ITプロジェクト

ソフトウェア開発のメソッド

クイック・リファレンス

PMファクトリー　2016年

目次

はじめに

本書は、ソフトウェア開発において発生するさまざまな問題について、その真因や解決策のポイントを短時間に把握するためのリファレンス・マニュアルとして作成したものです。そのために個々の問題についての背景や説明は意図的に省略しており、問題の解説については各章の最後にまとめて「問題の解説」として記述しました。

本書の構成は、ソフトウェア開発における主要な局面を15分類し、それぞれを一つの章にまとめた形となっています。

第1章　緊急事態対応のメソッド
第2章　見積りのメソッド
第3章　要求仕様のメソッド
第4章　納期問題のメソッド
第5章　時間不足解消のメソッド
第6章　目標設定のメソッド
第7章　コミュニケーションのメソッド
第8章　プロジェクトマネジメントのメソッド
第9章　設計・製造のメソッド
第10章　ドキュメントのメソッド
第11章　単体・結合テストのメソッド
第12章　モチベーションのメソッド
第13章　総合・運用テストのメソッド
第14章　本質を見極めるのメソッド
第15章　人材育成のメソッド

各章においては、その問題に関する代表的な問題の原因や対応方法について基本的に箇条書き的な記述方式をとっています。

第1章　緊急事態対応のメソッド

メソッド＃1　【緊急対応時の行動の原則】

①（観察）　まずメンバーを集め、データを集めること。
②（判断）　次に集めたデータを分析・統合しその意味を判断すること。
③（決定）　対策案を決めること。
④（実行）　最後に対策を打つこと。

因みにこの方法は緊急時だけではなく平時における問題解決の手順と同じですが、違うのは緊急対応ですからこのプロセスを全員で短時間に実行しなければいけないということです。リアルタイムでの情報共有と調査分析の担当割りなどをリーダーの差配の下に素早く実行する必要があります。②に最も時間がかかるという点にご注意下さい。またこのプロセスは何度も回す必要がある場合もあります。

前記の①～④の行動原則はOODAループ理論として知られています。O：Observation（観察）、O：Orientation（判断・方向性の確認）、D：Decision（進むべき方向の意思決定）、A：Action（行動・実行）。

メソッド＃2　【モグラたたき状態からの脱出法】

①リーダーは直ちに全員を集めること。
②各人が保有する情報を全て一箇所に集めること。
③問題となる情報を整理分類し、可能な限り仕事の全体像（最終目標）を見えるようにすること。
④最終目標と思われるものに対して自分たちがどの地点まできているのかを想定して見ること。
⑤最終目標と乖離している残件項目を明らかにし、必要な工数の概算を見積ること。
⑥各担当の残件負荷のバランスを比較し極力平準化を図り、役割を再配分すること。現有勢力で対応が不可能と判断したら必要な人材と人数を新たに投入すること。
⑦毎日の短時間情報共有会議を行い、各人において①今日やったこと　②明日やる予定　③今抱えている問題の三点につき報告をおこない、リーダーは適時アドバイスを行うこと。

肝心なことは、このような状態に陥らないで済むように、開発の準備段階から顧客との密接なコミュニケーションを通して、顧客の要求の全体像を早い時点で把握し、上記に挙げた七項目の活動を開発初期から継続して行うことに尽きます。

第2章　見積りのメソッド

1.　見積り問題のメソッド

メソッド＃1 【見積精度向上の三つのポイント】

①仕様に関する知識に習熟すること。
②無理・無駄・リスクを排除する改善活動の実行。
③使用技術のレベルアップ。

これらの活動の積み重ねが精度の高い見積りとなり、顧客を納得させられるデータの裏づけとなります。

メソッド＃2 【見積りチェックリスト】

☐ 見積り回答期限の事前確認は行ったか。
☐ 見積りガイドラインを使用して見積りを行ったか。
☐ 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。
☐ 仕様の不明点は全て顧客に確認したか。
☐ 仕様の疑問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。

その他、過去の自他における見積り失敗項目をチェックすること。過去の見積りミスについてもこのチェックリストに加えればさらに有効なものになるでしょう。他の見積り担当者たちとの情報共有を進めてください。

メソッド＃3 【見積り回答の原則】

①見積りの使用目的を確認すること。
例;個人的な勉強のため／仕事に直結しない参考値のため／顧客要求のため／など。
②口頭での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こと。
③顧客の正式要求のものならば、要求日および妥当な回答希望日を記入した見積り依頼書を両社の正式なルートを通して、要求仕様書と共に提出していただくよう伝えること。
④正式要求書の場合、回答期限に間に合いそうもないと判断された場合は、その理由を明確にし、時間を置かず直ちに要求者と期限の交渉を行うこと。

メソッド＃4 【概算見積りのやり方】

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こと。
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こと。
④それぞれに対して、必要工数を大雑把に 3 段階（大・中・小）に分けて記入します（5 段階；1～5 でも良い）。各段階に要する工数は、例えば、大＝1 ヶ月、中＝2 週間、小＝1週間等に決めておきます。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・小を記入しておきます。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と良いでしょう。
⑤これらの仕事を何人で実行するかを想定しておきます。
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開発期間でしょう。

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。

2. 見積り問題の解説

1） 開発におけるコスト構造について

見積りに関する問題はすべて、適正な利益を獲得するためにはどのような見積りを行うべきかという一点に集約することができます。

開発の損益は次の算式で表すことができます。
損益＝受注額－開発費 （受注額はほぼ見積り回答額と同額としておきます）。
即ち、受注額＞開発費の場合に黒字となり、逆の場合に赤字となります。
黒字幅を大きくするためには、可能な限り受注額を大きくし、開発費を小さくするしかありません。

受注額は、オフショアも含めた市場の相場によって制限を受けます。必要なコストに利益を上乗せしたものをそのまま見積り額として提示して受注できた時代（コストドリブン）は昔のことです。現在において、価格は市場の競争が決めるマーケットドリブンの時代になっています。
過激なマーケットドリブンな経済環境の中においては、より一層の低価格が求められ、受注額は低下の一途をたどっています。このような状況下において利益を出すためには、より一層のコスト削減、すなわち開発費の削減が求められています。
因みにソフトウェア技術者料金（東京地区）の 2002 年から 2012 年までの 10 年間の推移は、PMで－20．0%、SE1で－17．8%、SE2で－14．2%、PGで－9．6%の下落となっています。（出典：経済調査会「積算資料」）

開発費すなわちコストは次の算式で表すことができます。
開発費＝直接開発費＋間接経費＋失敗等によるロス
開発費を削減するためには、直接開発費・間接経費・ロスの削減を行うしか方法はありません。それぞれのコスト要因を決定しているものは次の通りです。
①直接開発費は、開発組織の能力、すなわち生産性によって決まります。
②ロスは、開発組織の能力不足、すなわち失敗修復のための損失コストによって決まります。
③間接経費は、開発組織の間接人員経費および設備費等によって決まります。

上記三つのコスト項目において開発チームが開発活動の中で直接的にコントロール可能な項目

は、下記の 2 点です。
①直接開発費の低減、すなわち開発組織の生産性の向上。
②ロスの低減、すなわち失敗等に起因した手戻り作業の削減による品質の向上。

生産性および品質の向上は、仕様理解力および技術力の向上の二つによって実現可能であり、更にこれらを可能にするためには開発者における改善活動が必須の要件となります。
受注額の下落が続く中において開発企業が利益を出し、生き残る方法は、開発者賃金の大幅カットか改善活動のどちらかまたは両方の実施しかないと言っても過言ではないでしょう。
「忙しくて改善活動をやっている暇などない」という開発組織は間違いなく消滅する運命にあります。開発チームは、何がなんでも、改善活動を実行することで、生産性向上および品質向上を継続的に実現しなければならない義務と使命が課せられています。

2） 見積り回答書に対する認識について

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。

3） 受注者にとっての見積り回答書の重要性について

受注者の最大の目的は、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要件は以下の二件だけです。
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。

第3章　要求仕様のメソッド

1.　要求仕様問題のメソッド

メソッド＃1 【顧客との仕様打ち合わせにおける三つのポイント】

①要求仕様の全体像を把握すること。
②顧客要件の優先順位・価値の重要度順を把握すること。
③不明点・疑問点を全て払拭すること。

メソッド＃2 【仕様凍結のポイント】

①最初に要求仕様の概要と全体像を把握すること

要求仕様の全体像を把握するために最初に仕様項目および要求概要を顧客価値の高いもの順に並べ整理しておく。

②仕様凍結は顧客価値の重要度順に

仕様を全部一辺に決めようとしないで、顧客価値の重要度順に凍結していくこと。

③仕様凍結は直接コミュニケーションで

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て、こちらからも対案を出し、集中検討会や合宿で一気に仕様決定を行うこと。従来のように相手が決めてくれるのを待っている姿勢では何も変わりません。

④開発着手は顧客価値の高い仕様の順に

仕様が先に凍結された顧客価値の高いものから開発着手を行うこと。

これらの仕様凍結のやり方を顧客・ベンダー・下請け三者における共通原則とすることで、いつまでたっても決まらない仕様問題やあいまいな仕様問題などが解消できるでしょう。

メソッド＃3 【仕様凍結にあたっての注意点】

仕様凍結の遅れは開発工程の遅れや品質の悪化を生み、次の開発の仕様確定の遅れを呼ぶと言うような悪魔の連鎖を生みます。このような悪魔の連鎖を断ち切るためには、所定の時期までに仕様を絶対確定させるという使命感が発注者側・受注者側の両者において必要です。最も効果的な仕様凍結方法は、両者において仕様凍結最終日を明確に約束した上で合宿や集中検討会を開き、両者が直接顔を突き合わせて仕様が凍結するまで絶対にやりぬくということです。メールや電話のやりとりなどで仕様凍結しようなどと思ったら、今まで通りいつまでたっても仕様凍結などできないと覚悟しておいた方がいいでしょう。

基本的な行動姿勢は、「誰が」「何を」「いつまでに」を明確にすることです。

メソッド＃4 【仕様問題の解決策】

①仕様凍結の最終期日を切り、お互いに合意しておくこと。
②顧客価値の高い仕様からの仕様凍結および開発を行うこと。
③あいまいな点・不明確な点・矛盾点などについて直接コミュニケーション・電話によるヒアリング・Q＆Aシート等による確認を設計開始前に終了させること。
④開発の初期工程においては要求元担当者との密接なコミュニケーション（週2～3回）を重ね仕様変更・追加の有無を確認することで後工程での頻繁な仕様変更や仕様凍結の遅延を防ぐこと。
⑤仕様の条件やノウハウである要求仕様の背景や理由について必ず確認し文書化すること。
⑥経験不足な部分は経験者への質問や相談を行うこと。

⑦待ちの姿勢ではなく、こちらからも積極的に仕様提案を行うこと。
⑧仕様提案力をつけるためにベンダー側のSE・顧客打ち合わせ等へ参加すること。

メソッド＃5 【仕様理解に関する注意点およびポイント】

①まずは仕様の全体像の把握から始めること。
②要求仕様調査時に疑問点・不明点の発掘を行い、不明点解消に向けて要求元に対して積極的なアプローチを取ること。
③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。
④不明仕様のQ＆Aは直接対話による確認・決定を行うこと。
⑤不明なことは直ちに分かっている人に確認すること。
⑥要求仕様の背景や意味を必ず明確にしておくこと。
⑦習得した仕様知識を、ドキュメントによって他のメンバーに伝えること。
⑧早期の仕様凍結を行うこと。
⑨基幹仕様未決定で開発に着手してはいけない。

メソッド＃6 【検討・調査時間の使い方の3分割法】

あるものごとに関する検討・調査をするにあたって最初にすべきことは、この検討・調査で許される時間や日数がどれ位なのかを決めておくことです。次に許された時間を3等分し、最初の1／3の時間や期間でまず仕事の全貌を把握します。把握の途中で不明点は一旦そのままにしておき、赤印などで印をつけておき、全体を見終わったところで、印をつけた箇所について①全く分からないもの、②短時間の検討・調査で分かる項目、③検討・調査に長時間かかりそうなもの、に3分類し、①についてはすぐに依頼者に質問を出したり、有識者に聞いたりします。
次の1／3の時間で②と③の検討・調査を行いますが、ある程度の時間をかけても分からなければ、その時点で②③について依頼者に質問を出したり有識者に聞いたりします。不明点の解消はこのサイクル内で完了するように心掛けることです。

最後の1／3の時間で依頼者からの要求と自分が検討・調査した結果が整合性のあるものになったかどうかを確認することです。

実際の検討・調査時間は仕事の難易度や量によって異なってくるでしょう。上記はあくまでも検討・調査時間・期間だけについての話であり、開発全体の時間・期間に関するものではありません。

【ポイント】

①許容時間の把握。
②許容時間を3等分する。
③最初の1／3の時間で、調査対象のおおまかな全体像を把握する。
④次の1／3の時間で、詳細の調査・検討を行う。
⑤最後の1／3の時間で、要求内容と調査結果の整合性の確認を行う。

メソッド＃7 【要求仕様書のチェックポイント】

過不足のない要求仕様書の条件として、IEEE830は下記の品質特性を定義しています。

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

また自然言語で記述される要求を仕様化する方法「USDM（Universal Specification Describing Manner）」における具体的な記述方法をIEEE830と対比させたものが次に示した「要求仕様書のチェックリスト」です。

