーまたは販売会社のこと。ユーザー企業から発注されるソフトウェア開発の元請け企業となる場合が多い。（脚注６７、ｐ４６）

【ほ】

ボトルネック（Bottle　neck）　進行の障害や妨げとなるもの。（脚注１３５、ｐ１１２）

【ま】

マネジメント　ものごとを適切に運用し、所定の成果が生み出されるように組織を運営すること。（脚注１７、ｐ３）

【め】

メンテナンス性　仕様変更の容易性が高いこと。いわゆるスパゲッティプログラムは、低メンテナンス性の代表例。（脚注１５１、ｐ１４６）

【も】

目的　ある行為を行う動機や理由を示したもの。「～のために」と表現されることが多い。（脚注９７、ｐ７４）

目標　目的を達成するための具体的な手段を示したもの。「～によって」と表現されることが多い。（脚注９８、ｐ７４）

元請け　顧客から仕事を最初に受注する企業のこと。いわゆる１次ベンダーのこと。（脚注２３、ｐ８）

モニタリング　ＱＣＤ等の目標値の達成推移状態を記録・監視すること。（脚注１５４、ｐ１５６）

【ゆ】

優先順位ベース開発　顧客価値の高い要求仕様の開発を優先させ、不要不急な開発を避けること。（脚注１４２、ｐ１１５）

【よ】

要素部品　ディスプレードライバー、プリンタードライバーなどのＩＯ系デバイスをコントロールするソフトウェアなど。（脚注５７、ｐ３６）

予算進捗管理表　各工程の予算と実績の乖離状況を把握することで資金面におけるプロジェクトの健全性をチェックする管理表。（脚注１６４、ｐ１７２）

【ら】

ラップアップミーティング　開発完了時に行われる、開発行為の全体を振り返る会議。（脚注１３０、ｐ１０７）

【り】

リーダーシップ　メンバーたちの先頭に立って道を切り開き、障害物を取り除き、成功を阻害する者たちの盾となり、進むべき方向を指し示し、チームのメンバーが最大限の力を発揮できるようにすること。すなわちプロジェクトを統合管理すること。（脚注１２６、ｐ１００）

リードタイム　開発に要する期間を示したもので、開発開始・終了等の時期は明示されていない。（脚注９６、ｐ７０）

リスク　現時点では問題として現れてはいないが問題として現れる可能性のあるもので、プロジェクトを失敗させるすべての要因のこと。（脚注１４、はじめにｐ５）

リスクヘッジ（Risk　Hedge）損失・失敗の防衛策のこと。（脚注５１、ｐ３２）

リスクマネジメント　開発プロセスに内在するリスクを早期に発見し、解消することでプロジェクトのＱＣＤ目標を達成させること。（脚注３６、ｐ１９）

リスク管理表　そのプロジェクト特有の想定リスクおよび開発組織で繰り返し起こしている障害やミスなどの過去の失敗を中心としたリスクを記載した管理表で、開発の工程ごとのリスク管理表が必要とされる。（脚注１２８、ｐ１０５）

リソース　開発の実行に必要な人材・資材・資金・情報・時間などの有形無形の資源のこと。（脚注１５、ｐ２）

リリース　完成したソフトウェアおよび関連成果物を顧客へ引き渡すこと。（脚注９１、ｐ６０）

稟議書　りんぎしょ　プロジェクト遂行の許可申請書のこと。通常その事業の損益見込み計算書が添付されている。（脚注５３、ｐ３５）

【る】

ルーチンワーク　手順や手続きが決まっている定型作業のこと。（脚注１６、ｐ２）

【れ】

レスポンス　ソフトウェアの応答速度。たとえば、あるキーを押してから表示が行われるまでにかかる時間など。（脚注８１、ｐ５１）

レビュー　工程毎の成果物に対して、その成果物が意図された通りに作成されたかどうかを検証し誤りを発見する行為のこと。（脚注１５８、ｐ１５９）

レビュワー　レビューをする人。反対にレビューをしてもらう人はレビューイと呼ばれる。（脚注１８３、ｐ２１８）

連携　連絡提携という言葉を略したもので、連絡を密に取り合って、一つの目的のために一緒に物事をすること。（脚注５８、ｐ３６）

連帯　共通の目的に向って、共に責任を分担して行動すること。（脚注１８１、ｐ２１６）

【ろ】

ロードブロック　元の意味は道を塞いでいる障害物のこと。先に進むためにはその障害物を片付ける必要があると言うような文脈で使用される言葉。（脚注９０、ｐ６０）

労働負荷状況管理表　開発者の人的な消耗度を毎月の時間外労働時間によって管理するもの。（脚注１６５、ｐ１７４）

【わ】

ワークパッケージ ＷＢＳ（作業細分化構造書）の最小単位の仕様のこと。
（脚注８９、ｐ５８）

【英・数字】

ＡＢＣ分析 障害内容を顧客の視点から見て、その深刻度のレベルを高いものの順に、Ａ：重度障害、Ｂ：中度障害、Ｃ：軽度障害と定義し、その発生件数・率などによって品質の傾向を把握するためもの。
（脚注１０３、ｐ７７）