要求仕様書の作成者においてはこれらの要求仕様書における基本的な品質要件を満足させる必要があり、一方要求仕様書の受領者においては受け取った要求仕様書の品質レベルを確認することに役立つでしょう。またこのチェックリストは要求仕様書のみならず設計書を初めとしたあらゆる開発ドキュメントの品質レベルの指針としても活用できるでしょう。

【要求仕様書のチェックリスト】

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こと。使用する場合は、〇月〇日までに決めるとコメントをつけること。

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
□ 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、範囲をあらわしていることを意識して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を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。

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・分離して表現している。つまり、要求が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）

2. 要求仕様問題の解説

（注. 件数データおよび%は本問題に関する過去の相談データから抽出したもの。）

1） 決めない・決まらない要求仕様問題について

この問題の約70%はベンダー側の問題であり、その主な原因は、ベンダー側における開発能力の低下にあり、二次的な原因としてはベンダー側の能力低下をカバーできない下請開発側の開発能力不足にあると考えられます。ここで言う開発能力はソフトウェア技術力のみならず仕様決定能力やプロジェクトマネジメント能力などソフトウェア開発に必要なすべての能力を含んだものです。

ソフトウェア開発の初期の時代においては、ベンダー自身の手によって開発が行われていましたが、ソフトウェア・ハードウェア技術の進化および顧客ニーズの深化・多様化の流れの中で、構築されるソフトウェアシステムの巨大化・複雑化が進み、やがてベンダー単独ではシステム構築が困難となり、ソフトウェア製造の外注化が行われるようになりました。初期の段階ではベンダー側において仕様決定および設計を担当し、下請側においてプログラム製造という分担が行われていましたが、現在はさらに設計までも下請側にシフトされ、製造に関してもオフショアにシフトされるような状況になっています。

このような状況下で、ベンダー側においては設計・製造の能力を低下させ続けており、ソフトウェア開発の基盤能力から遠ざかったベンダー側技術者たちは外注管理事務者になってしまった感があります。さらにベンダー側技術者たちにおいては、要求仕様の決定は、SEの仕事であるという思い込みがあり、仕様決定能力も設計・製造能力も失ってしまったかのようです。これが現在のベンダー側技術者たちが、仕様を迅速に決定できない理由だと思われます。

◎ベンダー側における問題点

ベンダー側における問題点を整理すると下記のようになります。

①設計・製造能力の低下
②プロジェクトマネジメント力の低下
③仕様知識の低下
④外注依存の傾向

◎ベンダー側にて必要な行動

ベンダー側に必要な改善点は、”統合的開発能力の獲得”であり、具体的には次のようなことになります。

①仕様ノウハウを学習・蓄積すること。
②一定の割合で設計・製造を実行し、開発技術力を保持しておくこと。
③外注まで含めた統合的プロジェクトマネジメントを実行すること。

◎下請け側における問題点

指示された通りにプログラムだけを製造すれば良かったという長年の習慣からの脱皮ができなかった下請け会社の場合は、ベンダー側において衰弱した設計能力やプロジェクトマネジメント能力を補完することができず、オフショア勢力に仕事を奪われることになってしまいます。

下請け側における問題点を整理すると下記のようになります。

①設計能力不足
②プロジェクトマネジメント能力不足
③仕様知識不足

下請け側に必要な改善点は、”自律的開発能力の獲得”であり、具体的には次のようなことになり

ます。
◎下請け側にて必要な行動
①仕様ノウハウを学習・蓄積すること。
②設計能力を進化させること。
③製造能力を進化させること。
④評価能力を進化させること。
⑤プロジェクトマネジメントを実行すること。

ベンダー側および下請け側双方にて上記の改善施策を一致協力して実行することで「決まらない仕様」の問題を解決していく必要があります。

2） あいまいな要求仕様問題について

提起されたあいまいな要求仕様問題の原因別の順位は次の通りでした。
①仕様の全体像・意味・背景・理由・範囲が不明であること　6件
②コミュニケーション経路・方法問題　5件
③知識・能力不足　4件
（注. 件数データは要求仕様に関する過去の相談データから抽出したもの。）

上記内容から読み取ることができる文脈は、開発すべき仕様の意味・背景・理由はもとよりその全体像も明らかでないままに、多層の組織間を通したコミュニケーション経路により、更に一層開発すべきものの姿が本来のあるべき姿から遠のき、開発者たちの憶測や想定に頼った開発が行われているのではないかという恐ろしい現実があると言うことです。この問題は、最初の「決められない仕様」問題との重層的問題となっており、誰が考えてもこの状況では成功するプロジェクトなどほとんどないのではと思われます。「何のために」が分からない仕様では、「どのように設計・製造したら良いか」も分からず、「どのようにテストしたら良いか」も分からないのは当たり前のことです。
この”あいまいな要求仕様問題”に対する処方箋は下記のようになります。

【あいまいな要求仕様に対する処方箋】

①要求仕様の背景・意味・理由・全体像の明確化
ベンダー側は、要求仕様書の最初のページに、その仕様が必要になった背景・理由、及びその仕様の意味・全体像についての説明を必ず記述すること。ベンダー側の要求仕様書にこの記述がなければ、下請け側は強く記述を要請する必要があります。
②同時・集中的仕様検討会の実行
要求仕様の凍結に当たっては上位組織から下位組織に向かって、一段ずつ伝達を行うのではなく開発関係全部署の担当者が一同に会して仕様決定を行うことを原則とすること。
③過不足のない要求仕様書の作成および確定内容に基づく開発の実行
想像・想定によらず、要求仕様書の記述に基づいた開発を原則とすること。
上記を実行しない限り、あいまいな仕様問題はいつまでたっても解決しないでしょう。

3） 仕様の間違い・誤解問題について

この問題の原因別順位は次の通りでした。
①問題意識の不足　4件
②システム全体からの視野の不足　1件、常識力不足　1件、情緒的な思考　1件

仕様の読み間違いや誤解の原因の半分以上は、仕様そのものに対しての疑問点や不明点を発見する力の不足にあります。またその他の3件に関しても自分の頭で考える力の弱さに原因があります。この問題は、要するに、ものごとを自分の目でしっかりと観察し、自分の頭でその状況を判断するという自律性欠如の問題です。この自律性を阻害する最大の要因は、仕事における時間がないと言う”あせり”にあるものと思われます。

あせりの気持ちは、とにかく仕事をこなさなければならないという切迫した状態に人を追い込み、たとえ目の前にある要求仕様書に欠陥があったとしても、とりあえず大丈夫だと言うことにしておき自分の疑問に封印をしてしまう作用をもたらしているのでしょう。何も考えずに流れ作業的に仕事をこなして失敗したという問題はまさにこの問題に他なりません。さらにこのあせりの気持ちが高じてくると、必要な作業手順や工程までもスキップしてしまうというような大きな過ちをおかしてしまいます。あせりをなくすためにはあらゆる時間を失う行為を排除する必要があります。時間不足問題の解決なくして開発問題の解決なしと言っても過言ではないでしょう。

これに関するピーター・ドラッカーの箴言を紹介しておきます。

「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。」（ピーター・ドラッカー）

4） 先行着手の弊害について

提起された問題は、すべて仕様決定遅延による時間不足に起因した問題でした。この問題は、仕様の間違い・誤解問題における”あせり”と同様の問題です。前項をご参照ください。

第4章　納期問題のメソッド

1.　納期問題のメソッド

メソッド＃1 【スケジューリングのポイント】

①全ての工程において、割り込み作業や不測の問題は必ず発生するという認識を持つこと。
②全ての工程のスケジューリングは必ずバッファを持たせること。
③バッファを設けるためには、常に業務の改善や効率化を行い、各作業を前倒しできるようにすること。全てうまく行くという前提で組んだギリギリのスケジューリングは全く非現実的であり必ず破綻します。

メソッド＃2 【スケジュール遅延の原因】

①実行すべき開発内容が不明確。仕様凍結が遅延しており、仕様追加・変更が続いている。
②見積り時における開発期間が短すぎる。開発作業に必要な日数の見積り間違い。仕事量の見誤りや発注者側の押し付けなどの結果、本当は実現できないスケジュールを組んだり、並列に開発できない仕事までも無理やり並列スケジュールで組んでいる。
③リーダーが開発仕様の全体像を押さえていない。
④メンバーが担当仕様を十分に理解していない。
⑤各作業者への仕事の割振りが曖昧、不均等、不適切、能力レベルに応じた分担になっていない。
⑥予定目標と実績進捗が日々管理されていない。

プロジェクトのリーダーおよび主要メンバーに何故頻繁にスケジュール遅延が起きるのかヒアリングしてみれば、上記の内のどれが現在の主要問題なのか判明するでしょう。

メソッド＃3 【開発遅延防止の対策】

①仕様の早期凍結。
②適切な見積りによる妥当な開発期間・予算の獲得。
③割り込み作業（障害対応、見積り依頼、など）への緊急度・優先度を考慮した対応。
④改善活動の実施による、プロジェクト遂行能力・仕様理解力・技術能力の継続的な向上。
⑤日次情報共有会議の励行によるチーム内での情報共有。
⑥他人や他部署との約束納期を常に意識し、仕事はある程度早めに終了させておくこと。
⑦各作業の前倒し完了を可能にするための効率化や改善活動を実施すること。

メソッド＃4 【納期問題の解決法】

①見積り交渉時における開発内容に見合った工期・予算の確保。
②主要仕様の適切な時期までの凍結。
③開発効率化のための普段の改善活動の実行。
この 3 点を可能にすれば正しい開発、良い品質、納期の確保、利益の創出は自然に達成される。

メソッド＃５ 【仕事における優先順位の設定法】

①この仕事の意味・意図・背景を最初の時点で正確に把握すること。
②仕事内容をブレークダウンすること。
③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。
④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。
⑤時間的余裕があるものは重要機能の順に処理すること。
⑥優先順位が判断できない場合は早めに上長に相談すること。
⑦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。

メソッド＃６ 【仕事のスピードを落す真因】

①仕事内容を完全に把握しておらず、不明点や疑問点を放置していること。
②仕事内容を実行するために必要な知識の不足や仕事の進め方についての能力不足。
③現在の自分の能力の把握不足。
④仕事の質・量の見積り間違い。
⑤他のメンバーとのコミュニケーション不足による情報不足。

メソッド＃７ 【仕事のスピード向上の基本】

①事前準備をおこなうこと。仕事にとりかかる前に、その仕事の内容の調査や全体像把握、その仕事の段取りの計画、その仕事のリスクの把握などを実行する。
②仕事の最中においてはQCDの数値データを記録しておくこと。また人との関係性の問題点についても文章として残しておく。
③振り返りの実行。仕事が終了した時点で、仕事中で起こしたミス、失敗についての真因を把握し再発防止策を立て、次の仕事に備える。
④以上のことをすべて記録しておき、ものごとを数値で語れるようにしておく。

メソッド＃８ 【割り込み要求への誠意ある対応】

①その場で内容を聞く余裕がまったくない場合は後刻の時間を指定して後で聞くこと。
②現状の仕事と割り込みの仕事についてチーム全体としての優先順位を考え、緊急度・重要度の高いものを優先させること。
③自分の担当業務に割り込ませることが不可能と判断した場合は、他の人に割り振ることや他の代替案を示すこと。

メソッド＃９ 【割り込み作業に対応するポイント】

①顧客との会話を絶やさず、早い時点での割り込み要求をキャッチすること。
②全ての工程において必ず割り込み作業が入ることを前提に、各工程には一定のバッファ期間をもたせておくこと。
③バッファ期間の確保のために、各工程の作業を前倒しで実施できるように事前の準備や作業の効率化および改善活動を継続的に行うこと。
④開発工程後半での仕様の追加および変更は、次回リリースにしていただくように交渉すること。
⑤後回しとなった作業については、再開時に問題がないように中断情報を記録しておくこと。

メソッド＃10【割り込み作業による時間の消費を削減する方法】

割り込み作業が常に優先的な仕事とは限りません。複数の仕事が重なった場合は、必ず依頼者や上司とも相談の上、優先順位を決めて取り組む必要があります。また避けられない緊急の割り込み作業は必ずあるという前提で、各仕事は可能な限り前倒しを行うようにしておく必要があります。前倒しするためにも改善活動は必要です。

2. 納期問題の解説

納期問題の起点は無理な短納期にあり、さらに種々の割り込み作業や前工程の遅延が時間不足を招き、不明確な期限問題が障害となっている姿が浮かんできます。

しかしながら開発における時間不足の原因を顧客側の無理な短納期要求だけに帰することは適切ではないでしょう。時間不足を招いている主な原因は次の通りです。

1） 開発時間不足の原因

①顧客側から強制された無理な短納期
②必要な開発期間の見積り間違い
③適切な時期までに決まらない要求仕様
④あいまいな要求仕様
⑤低い仕様理解力・仕様知識
⑥低いプロジェクトマネジメント能力
⑦低い技術力

2） 開発チームの生産性について

生産性とは、時間あたりに処理できる仕事量です。開発チームにおける生産性は次の数式で表すことができます。

開発生産性 ＝ 開発量 ÷ 所要時間

ただし、開発された成果物が顧客に受入れられる品質レベルであると言う条件においてのみこの数式は成立します。この生産性数式の右辺である所定の品質の開発量および所要時間を支配しているものは、開発チームにおける仕様理解力（仕様知識）および開発技術力（プロジェクトマネジメント能力＋技術力）に他なりません。これを数式であらわすと次のようになります。

開発生産性 ＝ 仕様理解力 x 開発技術力

すなわち開発生産性を向上させるためには、仕様知識を増やし仕様理解力を向上させ、必要なソフトウェア技術の習得およびプロジェクトマネジメント力を磨くことにより開発技術力を向上させることが必要です。

3） QCDの相互関係について

一定の能力をもった開発チームにおいて所定のソフトウェア開発を完遂するためには、その能力に応じた開発期間が必要なことは誰にでも理解できることです。あるコンピュータである処理を実行するためには10秒かかるものを、5秒で止めたら絶対に正しい答えが得られないのと同様です。

その開発組織の能力で2ヶ月必要な開発を、人員・人材の増強なしに1ヶ月で完了させることは不可能です。それでも無理な短納期を強制された場合に起こることは、実行すべきことが実行されない、すなわち手抜き仕事が行われるということです。すなわち低品質の仕事しかできないという結果になります。ごく単純化した数式で表すと次のようになります。

Q（品質） ∝ D（開発期間） 品質は開発期間に比例する。

またコストと開発期間の関係を、単純化した数式で表すと次のようになります。
C（コスト） ∝ D（開発期間） コストは開発期間に比例する。

上記の二つの数式でも明らかなように、品質もコストも時間の支配下にあり、ソフトウェア開発に限らず全ての人間活動の成否を決定しているものは、”時間である”という強い認識を持つ必要があります。すなわち、我々のソフトウェア開発において利益をあげ、品質を確保し、納期を達成するために実行しなければならないことは、開発の全工程において「時間を失う行為は全て排除しなければならない」ということです。時間を失う行為の排除に必要な活動は次の通りです。

4） 時間を失う行為の排除に必要な活動

①無理な短納期に妥協せず、合理的・科学的アプローチによるタフネゴシエーションを行うこと。
②要求仕様の全貌およびリスクの把握により必要開発期間・コストの見積り間違いをなくすこと。
③開発着手前に基本部の要求仕様の凍結を行うこと。
④仕様検討期間中に要求仕様の疑問点・不明点を全て解消すること。
⑤開発対象システムの仕様・機能を体系的に理解すること。
⑥プロジェクトマネジメント能力を磨くこと。
⑦技術力の向上に努めること。

第5章　時間不足解消のメソッド

1.　時間不足問題のメソッド

メソッド＃1 【時間不足の原因】

①そもそも実行に必要な時間が確保されていない。
②失敗および不要な作業による無駄な時間の浪費。
③割り込み作業による時間の消費。

多くの時間不足の原因は、上記原因の組み合わせによって発生していますので、それぞれの原因が引き起こしている問題を解決しなければ時間不足は解決しません。

メソッド＃2 【時間不足の解消法】

①早期の要求仕様凍結。
②見積りにおいて妥当な納期および費用の獲得をすること。
③顧客価値の重要度順に開発・評価を実行すること。
④仕事の効率化を図ること。今よりもっと効率的なやり方を考え実行すること。
⑤無駄の排除。不要なことや重複した業務をなくすこと。
⑥失敗の原因になりそうなリスクの回避・排除を事前に実行すること。

これらのことを実行しない限り、納期第一優先で品質もコストも犠牲にされ、さらに開発プロセスにおいても本来必須であるべき事前調査・設計書作成・レビュー・ラップアップなどの不履行や手抜きが続くことになるでしょう。

メソッド＃3 【必要な時間を確保する方法】

最も影響しているものは開発工程の始めに行われる見積りおよびスケジューリングです。見積りをできるだけ安くするために多くの見積りにおいては失敗やリスクの考慮を省いた見積りが行われやすく、また発注者側からの強制的な金額・期間のカット要求がある場合もあります。この様な状況に対処し必要な工数・期間を確保するためには、妥当性のある見積りを提示し、それを相手に納得していただく技量がプロジェクトマネージャやリーダーに必要です。その技量の基本となるものが開発における生産性のデータです。日ごろから開発におけるQCDのデータを収集していなければ、とうてい正確な妥当性のある見積りを提示することはできません。

メソッド＃4 【自分の時間を確保する方法】

時間の確保についての基本的な考え方は三つです。

1） やる必要のないことはやらないで済むようにすること

①リスクの排除

不要な失敗を防止するためにリスクを想定し事前にリスク排除を実行すること。失敗による後戻り作業は最大の無駄です。特に要件定義遅れによる後工程における仕様変更・追加の頻発は最大のリスクです。

②無理・無駄の排除

自分およびチーム内にある無駄や無理を徹底的に発見し排除すること。

2） やるべきでないことはやらないで済むようにすること

①責任範囲の明確化

自分あるいはチームの責任範囲でない活動は、本来やるべき人や部署にやってもらうような行

動をすること。但し自分らの責任外としては放置してはいけません。フォロー、督促などコミュニケーションを密接にし、支援が可能なことは支援すべきでしょう。責任範囲があいまいな場合は注意が必要です。

②役割の明確化

チーム内においては各担当の役割りの範囲を明確にしておくこと。また普段から部下の育成に力を注ぎ、自分の仕事の低レベルの部分はサブのメンバーに移管できるようにすること。

3） やるべきことは完全に実行すること

自分およびチームの責任である仕事は、設定した目標を目指して一致団結して完全にやり遂げること。これ位でいいだろうという気持には要注意です。

メソッド＃5 【時間を生み出す方法】

時間を生み出す方法の基本は、仕事自体の効率化と無駄およびリスクの事前排除にあり、具体的には下記のようなことが時間を生み出す行動となります。

①要求仕様を早期に凍結すること。
②見積りにおいて妥当な納期および費用を獲得すること。
③顧客価値の重要度順に開発・評価を実行すること。
④失敗の原因になりそうなリスクの回避・排除を事前に実行すること。
⑤改善活動を実施すること。仕事の効率化を図ること。今よりもっと効率的なやり方を考え実行すること。無駄の排除。不要なことや重複した業務をなくすこと。
⑥データやドキュメントに基づいた合理的な仕事の進め方や適切な段取りを行うこと。
⑦顧客間や社内における密接なコミュニケーションを実行すること。
⑧目的や意味が正しく伝わらない作業指示は無駄な作業や手戻りの原因となる。
⑨柔軟な設計は次なる顧客要求にも少ない時間で対応できる。
⑩中間レビューは結果として時間の節約に有効であり、レビューの精度を上げる。
⑪仕事の進捗を正確に把握するためには、量の管理だけではなく、実施単位まで落とし込まれた質の管理も行う必要がある。
⑫テスト漏れなどによる業務の後戻りは仕事のスピードを落とす無駄なものであり、モレを取り込んだテストパターン表やその表の流用は効果的である。
⑬求められている成果を確認せずにあいまいな仕事をすると手戻りの原因となる。
⑭優先順位を考えずに安易に割り込み仕事を請けると時間効率性を低下させてしまう。
⑮完了期日を確認しないまま仕事を進めてはいけない。

メソッド＃6 【仕事に余裕を生み出す方法】

①事前準備の実行

次に着手する仕事の事前準備として開発内容および段取りを明確にしておくこと。不明点や疑問点は直ちに明確にすること。

②振り返りの実行

仕事の終了時に必ず自分および他人の失敗の真因を明確にし、再発防止のアクションを行うこと。再発防止は改善活動としてチームでの取り組みを行うこと。

③改善活動の実行（弱点の克服と生産性の向上）

チームの弱点を明確にし、その改善活動を行う中で、苦手な分野を克服し、新たな知識を継続的に獲得すること。改善活動は、見積り力の向上、仕様の早期凍結および開発・評価力の向上などに注力すること。

メソッド＃7 【無駄な時間把握のポイント】

①仕様の不明点や疑問点を解消しないで想定開発による後戻りがないか。
②仕様を完全に理解して設計に着手しているか。
③過去における設計ミスや頻発するミスを繰り返してはいないか。
④重要な仕様の順に開発を進めているか。
⑤重複した作業をしてはいないか。
⑥仕事の段取りを間違えてはいないか。
⑦自分では解決できない問題にいつまでも捕まってはいないか。

メソッド＃8 【失敗および不要な作業による無駄な時間を排除する方法】

開発の各工程における失敗の原因を常に明らかにし、その改善対策を継続的に実施しなければ何度も同じ失敗を繰り返します。この問題に対処するためには失敗に学び、その対策をチーム内外に広げていくような活動、すなわち改善活動を続ける必要があります。また失敗に至らない場合でも、より効率的な業務の実施方法を考え、それを実行することも改善活動です。

2. 時間不足と品質悪化の負の連鎖

開発者たちの多くはスキル向上の障害となっている原因として”時間不足”をあげていますが、時間不足が解消されたら果たしてみなさんは新たな技術の学習に取り組むでしょうか。経験から言えることは、Noです。暇になったらなっただけ遊んでしまうのが人間のさがです。結局、自分が獲得したスキルは、最も過酷なプロジェクトの中でもがき苦しんだ中で習得したものだけだと言えます。

時間不足だと思うなら、その原因を探し、その問題を解決する行動を起こすことが合理的かつ妥当なことでしょう。現在の状況は技術スキルの向上どころか低下の悪循環を招いています。見積りにおいて妥当な開発期間の獲得に失敗し、要件定義の適切な時期における明確化に失敗し、その結果があせりを生み、プロセスや手順のスキップや中断を生み、多くの不具合を出し、その結果が多くの後戻りによる時間と工数のロスを生むという状態に陥っているのです。その悪循環の経過は以下の通りです。

（1）妥当な開発期間・費用の獲得失敗（見積り交渉の失敗、見積り誤り）
（2）早期仕様凍結の失敗
（1）および（2）を起因として、以下の悪循環の連鎖が繰り返される。
⇒（3）時間不足の発生
⇒（4）あせり
⇒（5）想定による開発（仕様未凍結のままの先行着手）
⇒（6）プロセス・手順のスキップ・中断
⇒（7）不具合の多発
⇒（8）後戻り・不具合修正（時間と工数のロス）
⇒更なる時間不足、「（3）時間不足の発生」への後戻り

第6章　目標設定のメソッド

1.　目標設定問題のメソッド

メソッド＃1 【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。

メソッド＃2 【目標の立て方】
①現状の問題点を洗い出し、その数値化を行うこと。
②問題点の真因を明確にし、改善策を立案すること。
③改善策の期待効果に見合った目標値を設定すること。

2.　目標設定問題の解説

当たり前のことですが、我々の目的は顧客の要求に従って顧客が満足できるソフトウェアを提供することにあります。しかしながら、この大目標を達成するためには、そのソフトウェア開発業務において更に具体的な目標を設定する必要があります。ソフトウェア開発業務における代表的な目標の指標として、Quality（品質）、Cost（コスト）、およびDelivery（納期）の三つがあります。これらの三つの指標を更に具体的にしたものが、例えば不具合発生率、利益率、生産性などです。自らの目標を設定し、その目標に向かって必要な活動を行わなければ、目標を達成することは不可能です。何らの具体的な目標も持たずに、行き当たりばったりの惰性的開発しか行えない組織においては、常に品質問題・赤字問題・納期遅延などの深刻な問題を抱える結果を招いています。

相談内容においてQCDの目標設定や数値化の必要性に言及したものは下記の通りでした（重複カウントを含む）。
1. QCDの目標設定の必要性に言及したもの　7件（35%）
2. QCD目標の数値化に言及したもの　1件（5%）
3. 具体的な数値目標の設定に至っていないと思われるもの　19件（95%）
（注. 件数および%データは目標設定に関する過去の相談データから抽出したもの。）

上記の分析結果は、ソフトウェア開発においてQCDの目標設定が必要であると感じている人は40%存在するが、それでも95%の人は具体的なQCDの目標値の設定なしに業務を遂行しているという非常に残念な結果を示しています。

第7章 コミュニケーションのメソッド

1. コミュニケーション問題のメソッド

メソッド＃1 【短時間日次会議実行のポイント】

◎メンバーからの報告

①昨日（or今日）何を実行したかを報告する。

②今日（or明日）何を実行するのかを報告する。

③今抱えている問題や困っていることを報告する。

◎リーダーからの応答

④各報告に対して、リーダーからアドバイスおよび全員で共有すべき情報を伝えること。
リーダーからの指示やコメントについては、チームメンバーによる誤解を防ぐために、丁寧に伝えること。

一人あたり5分程度の短時間情報共有を毎日行うこと。時間がかかる問題点の検討については、別途関係者のみで検討会を行うこと。また各メンバーが上記の報告をスムーズにできるようにするためには、就業時または始業時に、報告内容を各自の日報に記入することを習慣化および義務化する必要があります。これらの日次会議の実行によりメンバー全員の仕事の自律的な把握能力、報告能力および問題に対する気づき能力が養成され自分でものごとを考える力がついてくるでしょう。毎日ベースのコミュニケーションは問題点や進捗状況の共有のみならずメンバー間の性格的ないしは感情的な問題をも軽減するでしょう。