ＣＭＭＩ（Capability Maturity Model Integration、能力成熟度モデル統合版） プロジェクトの成功指標であるＱＣＤを担保するプロセス（工程）がどの程度組織に定着しているのかを示す国際標準モデル。レベル１から５までの５段階で評価される。レベル５が最高レベル。
（脚注６３、ｐ４２）

ＤＭＡＩＣ Define（定義）、Measure（測定）、Analyze（分析）、Improve（解決策）、Control（管理）のステップからなる経営変革手法であり、VOC（Voice of Customer、顧客の声）を基にして事業活動を分析し、データドリブンでプロセスの改善を進める。
（脚注１４９、ｐ１３８）

ＦＰ（ファンクション・ポイント） ソフトウェアの機能数および複雑度による重みづけをした点数の合計から開発工数を見積る方式。（脚注９５、ｐ６６）

Human Activity（人間的活動） 人間性（情緒性）に関する人間力発揮の活動のこと。
（脚注８、はじめにｐ３）

ＩＰＡ・ＳＥＣ 独立行政法人情報処理推進機構（IPA）技術本部ソフトウェア高信頼化センター（SEC）の略称。（脚注１４３、ｐ１２１）

ＩＳＯ 国際標準化機構。ISO 9001 は、品質マネジメントの国際標準規格。ISO21500 は、プロジェクトマネジメントの国際標準規格。（脚注２６、ｐ１２）

Job Activity（実務的活動） 実務性（合理性）を必要とする技術的な実務活動のこと。
（脚注７、はじめにｐ３）

ＬＯＣ（Line of Code） ソフトウェアの規模を表す指標のひとつで、ソースコードの行数を意味する。
（脚注９４、ｐ６６）

ＯＳ オペレーティングシステム（Operating System）。コンピュータの基本的な動作を制御する、システムソフトウェア。（脚注３１、ｐ１４）

Ｐ．ドラッカー（Peter Ferdinand Drucker） 経営学者。『マネジメント』はその代表的な著書。
（脚注１５０、ｐ１４２）

ＰＭＢＯＫ（Project Management Body of Knowledge）プロジェクトのマネジメントに必要なガイド、手法、メソドロジー、ベストプラクティスをまとめた国際的に標準とされているプロジェクトマネジメントの知識体系のこと。（脚注６、はじめにｐ１）

Q&A Question＆Answer 質疑応答の略。ここでは仕様の不明点および疑問点に関する質疑応答のこと。（脚注７１、ｐ４９）

ＱＣＤ Quality 品質、Cost コスト、Deliverly 納期、の頭文字をとった略語で、プロジェクトの成功指標とされている。（脚注１８、ｐ３）

ＲＡＳＵＩ 優れた製品の５つの特性である、Reliability：信頼性（不具合・障害率の低さ）・Availability：可用性（継続的稼動能力）・Serviceability：保守性（保守・メンテのしやすさ）・Usability：使用性（使いやすさ）・Installability：設置の容易性、の略称。（脚注１０４、ｐ７８）

ＳＥ システムエンジニア（System Engineer）の略称。本書では、顧客の要件を要求仕様にまとめる職種として使用。（脚注５２、ｐ３４）

ＷＢＳ（Work Breakdown Structure、作業細分化構造書） 要求仕様の全てを細分化とともにツリー構造化したもの。最上位は納品されるシステム名等で、順に成果物の大項目・中項目・小項目に細分化される。最小単位はワークパッケージと呼ばれる。（脚注８７、ｐ５８）

5W4H「５Ｗ：Why、What、Who、When、Where ４Ｈ：How to、How many、How much、How long」の略で、５つのＷで始まる疑問詞は「誰が・いつ・どこで・何故・何を」したのか、つまり行為の当事者・時間・場所・動機（理由）・対象物を明確にすることでその行為の意味を明らかにする。また４つのＨは、その行為を実現するための手段・方法・程度などの科学的データを明らかにする。これらは人の行為の全体像を特定する良い方法である。（脚注１７９、ｐ２０６）

チェックリスト一覧表

索引

【へ】

【ほ】

著者プロフィール

佐野洋（さのひろし）
現在，フリーコンサルタントとして PM ファクトリーを主宰。
数十年にわたり POS システムのファームウェア開発およびプロマネ業務に従事。
好きな言葉は「Boys be ambitious!」。
信条は「弱き者も強き者も共にその生涯を生き抜くこと」。

コンタクト

eMail：pmf_hsano@yahoo.co.jp
URL：https://pmfactory-hsano.jimdofree.com/

お問い合わせ

本書に記載されている内容についてのみのお問い合わせとさせていただきます。
またお問い合わせにつきましては、eMail：pmf_hsano@yahoo.co.jp 宛てにお願いいたします。
なお、ご質問の際には、書名、該当ページ、氏名、返信先を明記していただきますようお願いいたします。
お送りいただいたご質問には、可能な限りお答えできるように努力いたしますが、当方にて不適切だと判断されるご質問には回答を差し控えさせていただく場合もあります。あらかじめご了承のほどお願いいたします。

著作等

『SE 稼業は忘己利他（もうこりた）―現場に転がる箴言集』 技術評論社ｗｅｂサイトにて連載
URL：http://gihyo.jp/dev/serial/01/engineer-proverbs

CONTENTS