メソッド＃2 【感情本位から目的本位への転換のポイント】

①自分の関心を自分自身から仕事そのものに移すこと。顧客の要求するものは何かということを良く理解し、その実現に向けて情熱を注ぐこと。

②たとえ嫌いな相手であっても仕事に必要なことは必ず実行すること。

③人の好き嫌いで自分の行動を変えないこと。誰に対しても一貫した姿勢と行動を保つこと。

④自分ためにという過度な自己中心性を捨て、顧客要求の実現に向けてチームやメンバーのために貢献すること。

自分を害するものを減らしたければ、これらのことを実行する必要があるでしょう。

メソッド＃3 【顧客とのコミュニケーションのポイント】

①事前準備を行っておくこと

話題や議題が決まっているような会議においては、事前に聞かれそうな内容について事前検討を行い、答えを用意しておくこと。客先レビューならば、当然要求仕様の理解についてチェックされますので、仕様の理解内容およびその実現方法について説明をすれば良いだけのことです。客先レビューの場で部下がちゃんと応答できないのは、上長が部下に開発仕様の目的・意図・背景などの骨子を説明していないことや、社内での事前レビューを行っていないからでしょう。出たとこ勝負で、部下に丸投げでは自分が恥をかいても当然でしょう。原因は部下にあるのではなく、上司のあなたにあるのですから。

②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全員に発表やプレゼンの機会を与えること

公式の場、特に客先との会議の場に慣れていない人、特に若手の人にとっては上手に話をすることは非常に難しいことです。緊張する場面での質問には上手に答えられないでしょう。普段

の業務活動において、社内レビュー時、社内会議、社内報告会、社内発表会、日次会議などを利用し、チームメンバーの全員が議長や発表者になる機会を設ければ、客先とのレビュー会議などにおいて物怖じすることもなくなるでしょう。社内でのコミュニケーションが不活発であるのに客先とのコミュニケーションがうまくいく道理はありません。

メソッド＃4 【顧客からの問合せに対応する仕組み】

①開発担当と顧客問合せ担当の分離

ある程度規模が大きな、あるいは緊急の開発を行っている場合は以前にリリースしたソフトの障害問合せや対応を現在の開発業務と平行して行うと進行中の開発品質や進捗に少なからず影響を与えてしまいます。人数にも限りがありますのでチームを完全に二つに分けることはできませんが、その時どきの状況に応じて開発がメインのメンバー、障害対応がメインのメンバーなどの細かい役割分担を随時決めて対応する必要があります。

②顧客からの障害調査依頼や問合せに対する対応および回答の窓口の一本化

外部からの問合せに対して開発メンバーそれぞれが自分の担当領域について受付および対応を行っている場合が多いと思いますが、窓口担当を正と副の 2 名程度に絞った方が効率的にいくのではないかと思います。窓口担当には判断能力のあるリーダークラスの方が適当ですが、そのような役割をもつ人がいれば問い合わせ内容と開発の進捗状況の優先順位および調査業務の振り分けもスムーズにいくでしょう。

③外部からの問合せの一元管理

外部からの問合せについては重要な情報がほとんどですから、問合せ票などで一元管理を行う必要があります。不具合管理票と同様の管理が必要となります。

④問合せや障害そのものを減らす改善対策の実行

現在までの問い合わせ内容や障害通知について分類や分析を行い、それぞれがどのような原因で問合せに至っているのかを分類整理し、それぞれに該当する仕事や工程の業務品質を上げる必要があります。

2. コミュニケーション問題の解説

コミュニケーションの問題は大きく分類すると、「対人関係における感情的な姿勢の問題」および「コミュニケーションの方法に関する問題」の二つに分類することができるでしょう。

【コミュニケーション不良の原因】

①対人関係において感情的な姿勢が強いこと。
②コミュニケーションの方法を知らないこと。

対人関係における感情的な姿勢の問題の真因は、本来科学的・合理的なアプローチでしか解決できない仕事上の問題に対して、感情的なアプローチで対応しようとしていることに原因があるものと思われます。例えば、自分に自信がなく恥をかきたくないため「不明点についての質問をしない」と言う感情的な対応は、不明確な仕様のまま想定による開発を行った結果大きな手戻りや不具合を発生させてしまいます。 一方、「分からないから聞く」という科学的・合理的な対応を行えば、正確な仕様の基に正確な開発を進めることが可能となります。どちらが良いのかは明白です。

コミュニケーションの方法に関する問題については、具体的なコミュニケーション方法を知らないために感情的な対応に終始し、自ら問題を複雑化してしまいます。例えば、仕様を説明するにあたっては、整理されたドキュメントを元に説明しなければ、いくら口頭によって説明を尽くしたとしても誤解や漏れを防ぐことはできないでしょう。

この当然なドキュメント・ベースに拠る仕事のやり方を軽視している限り、伝わらないのは自分や相手の能力のせいであるという感情論から抜け出すことはできないでしょう。たった一枚でも良く整理された資料を使用して過不足のない情報の伝達を行うことと、資料の作成時間を惜しんだ結果、漏れや誤解によるもっと大きな時間と労力の浪費は、どちらが得なのかは明白です。開発情報は以心伝心で伝えられるほど簡単なものではないでしょう。

3. 信頼関係問題の解説

信頼関係は下記の三つのグループに分類することができます。
①自分自身に対する信頼感 ②上司・部下の信頼関係 ③顧客との信頼関係
それぞれの問題に関する分析を以下に示します。

1） 自分自身に対する信頼感を低下させる原因

自分自身に対する信頼感を低下させる原因として相談者が考えているものは、下記の三つに分類することができます。

1. 自分の性格に関する問題
2. 自分の価値観に関する問題
3. 自分の技術的な能力に関する問題

詳細内容およびそれぞれに対するアドバイスは以下の通りです。

1. 自分の性格に関する問題

〇失敗の隠蔽 ⇒ 恥を恐れず、過ちの速やかな修正行動を。
〇我欲 ⇒ 具体的に自分事として現実の直視を。
〇臆病な性格 ⇒ その性格を自己防衛に使うのではなく仕事に活かすこと。
〇他人を信用できない ⇒ 感情的な反応をやめ仕事本位・目的本位で他人との連携を。
〇仕事の好き嫌い ⇒ 好悪の感情を乗り越えて合理的な問題の解決を。
〇感情本位な仕事のやり方 ⇒ 合理性＋妥当性のある思考や行動を。

2. 自分の価値観に関する問題

〇私事優先 ⇒ 公私の重要度・緊急度のバランスを。
〇繁忙時の消極的対応 ⇒ 柔軟な対応を。

3. 自分の技術的な能力に関する問題

〇視野の狭さ、経験不足 ⇒ ドキュメント・ベースの仕事を。更なる学習を。
〇他人との力量の差 ⇒ 失敗の経験に学ぶこと。

（1） 感情的・情緒的なアプローチでは信頼感の獲得はできない

自分自身に対する信頼感を低下させている原因として、多数の人は自分の性格に起因すると考えており、自分の技術能力および価値観についての問題提起は少数です。これらのことを相談者の総体としての言葉に表すと「自分が自分を信じられない原因の多くは、自分の性格に原因がある」と言っているようです。これらの思考の傾向は、開発業務のみならず生活行動全般における信頼関係の構築問題に対して、感情的ないしは情緒的なアプローチである「自分の性格を変えること」で対応しようとする傾向を生み出しており、その結果として、いつまでたっても信頼関係の構築ができない状況に陥っているものと思われます。

（2）科学的合理的なアプローチが信頼感の獲得をもたらす

一方、自分自身に対する信頼感すなわち自信を獲得するためには、自分の性格を変えようとする

ことではなく、科学的合理的なアプローチが必要です。アドバイスの内容を集約すると次のようになります。自分自身に対する信頼感、すなわち自信を獲得するためには、合理性および妥当性のある思考や行動が必要であり、目前の問題を自分事として直視し、好悪の感情を乗り越えて、仕事本位・目的本位で仕事に取り組み、失敗の経験に学び、過ちに対しては恥を恐れず速やかな修正行動をとり、仕事の遂行にあたってはドキュメント・ベースを基本とし、問題解決にあたっては緊急度・重要度のバランスを考慮した柔軟な対応を行い、自己の性格を過剰な自己防衛に消耗させることなく仕事や生活に活かしきることが必要です。

2） 上司・部下の信頼関係を低下させる原因

上司・部下の間の信頼関係を低下させる原因としては、下記の三つに分類することができます。

1. 相手に対する配慮不足の問題
2. 自分における利害得失の問題
3. 消極性に関する問題

詳細内容およびそれぞれに対するアドバイスは以下の通りです。

1. 相手に対する配慮不足の問題

・部下の意見を聞けない ⇒ 相手の疑問点・不明点を聞きだすこと。
・一方的な自分の意見の押し付け ⇒ ていねいなコミュニケーションを。
・部下に誠実に接してほしい ⇒ 動いてくれるような頼み方をする工夫が必要。
・メンバーの公平な扱いができない ⇒ それぞれの能力に応じた指導を行うこと。
・ミスが多い後輩社員 ⇒ 相手が理解できるレベルの説明を。
・パソコンをしながらの片手間のレビュー ⇒ 人を軽視した愚かな行為。

2. 自分における利害得失の問題

・メンバーに対するヘルプは自分の役に立つのか？ ⇒ 情けは他人のためならず。
・面倒見の悪い先輩社員 ⇒ 自分だけの利によらず仲間との連携を。

3. 消極性に関する問題

・場所が分散している部下のまとめ方が難しい ⇒ こまめな声かけと短時間情報共有会議を。
・上司の過剰な期待が重荷になっている ⇒ 過剰な期待や要求には条件提示を。

（1）共通の問題は自己中心性にある

上司・部下の間の信頼関係構築を阻害する原因の大部分は、相手に対する配慮不足の問題で、自分の利害、自分の消極性がこれに続いています。これらの三つの問題に共通する心理的な特徴は一方的な自己中心性でしょう。自己中心性は、感情的・情緒的な思考・行動の最たるものの一つです。

ここで取り上げられた問題は全て、「自分の意見を通したい」「自分の都合の良いようにしたい」「自分の利益にならないことはしたくない」、と言う言葉で表すことができるでしょう。チームや組織を組んで仕事を遂行することの意味を理解していない人、目先の自分の利益に執着する人たちにおいては、いつまでたっても最大の価値観は”自分のためにだけ”なのでしょう。

人間においては、その生存や成長を確保するためには一定の自己中心性が必要ですが、チームや組織内の各メンバーが際限なく自己中心的な行動をとってしまえば、その結果は間違いなくチームや組織の崩壊につながってしまいます。自己中心性を一定のレベルに抑え、「他人の意見を聞きつつ」「他人の都合を配慮しながら」「目先の利を他人に譲り」、最終的には自分のみならず他人も共に大きな成果を手にする道を選択することが合理的かつ妥当性に適った方法だと言えます。

3） 顧客との信頼関係を低下させる原因

顧客との信頼関係を低下させる原因としては、下記の四つに分類することができます。

①相互義務の不履行
②自分のおかれた状況の本質が見抜けないこと
③楽をしたいという自己中心性
④スキル不足

詳細内容およびそれぞれに対するアドバイスは以下の通りです。

①相互義務の不履行

・守られにくい約束　⇒相互義務の履行を。
・横柄な態度の顧客　⇒いったん顧客側の立場になって本質を見定めること。
・誠実かつ責任ある行動は難しい　⇒分からないことは”感じとる”より“素直に聞いてみる”こと。

②自分のおかれた状況の本質が見抜けないこと

・進捗遅延の中での顧客の厳しい要求　⇒問題の本質は開発の遅延とその対策。
・顧客内部情報の報告先に迷う　⇒内密情報は信頼関係のある者だけに伝えること。

③楽をしたいという自己中心性

・不都合な時間帯の電話依頼対応　⇒電話には直ちに出て緊急・重要度を判断し対応すること。
・困難な仕事を避ける思考・行動　⇒楽をして良いものを手に入れる方法はない。

④自分の技術的な能力に関する問題

・自分のミスや知識不足　⇒具体的な改善目標を設定し、地道な解決の取り組みを。
・業務実績を積み上げられない　⇒気分本位ではなく顧客価値の重要度順の仕事を。

（1）対外的信頼関係の基本は、自己に対する信頼感・社内での信頼関係の構築にある

顧客との信頼関係構築に特有の問題は、①相互義務の不履行および②自分のおかれた状況の本質が見抜けないこと、の二点に現れています。

相互義務の不履行は、お互いの自己中心性に原因があり、状況の本質が見抜けない原因は、感情的・情緒的な思考・行動に原因があります。

また、③楽をしたいという自己中心性については、（2）上司・部下の信頼関係を低下させる原因において言及した通りで、④自分の技術的な能力に関する問題については、（1）自分自身に対する信頼感」において説明した通りです。いずれにしても、自分自身に対する信頼感および上司・部下の信頼関係において克服されていない問題は、信頼関係の中で最も大きなプレッシャーを受ける顧客との信頼関係の中で、端的な問題として現れて来ると言うことです。

要するに、対外的な信頼関係を構築するためには、自分自身における信頼感の構築および社内での信頼関係の構築を先に実現する必要があると言うことです。

4） 信頼関係問題分析のまとめ

◎ 感情的・情緒的なアプローチでは信頼感・信頼関係の獲得はできない
◎ 科学的合理的なアプローチが信頼感・信頼関係の獲得をもたらす
◎ 共通の問題は自己中心性にある
◎ 対外的信頼関係の基本は、自己の信頼感・社内での信頼関係の構築にある

第8章　プロジェクトマネジメントのメソッド

1.　プロジェクトマネジメント問題のメソッド

メソッド＃1 【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
⑤開発仕様の早期提示

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。統合管理が行われないプロジェクトは必ず失敗するでしょう。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く必要があります。短納期でしかもドキュメントが不足している場合の最後の手段はベンダー側を巻き込んで全ての開発メンバーが同一場所に集結して開発を実行することです。

メソッド＃2 【プロジェクトマネジメント業務】

①適正な見積りを行うこと。
②早期仕様凍結を図ること。
③プロジェクト計画の立案と実行を行うこと。
④プロセス管理を行うこと。
⑤プロジェクト進捗管理を行うこと。
⑥リスク管理を行うこと。
⑦課題管理を行うこと。
⑧損益管理を行うこと。
⑨プロジェクト完了報告を行うこと。
⑩日次情報共有会議を行うこと。
⑪プロジェクト関係他社チームとの情報共有・連携を行うこと。

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でしょう。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。

我々を取り巻く環境は十数年前なら元請けベンダーのプロジェクトマネジメントに従って動いていれば済みましたが、現在はすでに元請けベンダーがやるべきプロジェクトマネジメントが放棄されているような状況が多々見受けられます。このような状況下で下請け各社が自分の身を守るためには、プロジェクトリスクの徹底的な解決を筆頭とした前記のプロジェクトマネジメント業務を自ら実行せざるを得ない時代になってしまっています。

プロジェクトリーダーの人材不足を解消するためには、まずあなた自身がプロジェクトマネジメント能力を身につけることを目指していただきたいと思います。そしてその影響力を他のメンバーにも及ぼし、次に続く人材の育成を仕事を遂行していく中で実現していただきたいと思います。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大規模な改善活動から多くの有能なプロジェクトリーダーが生み出されることでしょう。

メソッド＃3 【プロジェクトリーダーの役割】

①適正な見積り回答の実行

・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こと。
・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／納期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。
・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。
・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。
＊「リスク管理表」が存在しないことが最大のリスクとなります。
＊①～⑪すべての実行品質がリスクの原因となり得ます。失敗すれば損失となり、成功すれば利益となるでしょう。

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。

メソッド＃4 【複数プロジェクトの管理方法】

①管理方法のパターン化

日々出てくる問題に対処療法的に対応していたのでは、いくら時間があっても足りません。そのためには個々のプロジェクトの管理手法が確立されている必要があります。そのためには、何をいつチェックすべきか、ということが可視化されている必要があります。これらに必要な管理資料としては、プロセス管理表、要件管理表、品質管理表、進捗管理表、リスク管理表、課題管理表などがあり、常にこれらの資料にプロジェクトの現在状況を反映し、役に立つように整備しておけば複数プロジェクトを一人で管理することもできるでしょう。プロジェクトの規模が小さければこれらの管理資料の運用に必要な時間も相対的に少なく済み、さらにこれらの管理表への登録を開発メンバーのルーチンワークに組み込むことでリーダーの負担を減らすことも可能でしょう。このような管理ドキュメントによるプロセス管理に慣れない間は多少の負荷増加になりますが、単なる口頭ベースの対処療法的な管理による混乱や無駄な時間の消費に比べれば、ずっと効率的なやり方と言えます。

②日次情報共有会議の実行

個別の問題に振り回されることを避けるため、およびメンバーの教育や情報共有のために、プロジェクトメンバー全員参加の、短時間の情報共有会議を毎日実行することが効果的です。

メソッド＃5 【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、設計は国内下請け会社、製造は海外オフショア会社、評価は国内下請け会社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。この分業体制には多くのリスクや問題点があります。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るのを見てきました。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。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会社に発注されている場合が多いでしょう。

この工程別分業方式の致命的な欠陥は次の通りです。

1）各工程が異なった会社に分離・分散されたことによる共有すべき情報の分断化

一社で全ての開発工程を担う場合に比べて、複数社で分割した工程を担うことは極めて難しいことは誰にも容易に分かる道理のはずですが、多くの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、単にリスク分散だとかコスト競争だとか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ないこと。

②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ないこと。

③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ないこと。

ベンダー側において上記三点を実行していなければ、仕事は責任を放棄した丸投げ発注になり、プロジェクトは必ず失敗するでしょう。ベンダーの下に有機的に統合された分業体制を「情報共有分業」と呼ぶならば、そうでない分業体制は悪しき「情報分離分業」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本来一つの開発組織においても実行されてい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です。下請け会社においては、ベンダー側において欠落している組織機能の補完を行う行動をとることが新たなビジネスチャンスともなるはずです。

2） 文化・言語が異なる海外オフショア会社との間のコミュニケーションの阻害

十分な統合管理もできず、ドキュメント・ベースの開発もできず、十分な開発体制の構築もできないような組織において、文化も言語も異なる海外の会社に設計・製造・評価などの工程を発注したらどうなるかは誰にでも分かることでしょう。すでに分離・分断されている工程間にさらに深い溝を掘ることになります。要求仕様書の行間を読めというような非科学的な姿勢しか持ち合わせていない人たちが作成した貧弱な仕様書に基づいて設計や製造を発注されたオフショア会社においては、きっと書かれているものだけしか設計・製造できないでしょう。おまけにあいまいな日本語は非ロジカル言語であり、その日本語で書かれた貧弱な要求仕様書は中国やベトナムの技術者にとって意味不明なしろものなのかも知れません。まともなプログラムができなくて当たりまえのことでしょう。それを補うために新たにまたブリッジSEなる職種を作りだしたり、未完のプログラムの受け入れ検査組織を作ったり、バグだらけの未完のプログラムを国内で補修するために新たな技術者を用意したり、結局オフショアのコストメリットが無くなるばかりか、赤字に陥り、劣悪な品質や納期遅延の結果を生んでいるのでしょう。

オフショアのみならず開発を成功させることのできる開発組織は、前記の統合管理、ドキュメント・ベース開発および優れた開発能力をもった組織だけでしょう。このことはベンダー側および下請け側の両方に言えることです。

メソッド＃6 【複数社担当開発におけるリスク】

①仕様の理解が相互に異なったためにおこるデグレや実装漏れなど。
②同一モジュール製造に複数社がかかわった場合、1社の開発遅延がロードブロックとなり、他社が製造開始できないこと。
③1社の不具合によってテストが進捗できないこと。
④複数社におけるベースソース部の改修の同期がとれず正式版のビルドができないために結合・総合テストが開始できないこと。
⑤オフショア会社の場合の注意点は次の通りです。
・意思の疎通が取りにくい、時間がかかること。
・修正対応が遅いこと。
・仕様に記述されていないことは質問もしてこないし、製造することもないこと。
・日本語仕様の解釈能力が低いこと。ただし日本語仕様書自体の品質も悪いこと。

これらのリスクを解消する対策を実行する必要があります。週一回の定例会議だけではなく、自社のチーム内では日次の短時間情報共有会議を毎日実行し、日々の進捗の確認、全ての問題・課題の解消、情報共有を毎日実行する必要があります。またベンダーおよび他社とのコミュニケーションも密接に行う必要があります。次に肝心なことは、ベンダー側においては必ず複数社に対する統合管理を実施しなければならないということです。

メソッド＃7 【派遣問題・役務契約問題】

派遣や役務に関して以下のような問題があります。

1. 派遣法の改正で、派遣常駐者に対して発注ベンダーからの直接的な作業指示が禁止されたこと。しかしながら現状においては偽装派遣が横行している。
2. 役務発注においては、JOB発注と違って、出来高制すなわちかかっただけの工数が請求されるため発注ベンダー側の予算が膨らみやすい。
3. 役務発注においては、発注ベンダー側の開発担当者が外注担当者になにもかも依存するような状況、すなわち丸投げに陥りやすくなり、結果として発注ベンダー側のプロジェクトマネジメント力や開発技術力の著しい低下を招いている。

外注開発者の立場からの見方だけをしていると、派遣や構内請負は環境が整っており働きやすそうに思えますが、業界全体を見てみると、上記のような見過ごすことのできないような大きな弊害が発注側および受注側の両方に発生しているのです。ものごとの大小にかかわらず、その本質を見抜くためには、自分の立場からだけではなく、相手の立場からも、さらに全体からの視野で見る必要があります。上記のような情勢の変化に対応するために、顧客との密接なコミュニケーションをとる方法や最適の開発環境を構築するためには何をしなければならないかについて、その時々に最適と思われる方法を考えて実行する必要があります。

メソッド＃8 【プロジェクトを成功させる三つのポイント】

①見積り交渉時における開発内容に見合った工期・予算の確保を行うこと。
②主要仕様の適切な時期までの凍結を行うこと。
③開発効率化のための継続的な改善活動を行うこと。

この3点を可能にすれば正しい開発、良い品質、納期の確保、利益の創出は自然に達成できます。これができない人、やらない人には永久に正しい開発はできないでしょう。

メソッド＃9 【作業手順とばしの防止策】

①同様の手抜きで過去に発生した問題の事例を直接本人に示してやること。
理屈を言っても分からない人たちに対しては、具体的に起った問題を目の前に突きつけてやることが最も効果的です。可能ならば、その本人自身が過去に起こした問題を突きつけてやることが最も効果的でしょう。

②作業手順書を作成し、全員に自己チェックを義務づけ報告させること。

③現在の手抜き行為に大きな影響を与えている要因の軽減・削減を行うこと。
「忙しすぎる」「疲れている」は改善活動の実行による仕事自体の効率化によって解決すべき問題であり、さらに言えば自主的な改善活動は開発者の論理性・合理性・自主性・自律性を育てる原動力となります。改善活動はリーダーおよびマネージャが主導しなければならない重要な責務です。

メソッド＃10 【解消すべき三つの時点のリスク】

①開発の入り口で押さえること

・適正な見積りにて妥当な開発期間と費用を獲得すること。
・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”として残すこと。
・QCDの数値目標の設定を行うこと。

②開発中に押さえること

・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。

③開発の出口で振返ること（ラップアップミーティング）

・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。
・失敗に学ぶこと。失敗あるいは創意工夫の記録を振返り、次の開発および開発者に申し送ること。

メソッド＃11 【スムーズな開発を実行するためのポイント】

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リーダーが開発仕様の全体像を押さえておくこと。
④開発メンバーに対して適切な仕事の配分を行うこと。
⑤開発メンバーに仕事内容を十分に理解させておくこと。
⑥リスクの解消および改善活動を実行すること。

メソッド＃12 【チーム組織編成のポイント】

①組織編制はリーダー、サブリーダー、メンバーなどのスキル階層別のピラミッド構成を築くこと。
②スキル階層を構成するためには、リーダーは複数のサブリーダーを育成し、サブリーダーはメンバーを育成すること。
③人員異動の要求に対してはチーム能力の急激な低下を防ぐために二番手のサブリーダーおよび二番手のメンバーを異動させること。

④異動させたメンバーに関しても自分の組織に必要な人材は、将来元の所属に戻す約束を取り付けておくこと。
⑤特定顧客に対するノウハウの属人的な保有のリスクを減らすために、必要なノウハウは必ず最新状態として技術ドキュメントに記載し更新を怠らないこと。

メソッド＃13 【失敗に学ぶポイント】

①ドキュメント化

後戻りや仕様追加対応において新たに獲得した仕様知識や実装理解について、他人でも理解できる形として全員でドキュメントに残しておくこと。

②仕様理解

スケジューリングミスの大元は開発仕様の理解不足による工数見積りミスだと思いますが、次開発においては開発初期の短期間において主要メンバーにての仕様理解と妥当な見積りに全力を挙げる必要があります。

③顧客とのコミュニケーション

顧客との密接なコミュニケーションを利用して、顧客が保有している仕様ノウハウ、実装ノウハウを継続して入手するようにすること。これらのノウハウに関しても個人レベルで保有せずに、必ずドキュメントに残してチームの財産として蓄積・利用すること。

④メンバーの育成

今回の開発で成長が見られたメンバーに関しては継続的にこのチームに残して更なる育成を図ること。

⑤損失の回復

次回の開発では今回の超過分を是非取り戻すこと。

メソッド＃14 【アジャイル思想の骨子】

①チームにおける日常的で密な直接コミュニケーション力
②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
③顧客との日々の直接コミュニケーションによる協調
④変更要求に対する俊敏な対応力
⑤顧客価値優先度の把握力
⑥意欲ある人材で構成されたプロジェクト構築力
⑦短期開発力
⑧一定のペースを持続できる開発力
⑨技術的な優位性の保持、良好な設計力の保持
⑩無駄・不要な仕事の削減力
⑪自律的チーム力
⑫定期的・効果的振り返りによる行動の変更・調節力

アジャイル開発の形式的な特徴であるイテレーション開発・スプリント開発（短期繰り返し開発）をまねることがアジャイル開発ではありません。まず我々が取り組むべきことは現在の開発手法の中にあっても、アジャイル的な上記の価値・原則の実力を身につけるところから出発するしかありません。

メソッド＃15 【空き要員を出さないためのポイント】

①客先の仕事の増減情報をこまめに入手すること。
②各要員における仕事の期間について正確な情報を常に入手すること。
③現在担当しているプロジェクト以外で、品質問題を抱えているプロジェクト情報を入手すること。
④チームリーダー間およびマネージャ会議等で客先の情報を小まめに交換すること。
⑤要員の仕事割り当ておよびスキル管理を小まめに実施し、3ヶ月先を見通した活動を行うこと。
⑥他のプロジェクトの応援先をさがすこと。
⑦現行業務以外の仕事で遂行可能な仕事も検討しておくこと。
⑧空きの期間が長引きそうな場合は、協力会社の要員を整理すること。
⑨万一、空き要員が出てしまった場合は、空きの期間に現行業務の改善活動を実行すること。

メソッド＃16 【協力会社からの人材採用のポイント】

1. 採否の条件
①必要な技術的能力を保有しているか。
②コミュニケーションに必要な自律的な報告・連絡・相談などの行動ができるか。
③チームプレーに必要な協調性・誠実性などがあるか。
2. 採用可否の 3 つチェックポイント
①最初の面談時 ②一ヶ月後のチェック ③3ヵ月後のチェック
前記の採否の条件を三つの時点にてチェックし採否を決めること。
3. 個人別人材カード
上記ルールに則った場合でも当たり外れは発生しますので、協力会社から採用した人たちに関して個人別の人材カードを作成しておくことをお勧めします。個人別人材カードには上記の 3 つのチェックポイントにおける採否の条件についての評価結果を記入しておき、一定期間ごとに同様の評価を継続しておきます。当初の評価は低かったが実際は良好な結果を出している人や、逆の場合について、その見込み違いの理由などを記入することで、新たな人材を採用する場合の見込み違いを防止する新たな採否条件を加えることができるでしょう。またこの個人別の人材カードは、プロジェクトチーム編成時における協力会社メンバーの選択資料としても有効に使えます。

2. プロジェクトマネジメント問題の解説

プロジェクトマネジメント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。提起された問題のほとんどは、このWhatとHowの複合問題と言えます。

プロジェクトリーダーが何をすべきか、ということに関しては、【プロジェクトリーダーの役割】を十分に認識した上で、各問題におけるアドバイスを参考にして、それぞれの問題にどのように対処すべきか、ということを実状に合わせて自分自身で考え実行されることを期待します。

さらに全問題に渡っての皆さんの懸案事項である「時間が足りない」ことに関しては、【自分の時間を確保する方法】および【時間を生み出す方法】の実行を期待します。

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。

3. チームプレー問題の解説

全体を通して皆さんの主張は、仕事を成功させるためには連携・連帯のチームプレーは必要だと分かってはいるが、忙しくてできない、対人関係に弱くてできない、能力がなくてできない、自分の利益にならないのでやりたくない、のオンパレードです。本当にチーム内の連携や連帯が自分の目先の利益や都合よりも高い優先事項であるとは誰も思ってはいないのでしょう。

信頼関係の構築を阻害している共通の問題は”自己中心性”にあり、同様にチーム内の連携や連帯を阻害しているものもこの自己中心性にあると言えます。つまり「自分の利益にならないことはしたくない」「自分の都合の良いようにしたい」ということなのです。このような感情態度や行動は人間の生存や成長を確保するためには一定のレベルは必要ですが、誰も何も譲らないような組織やチームは結局、自分も含めて崩壊してしまう結果を招きます。自分も仲間も皆ともに成長・発展していくためには、「他人の意見を聞きつつ」「他人の都合を配慮しながら」「目先の利を他に譲る」ことが合理的かつ妥当性に適った方法でしょう。プロジェクト活動は問題・課題の解決業務であり、必ず他人の支援が必要な局面に何度も遭遇するものです。自分一人では絶対に切り抜けられない局面が必ずあるものです。そんな時に、人を助けようともしない人間をほかの誰が助けようとするでしょうか。これは合理性や妥当性を越えた人情というものです。

第9章　設計・製造のメソッド

1.　設計・製造問題のメソッド

メソッド＃1 【設計の作法】

①仕様凍結および設計は重要な仕様の順に行うこと。
②設計をあせらずに、仕様を完全に理解してから設計を行うこと。
③仕様の不明点や疑問点を解消しないまま、推定や想定で設計を行わないこと。
④過去の設計ミスや発生頻度の高い設計ミスを網羅した自己チェックシートを組織的に運用すること。
上記の活動を組織的に実行すれば必ず設計ミスは大幅に減少します。

メソッド＃2 【詳細設計書記載内容の例】

詳細設計書に記載が必要な項目には以下のようなものがあります。
①機能概要説明　②画面フロー　③印字仕様　④伝送仕様・伝送シーケンス
⑤異常処理定義　⑥イベントマトリクス　⑦テーブル関連図　⑧メモリー処理定義
⑨関数関連図・関数一覧　◎レスポンス・パフォーマンス定義

メソッド＃3 【効果的レビューのポイント】

①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。
②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する。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こと。
③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。
④共通レビュー項目の雛形を作成しておくこと。また既存のレビュー資料で流用できるものは流用すること。
⑤影響度の確認レビューを行う。
⑥セルフチェックの時間を設け、レビューしきれない部分について事前にチェックを済ませておく。
⑦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こと。
⑧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すこと、および効果的なセルフチェック資料を作成すること。
⑨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こと。

メソッド＃4 【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されること。

①事前に自己チェックを行うこと

・詳細レビューは自己チェックにて実施しておき、単純なミスはこの段階で全て排除しておくこと。
・過去の高頻度の設計ミスなどを網羅した自己チェック用の基本的チェックシートを用意すること。

②中間レビューと本レビューの二段階レビューを行うこと

中間レビューにおいては重要仕様の理解が正しいか、およびその内容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こと。本レビューにおいては重要仕様についての最終チェックを

行うと同時に、時間の許す限りその他の仕様のチェックを行うこと。

③レビュー内容のポイント

・不明な仕様を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こと。
・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。

メソッド＃5 【設計レビューの受け方のポイント】

①事前に説明する設計骨子を整理しておくこと
②設計骨子の重要な順に、理解した要求仕様内容及び実現方法である設計内容を説明すること。
③疑問点・不明点は都度レビュアーに判断を仰ぐこと。

メソッド＃6 【効率的なコードレビューのポイント】

①関係者だけが参加すること
②重要な結合部の順に行うこと
③詳細設計書に基づくこと
④過去の失敗例を参照すること
⑤事前の自己レビューの実施
⑥中間レビューの実施

メソッド＃7 【レビューの成否はドキュメントの出来栄えに依存する】

レビューはドキュメントに基づいて実行する必要があります。基本設計のレビューにおいては要求仕様書との対応づけに基づき、詳細設計書のレビューにおいては基本設計書との対応づけに基づき、コードレビューおよび試験設計書レビューにおいては詳細設計書との対応づけに従ったレビューを行う必要があります。各々のドキュメント間の項目の対応づけが正確になされていればレビューがよりスムーズになり、抜けや矛盾点も発見しやすくなるでしょう。

また誤記、記入抜け、同じことを複数個所にバラバラに記述、分かりにくい冗長な日本語記述、などの設計書では効率的かつ正確なレビューを行うことはできません。過不足のない誰が読んでも同じ解釈が容易に行える設計書を用意する必要があります。レビューがうまく行かない最大の原因はまともな設計書がないことにあります。その意味で開発ドキュメントの最初の出発点である要求仕様書および基本設計書の完成度の高さが全ての工程における効率性および品質の高さの原点となります。

2. 設計・製造問題の解説

（注. 件数データおよび%は本問題に関する過去の相談データから抽出したもの。）

1） 開発全般に関する問題について

提起されたほとんどの問題の本質は、次の二点に集約することができます。

1. 情報共有がリアルタイムかつドキュメント・ベースで行われていないこと。
2. 開発に関係する複数会社間の開発行動の同期が取れていないこと。

この二つの問題を解決するための対策は次の通りです。

①ドキュメント・ベースによるリアルタイムな情報の共有

仕様のみならず開発に関する情報は、口頭だけに依存せず、必ずドキュメントにより、決定および変更が発生したその瞬間において、すべての関係者に共有されるようにすること。そのためには必要なルールおよび仕組みを事前に構築しておき、チーム内の日次情報共有会議の実行および関係会社間の密接なコミュニケーションが必須条件となります。

②統合的なプロジェクト管理の実行

リアルタイムな情報共有の実行を前提として、開発の主管会社であるベンダーは、複数の開発会社における開発行動の同期をとるための統合的な進捗管理および問題管理を行う必要があります。 ベンダー側にて人材不足の場合は、下請け会社はこれらの業務を補完する提案を行う必要があります。これは下請け会社における新たなビジネスチャンスとなります。

上記問題に共通していることは、情報および行動の両方におけるリアルタイムな同期が取れていないという時間認識および時間制御に関する問題です。「時間問題の解決なくしては開発問題の解決なし」と言えるでしょう。

2） 設計問題について

各々の問題および発生原因は下記の通りでした。

①パフォーマンスの考慮モレ
⇒システム機能の時間的基本要件であるパフォーマンス・レスポンスに対する認識の欠如。

②二重障害の発生
⇒時間不足のあせりによる設計プロセスのスキップ。

③インターフェース仕様決定の後回しによる後戻り作業の発生
⇒設計・製造時間不足のあせりによるインターフェース仕様決定の後回し。

④仕様調査をしながらの設計は時間がかかる、詳細設計の時間が足りない
⇒見積り時に実行すべき仕様調査の後回しによる見積り精度の低下と設計効率の低下。

⑤阿吽の呼吸でやったはずが抜けによる失敗が出る
⇒情緒的なコミュニケーションによる失敗。

⑥メモリーオーバーの不具合発生
⇒納期切迫にて仕様の調査・検討をスキップ。

⑦基本設計が終わっていないのに詳細設計着手
⇒基本設計工程のスキップ。

⑧設計の精度が低く障害発生の元になっている
⇒仕様・機能知識不足、設計書等のドキュメントの低品質。

上記の設計に関する問題の8割は、設計技術そのものに関するものではなく時間不足に起因する問題でした。たとえアドバイスとしてパフォーマンスを考慮した設計をしてくださいとか、設計を最初のプロセスとして実行してくださいと言ったところで、結局”時間が足りない”ということで、これらのアドバイスは一向に実行されない可能性が高いと思われます。これらの問題を一挙に解決する方法は、やはりこの”時間不足問題”を解消するしかないでしょう。開発に必要な時間の獲得に失敗している原因、自分たち自身によって時間を失っている原因の二つを解消する必要があります。

3） 設計書問題について

各々の問題の内容は下記の通りでした。

①ドキュメントのフォーマットに関する問題　2件
②設計書の誤記が減らない、メンテされない　4件
③仕様の目的・意味・背景の記述がない　1件
④複数の仕様書間の記述の不整合　1件

この問題の50%を占めている”設計書がメンテされない問題”の理由も、”時間がないから”と言うことでした。

4） 設計レビュー問題について

各々の問題および原因は下記の通りでした。

①レビュー対象が膨大でレビューし切れない　1件　⇒レビューの目的の理解不足。
②レビューでも見つけにくいミスがある　2件　⇒設計の効果的・効率的な段取り不足。
③時間不足でレビューを省いてしまう　2件　⇒時間不足によるあせり。
④レビューを受ける立場として効果的な説明ができない　1件　⇒説明力の不足。

設計レビューの目的は、全ての設計内容についての検証を行うことではなく、仕様の骨子および重要な仕様についての理解が正しく設計に反映されているか否かについての検証にあることが意外と理解されていません。そのために細目に渡って全ての内容に対して全件チェックを行わなければならないという非合理的な考え方にとらわれるために、なお一層、時間不足によるあせり感が高じてくるものと思われます。その結果として、レビューのポイントがずれ、効果的な自己検証や説明ができず、自己チェックやレビュー自体をスキップするという最悪の選択に至ってしまうものと思われます。開発におけるあらゆる活動を効果的・効率的に遂行するためには、その活動の目的について明確に理解しておくことが重要です。

5） 影響度問題について

①設計前の仕様検討不足によって後戻り作業が発生　1件　⇒時間不足によるあせり。
②影響度表が提出されない　1件　⇒ソフトウェア構造体の知識不足。

①の問題は、設計前の段階である見積り時や仕様検討時において、時間不足のあせりによってもたらされる既存機能への悪影響を検証しておく作業の不足ないしはスキップに原因があるものと思われます。 また②の問題は、ソフトウェア構造体を表すドキュメントや知識が不足している場合には、影響度表の作成自体ができないということに原因があります。

6） 製造問題について

時間不足によるあせりや判断力（自律性）の欠如による手抜き行為に注意が必要です。 ソースコードのコピペミス、すぐにコードに手をつけ失敗する、コードレビューのスキップ、詳細設計書を先に作成せずにコードを作成する、詳細設計書なしのコード作成などの問題があります。 ここで提起された問題のすべてが、時間不足というあせりによる手抜き開発と言えます。

7） 分析のまとめ（時間制御の失敗）

設計・製造に関する各問題グループに共通する基本的な問題は”時間制御の失敗”にあると言えます。各問題の分析をまとめると以下のようになります。

①時間制御の失敗

時間制御の失敗とは次のことを指します。

①開発に必要な妥当な時間の獲得失敗
②実行すべきタイミングの失敗
③複数組織間の時間的な同調・協調行動（シンクロナイゼーション）の失敗
④ものごとの実行順番の間違い
⑤”時間”というものに対する認識不足

②現在の工程別分業方式が招いている弊害

・統合的なプロジェクト管理、すなわち統合的な進捗管理および問題管理が行われていない。
・情報のリアルタイムな共有が行われていない。
・各社の開発活動のリアルタイムな同期が行われていない。

③開発時間不足のあせりが招く問題行動

・仕様調査不足により、仕様の理解や要求の全体像の把握が不十分である。
・仕様変更による影響度表の作成ができない。
・設計プロセスなどが実行されない場合もあり、開発業務自体の質の低下を招いている。
・コミュニケーションの質の悪化により、正しい情報がリアルタイムに共有されていない。
・開発ドキュメントのメンテナンスが行われず、ドキュメントの質の低下を招いている。
・各工程における有効なレビューが行われず、不具合を後工程に流している。
・仕様理解不足のまま、まともな設計書なしでのプログラム製造が行われている。
・ドキュメント・ベースの開発を実行することができない。

④”時間”というものの認識について

個人であれ組織であれ、それぞれが持っている時間は、言うまでもなく有限なものです。我々が仕事をするにあたって、最初に意識しなければならないことは、有限な時間においても、必要時間と許容時間（手持ち時間）の2種類があるという当たり前のことです。

必要時間とは、あるものごとを遂行するために必要な時間のことで、その長さは個人ないしは組織の業務遂行能力（パフォーマンス、生産性）に依存します。 一方、許容時間（手持ち時間）とは、業務遂行にあたって自分たちに許されている時間のことです。必要時間と許容時間が一致していれば時間不足という問題は発生しませんが、競争社会の中にあっては常に他より早くという圧力を受け続けるために、許容時間が必要時間よりも短くされる結果となりやすい傾向が強くなります。

不足時間＝許容時間－必要時間で表されます。許容時間が80時間で必要時間が100時間ならば時間は－20時間の不足となります。 不足時間を無くすためには、許容時間を増やすか必要時間を減らすか、両方を同時に実現することです。

許容時間とは顧客と約束した時間そのものです。許容時間は見積り交渉によって決定されますので、妥当な許容時間を確保するためには、しっかりした裏づけに基づいた説得力のある見積り回答が必要になってきます。

一方、必要時間を減らすためには、開発組織の業務遂行能力を上げること、すなわちその生産性を上げることが必要になってきます。生産性の向上は、業務の効率化および失敗の削減の二面に

よって決定されます。
以上を整理すると下記のようになります。

【許容時間の確保と必要時間の削減】

①許容時間の確保

時間確保のために、説得力のある見積りに基づき妥当な開発期間・費用の獲得を図ること。

②必要時間の削減（業務の効率化による）

・要求仕様を早期に凍結すること。
・顧客価値の重要度順に開発・評価を実行すること。
・改善活動を実施すること。仕事の効率化を図ること。今よりもっと効率的なやり方を考え実行すること。無駄の排除。不要なことや重複した業務をなくすこと。
・データやドキュメントに基づいた合理的な仕事の進め方や適切な段取りを行うこと。
・顧客間や社内におけるコミュニケーションを良くし手戻りを防ぐこと。
・目的や意味が正しく伝わるような作業指示を行い、無駄な作業や手戻りを防ぐこと。
・柔軟な設計に努め、将来の顧客要求にも少ない時間で対応できるようにしておくこと。
・正確な開発進捗の把握のために、量の管理だけではなく、質の管理も行うこと。
・割り込み仕事は優先順位の判断により対応し時間効率性の低下を防ぐこと。

③必要時間の削減（失敗の削減による）

・失敗の原因になりそうなリスクの回避・排除を事前に実行すること。
・中間レビューは結果として時間の節約に有効であり、レビューの精度を上げる。
・テスト漏れなどによる業務の後戻りは仕事のスピードを落とす無駄なものであり、モレを取り込んだテストパターン表やその表の流用は効果的である。
・求められている成果を確認せずにあいまいな仕事をすると手戻りの原因となる。
・完了期日を確認しないまま仕事を進めてはいけない。

許容時間確保、必要時間の削減（業務の効率化・失敗の削減）はそれぞれに相乗効果をもたらします。

第10章 ドキュメントのメソッド

メソッド＃1 【文書作成のポイント①】

1. 受け取る側の利益・心情を優先すること

①まず結論の記述から始めること。
忙しい相手はまず結論を知りたいはずですから、相手が持っていると思われる要望や疑問にまず答えることが大切です。

②自分の言いたいことを優先しないこと。
特に不始末の詫び状などでは言い訳から記述を始めるのは禁物です。

2. 正確性を確保すること

①文章の主題・件名と本文の内容を合致させること。
主題や件名が内容と合致していないと、受け取った相手は混乱します。特にメールの返信におけるRe:〇〇は要注意です。内容や話題が変わったら件名も新たに書き直す必要があります。

②文章において、「誰が、何を、いつまでに、どのように、どれくらい」等を明確に記述すること。

③数値やデータによって質や量を表現することで、文書の客観性を保つこと。

3. メリハリの利いた文章にすること

①結論に始まり、続いてその経緯について簡略にまとめること。
どんなに長い内容でも本文は1～2ページにまとめ、詳細内容は添付資料とする。

②同じ内容を、形を変えてダラダラと何度も繰り返さないこと。

③重要な事とそうでないことを区別すること。

メソッド＃2 【文書作成のポイント②】

①書く前に、相手に伝えたいことを重要なもの順にメモに書き出しておく。

②報告書は基本的に最初の1ページで全貌が分かるように書くこと。

③文章は重要なもの順に書く。例えば、最初に結論を記述し、次にその理由を説明する。

④一つの文章はできるだけ短くすること。複雑な内容は複数の文章に区切って書くこと。長文は難解になります。

⑤箇条書きは簡潔な表現に適している。

⑥数字・数値で論理的に記述すること。数字・数値は無用な誤解を生みません。

⑦あいまい表現はしない。例えば、”非常に”とか”おおよそ”などの形容詞的な表現は禁物です。

⑧文章だけで説明が難しいものは、マトリクス・図・表などを使用すること。

⑨技術用語はプロジェクト内や社内での統一用語を使用すること。

メソッド＃3 【ドキュメントの見える化のポイント】

①処理内容・テスト内容などの**マトリクス表**などによる 2 次元的な表記。

②フローチャート表記。

③論理記号、数学的表記の使用。

④異常系の記述の明文化。異常系に関するキーワードは次のようなものがあります。
排他処理不正、非同期制御不正、タイミングずれ、リトライ処理抜け・不足、タイマー値不正、レジストリ不正、設定値間矛盾、Mini・Max制御不正（メモリーリソース、データー長、伝文長、カウンター、ポインター）、メモリーリークなど。

⑤レスポンス条件・パフォーマンス条件の記述。

⑥ソフトウェア修正影響度表の作成および充実化

第11章　単体・結合テストのメソッド

1.　単体・結合テスト問題のメソッド

メソッド＃1 【単体テストの意味】

単体テストの対象は、関数やメソッドなど、個々のモジュールである。プログラム（モジュール）が一通り完成したところで、詳細設計通りに実装できているかを確認する。単純な機能面だけでなく、ロジックは間違っていないか、既定のコーディング・ルールや変数のネーミング・ルールに沿っているかなど、ソースコード・レベルでチェックする。このようなモジュールの中身までを検証するテストは、「ホワイトボックス・テスト」と呼ばれています。

メソッド＃2 【単体テスト用チェックリスト作成に関するポイント】

単体テストチェックリストの基である詳細設計書をプログラム作成前にきちんと作成しておくこと。

さらに詳細設計書の精度は基本設計書の精度に左右され、基本設計書の精度は要求仕様書の精度に依存しています。これらの上流工程における技術ドキュメントが整備されていなければテスト工程における効果的なチェックリストの作成は困難です。単にチェックリストの流用とか共通フォーマットの採用で解決できる問題ではありません。単体テスト項目の共通部のフォーマット化に関しては部分的な効果は期待できますので実行されると良いでしょう。またチェックリストの流用についてはソースコードのコピペと同様に非常に危険性が高いため単純な流用は避けるべきです。

メソッド＃3 【単体・結合テストの効果的やり方】

①流用可能なチェックリストは流用する。但し、流用内容を良く見極めて単なるコピペによる過ちをしないこと。
②製造の初期段階でソースレビューを実施し、単体・結合テストの負担を軽くしておくこと。
③テスト項目は必ずテスト仕様書を作成して行うこと。
④製造時のデバッグの質を上げること。
⑤テスト実施項目の実施の要・不要を精査すること。
⑥設計書の各機能項目とテスト仕様書の各テスト項目の“対応ひも付け”が明確に分かるようにしておくこと。
⑦テスト結果のエビデンス（印刷媒体）取りや整理は電子化・自動化ツールにて行うこと。
⑧自動化できるテストは全て自動化すること。
⑨テスト結果のレビューは必ず行うこと。
⑩日次会議にて毎日の不具合傾向を監視し、注力が必要な項目とそうでないものを峻別すること。

メソッド＃4 【結合テスト・システムテストの意味】

単体テストが済んだモジュールを組み合わせたサブシステムに対してテストするのが、結合テストである。更に、結合テストをクリアしたサブシステムを組み合わせ、最終的なシステムとなったものを検証するのが、システムテストである。

結合テスト・システムテストでは、基本設計通りに実装できているかをチェックするが、プログラムの中身は見ない。あくまで、与えた操作／入力データに対して、想定した動きをするかどうかを確認する。中身をブラックボックスとしてとらえるため、「ブラックボックス・テスト」と呼ばれる。

開発費や期間の削減に迫られて単体テストや結合テストのスキップを要求するような一部のベンダー側担当者も存在するようですが、そのような理の通らない要求に従ってはいけないでしょう。そのような行為はいわゆる手抜きと言われるものであり、そのような要求をしてくるベンダー側の担当者に何故そのような要求をするのか、またそれにどのような正当な理由があるのかを問いかけて下さい。このような状況が組織的に広がっているのならベンダー側および下請け側双方の管理者にその実態を報告すべきです。

メソッド＃5 【機能チェックの基本】

①あるべき入力条件に基づいているか。
②処理ロジックは合理的かつ妥当であるか。
③期待される出力が得られるか。
④複数の機能がある場合は、各機能の重要度順にレビューをすること。
上記の四点だけです。

メソッド＃6 【MIN／MAX制御の閾値】

データ処理に関するMIN／MAX制御の閾値には次のようなものがあります。
①メモリーリソース ②データ長 ③伝文長 ④カウンターサイズ ⑤ポインター値

これらのデータに関しては事前に設計書等に一覧表として整理した形で記載されている必要があります。この一覧表を元に、今回の開発に影響しているものは全てレビュー項目および評価チェック項目に加える必要があります。

メソッド＃7 【その他の異常系チェック項目】

その他の異常系チェック項目としては、以下のようなものがあります。
①排他制御 ②非同期制御 ③処理タイミング ④リトライ処理 ⑤タイマー値
⑥レジストリ ⑦設定間矛盾
今回の開発に影響しているものは全てレビュー項目および評価チェック項目に加える必要があります。

メソッド＃8 【各工程別のミスに対する対策】

①要求仕様ミスに対しては要求仕様書内容の正確な記述、適切な時期における凍結および効果的仕様レビューや仕様検討が必要。
②基本設計ミスに対しては要求仕様の理解力の向上、基本設計書記述能力の向上および効果的基本設計レビューの実施が必要。
③詳細設計書ミスに対しては基本設計書の理解能力の向上、詳細設計書の記述能力の向上および効果的詳細設計レビューの実施が必要。
④製造ミスに対しては、言語能力の向上、プログラミング技術の向上および効果的ソースコードレビューの実施が必要。

大雑把な原因だけではいつまでたっても数値に基づく効果的な改善は望めません。単体テストにおける不具合の原因を単に、ソースコードのレビュー不足、製造担当者のコーディングミス、単体試験仕様書の作成時期の遅れなどに求めることはできません。単体テスト・結合テスト・総合テストのすべてのテストにおける不具合の真因を工程別および要因別に記録・分析を行い大きな弱点の順に正しい対策を実行する必要があります。

2.　単体・結合テスト問題の解説

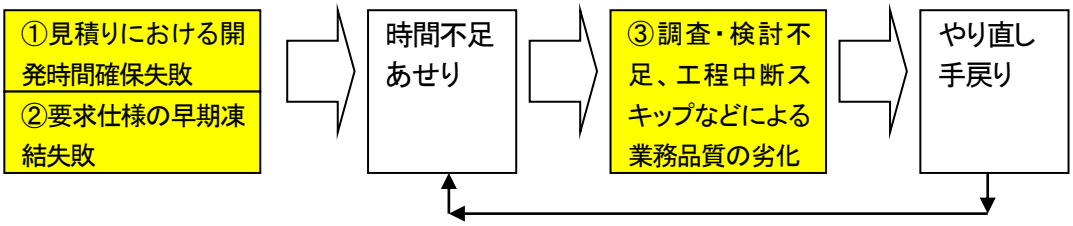
単体テストおよび結合テストにおいて提起された問題をその起因別に示すと次のようになります。

①時間不足に関するもの　9件、41%
②時間・時期の同期不良、実行すべきタイミングの失敗に関するもの　3件、14%
③前工程の不良によるもの　5件、23%
④不適切な実行内容に関するもの　5件、23%
⑤その他(コミュニケーション不良、連携不足)　2件、9%

(注. 件数データおよび%は本問題に関する過去の相談データから抽出したもの。)

時間に関する①と②の合計は12件、55%を占めており、また要求仕様定義工程・設計製造工程における大部分の問題が時間不足の問題に起因していたことから、前工程における不良の件数も含めると、時間に起因する問題の総数は17件、77%にも達しています。また時間不足によるあせりは必要な手順のスキップや不十分な検討・調査行為を生み、結果各工程における不適切な業務内容の原因にもなっています。

時間不足の二大元凶は、見積りにおける必要時間の確保失敗および要求仕様の早期凍結失敗にあります。この二大元凶によって、時間の不足はあせりを生み、その後の設計・製造・評価工程においては調査検討不足および必要な工程の中断ないしはスキップを招き、その結果やり直し、手戻りのために更に多くの時間を失ってしまうという悪循環に陥っています。図式化すると次のようになります。



①見積りにおける開発時間確保の失敗は、開発に必要な絶対的時間を失わせます。
②要求仕様の早期凍結失敗は、全開発工程の生産性、すなわちスピードと正確性を失わせます。
③調査・検討不足、工程中断・スキップなどによる業務品質の劣化は、全開発工程の正確性とスピードを失わせます

上記3点が我々から必要な時間を奪う3大元凶であると言えます。

我々開発組織は、この三点の元凶に対する対策を常に同時並行的に実行し続けない限り時間不足の悪循環から逃れる方法はありません。

第12章　モチベーションのメソッド

1.　モチベーション問題のメソッド

メソッド＃1 【苦手の克服法】

①苦手な仕事の項目を具体的に列挙してみる。
②列挙した苦手な項目にそれぞれ苦手の重み点数を3点法でつける（大の苦手に1点、中の苦手に2点、小の苦手に3点を配点）。
③列挙した苦手な項目にそれぞれ仕事の重要度の重み点数を3点法でつける（最重要業務に3点、中重要業務に2点、小重要業務に1点を配点）。
④「苦手の重み」と「仕事の重要度の重み」の点数を掛け算する。
⑤掛けた点数の高いものから順に克服していくこと。

上記は苦手意識が低く重要度の高い業務が最高点になり、苦手意識が強く重要度の低い業務が最低点となります。すなわち、実行しやすく効果の高いもの順に苦手を克服する方法です。どうせ、いつかは越えなければならない壁ですから、さっさと取り組んだほうが気も楽になり効果も上がることでしょう。詳細に分析したいのなら5点法でやってもいいですし、とりあえず苦手意識のランキングの下位のものから克服していくのも良いでしょう。

メソッド＃2 【チームの士気低下の原因】

①やらされ意識で行っている仕事

受身の姿勢で行っている仕事はいつしかルーチンワーク的な単純作業と誤認され、面白くない仕事と感じられ、言われたことだけを実施する仕事になってしまいます。

②目標が設定されていない仕事

目標がない仕事は、すなわち先が見えない仕事であり、どこまで頑張れば良いのか分からないため気力・体力の維持ができなくなる。

③過酷なスケジュール（長時間・長期間残業）

受身の姿勢および無計画と無目標が過酷な長時間労働を生む本当の原因です。

④チームプレーの欠如

チーム内で助け合いが行われない。リーダーの率先垂範が行われない。適材適所に欠ける。特定の人に負荷が偏っている。日次会議など頻繁な情報共有・目的共有・目標共有・問題共有が行われない。これらの問題のインパクトの大きいと思われる順に対策を実行すれば自ずと道は開けてくるでしょう。

2.　モチベーション問題の解説

モチベーションの問題は、自分の行動に有意義な意味を見出せない場合におこる感情反応だと言えます。やる意味が分からないものに対して人は行動を起こさないものです。

モチベーションに関する感情反応のパターンは次のようです。

（不利な状況　⇔　有利な状況　＊期待成果）

○　面白くない　⇔　面白い　＊知的興味の満足（心的満足感）
○　困難　⇔　簡単　＊労力対効果の最大化（利益獲得）
○　強制（受身的）　⇔　自主的（自発的）　＊自由性の確保（心身的満足感）
○　約束違反　⇔　約束遵守　＊行為を完遂可能とする条件の履行（成功の条件）
○　出口（目標）が見えない　⇔出口（目標）が見える　＊安心感の確保（心身的満足感）
○　期待はずれ　⇔　期待相応　＊行為を完遂可能とする条件の履行（成功の条件）

人の行動におけるモチベーションを左右するものは、「期待する獲得利益」および「人の安心・安全の保障」の二点に集約されると思われます。

【モチベーションの二つの構成要素】
①期待する獲得利益　②人の安心・安全の保障

仕事は常に限られた条件や環境のもとで遂行されなければならない事、さらに仕事を遂行する人間においてもさまざまな地位・能力・感性の差がある事を考えると、我々は、常に不利な条件の下で行動せざるを得ない状況にあることが分かります。期待する利益を獲得し、安心・安全を手に入れるためには、面白くないことを面白いことに変え、困難なことを緩和させ、強制によらず自主的に行動し、相互義務の約束を果たし、見えない目標を見えるようにし、期待相応の結果を生むための努力が必要であるということでしょう。

その意味で、モチベーションは他から与えられるものではなく、一つずつ自分および仲間との連帯の力で獲得して行かなければならないものだということが分かります。

第13章　総合・運用テストのメソッド

1.　総合・運用テスト問題のメソッド

メソッド＃1 【総合テスト（システムテスト）の特徴】

①開発の視点に沿ったテストです。すなわちテストの目的は、規定された仕様通りにシステムが動作するのかを確認することです。

②テストの準拠資料としては要求仕様書および設計書に基づいた総合テスト仕様書・総合評価チェックリストが使用されます。

③テスト方法は、操作手順に従って、所定の入力条件に対して期待される出力が得られることを確認することです。

メソッド＃2 【運用テスト（シナリオテスト）の特徴】

①顧客の視点に沿ったテストです。すなわちテストの目的は、顧客の期待する実運用の役割通りにシステムが動作するのかを確認することです。

②テストの準拠資料としては顧客の実運用を反映したシナリオテスト資料が使用されます。シナリオは一般的に顧客またはベンダー側SEによって作成される場合が多いでしょう。

③テスト方法としては、顧客の実運用に従った、一日の運用シナリオの実施、週次処理・月次処理・年次処理などを実行し、顧客が期待するデータ処理が行われることを確認することです。

一般的に総合テストと運用テスト両方のテストが行われることは少なく、総合テストが最終的なテストになる場合が多いと思われます。完全な運用テストが実施されるのは新規システム開発時で、更に顧客において運用テストを実施する能力がある場合に限られています。しかしながら総合テスト実施の場合だけにおいても、そのテストを更に有効にするためには顧客視点のテストも取り込んだテストを実施する必要性があります。総合テストにおける運用テストを実行するためには、顧客の実際の現場で使用されている全ての入力業務、入力バーコード、設定データを使用して、一日の業務フローを回し、週次ないしは月次処理や年次処理を回すテストを行う必要があります。なお顧客ごとに業務運用フローや入力データ・設定データは異なっていますので、ベンダーにおいては日ごろから各顧客におけるこれらの情報収集に努め資料の整備・作成を行っておく必要があります。下請け側としてはベンダー側におけるこれらの資料の欠落に気がついたら注意を促し、有効な総合テストを実行するように要求する必要があるでしょう。

メソッド＃3 【影響度表作成のための準備活動】

単体テスト、結合テスト、総合テストにおいて発見された他のモジュールに影響した不具合ならびに影響が及んでいると気づいたことを、全のテスト実行者が協力して影響度表に書き込むことを継続して実行することが必要です。当初は大雑把な影響度表しかなかった場合でも、派生開発の都度このことを実行していれば、完全な設計書がない場合でも、回数を重ねるごとに非常に精度の高い影響度表になっていくでしょう。開発チームにおける単体・結合テストおよび評価チームにおける総合テストにおいて両チームが連携してこの活動を地道に実行する必要があります。

メソッド＃4 【影響度表の作成方法】

①見積り時における実装調査において知りえた影響情報を記録しておくこと。
②設計作業時において知りえた影響情報を記録しておくこと。
③製造時において知りえた影響情報を記録しておくこと。
④デバッグ時において知りえた影響情報を記録しておくこと。
⑤単体テスト時において知りえた影響情報を記録しておくこと。
⑥結合テスト時において知りえた影響情報を記録しておくこと。
⑦総合テスト時において知りえた影響情報を記録しておくこと。
⑧市場障害対応時において知りえた影響情報を記録しておくこと。

メソッド＃5 【類似不具合の発生原因と防止策】

①不具合修正時に表面的な原因の修正のみしか実行しておらず、発生の真因の特定およびその恒久的な歯止めや解消を実行していないこと。
②過去の不具合情報を検索する仕組みや手段を持っていないこと。
③過去の不具合に対する組織的な取り組みを行っていないこと。
④不具合情報をレビューに生かせないこと。

例えば①に関しては、不具合部分のプログラム修正を行うだけではなく、問題の原因を単に「製造ミス」や「設計ミス」というレベルにとどめず、なぜ製造ミスや設計ミスをしたのかというレベルにまで踏み込む必要があります。真因はメンバー間の仕事の負荷の不均衡、知識不足、情報不足などでしょう。それらの真因に対する恒久的な対策を実行しなければ問題は解消されません。さらに関連ドキュメントの全ての修正も行わなければ、不良ドキュメントによる同じ不具合を何度も繰り返すことになります。

②に関しては、過去の不具合を検索するデータベースなどの仕組みを保有・運用していなければ膨大な不具合の中から自分が参考にしたい不具合を発見することは全く困難でしょう。

③に関しては、以上の取り組みを個人的に行ったとしても大きな効果はまったく望めないでしょう。組織的な仕組みやルールのもとにこれらの活動を着実に実行していく必要があります。

④に関しては、不具合の真因の特定と解消および不具合情報の容易な検索などの仕組みを構築しなければ、各レビューにそれらの情報を供給することはできません。

まずは、重大不具合や頻発する不具合に絞って上記の対策を実行されることを期待します。

メソッド＃6 【構成管理のポイント】

①人の判断や操作に依存する構成管理を極限まで減らすこと。
②開発中の成果物と中間リリースする成果物を管理するPCを分けること。フォルダを分けるだけではまだ人の判断ミスを誘発する可能性があります。
③リリース前に、作成したメディアによる実際のインストレーションおよび基本仕様の実装確認テストを行うこと。

2. 総合・運用テスト問題の解説

（注. 件数データおよび%は本問題に関する過去の相談データから抽出したもの。）

1） テスト方法の問題について

ここで提起された問題の原因は、単にテスト方法の是非という問題を越えて、大きく次の二つに分類できます。

1. 下請け評価チームの能力の低さ 7件、70%
2. ベンダー側の能力の劣化 2件、20%

下請け評価チームにおける問題は、評価レビューのやり方が分っていない、リーダーとしてやるべきことが分っていない、評価作業の意味・目的を理解していない、など評価業務能力が劣っていることを示しています。 一方、ベンダー側においては、総合テストや運用テストが実行できないという状況に陥っており、製造業務・設計業務についで評価業務についても実質的にアウトソーシング、すなわち他社依存になっているものと思われます。

2） テスト計画の問題について

提起されたテスト計画に関する問題の原因別順位は次の通りです。

①前工程の不良（仕様凍結遅延、単体テスト不良、リリースノート不備） 3件
②ドキュメントに拠らない作業指示・顧客要望打ち合わせ 2件
③レビュー能力不足、統合管理の欠落、組織編制知識不足、情緒的な思考、評価要員の指導・教育の未熟さ 各1件

上記の原因からみて、前工程の不良および評価チームの業務遂行能力の不足によって十分なテスト計画の策定ができていないことが分かります。

3） チェックリスト・評価用ドキュメントの問題について

提起されたチェックリストなどの評価用ドキュメントの品質の悪さの原因は、低品質な要求仕様書・設計書・影響度表など前工程である仕様検討工程・設計工程の業務品質の悪さに起因しています。またこれらの設計工程の業務品質の悪さは、見積りにおける開発時間確保の失敗、要求仕様の早期凍結失敗による時間不足に起因しています。

4） 不具合対応問題について

不適切な不具合対応問題の原因は下記の二つの群に分けることができます。

①感覚・認識・態度・常識に関するもの

- 不具合に対する感覚の鈍さ 4件
- 総合評価はバグ出し工程ではなく、仕様確認工程であるという認識の不足 1件
- 消極的・受身的な姿勢 1件
- 一般常識の不足 1件

全体の30%を占めているのは、不具合に対する感覚の鈍さです。不具合の発見を主要業務としている評価チームにおいてこの問題は致命的とも言えます。この問題は開発チームにおいても同様な状況だと思われます。いつもバグばかりに追われてしまった結果、”バグ慣れ”してしまったのでしょうか。バグ慣れという情緒的な心理状態は今すぐにでも改善する必要があります。

②不具合に対する実務能力の不足

・毎日ベースの不具合・リスク情報登録の未実行　1件
・恒久対策未実施、不具合検索システムの未保有、レビューに未組込み、組織的未対応　各1件
・影響度表が作成できない　1件
・単体テストのスキップ　1件
・仕様書・設計書等のドキュメントの不備　1件

評価チームにおいても実行可能なものは評価チームですぐに実施すべきであり、上記の全ての問題は、開発チームにおいても必ず実施すべき問題です。

5） ベンダー側問題について

ベンダー側の問題は、長年の外注依存、社内人員の軽量化、近年のオフショア化などの進展により、自己によるコントロールの効いたアウトソーシングが、義務と責任の放棄である丸投げ的な発注に変質してきたことによるベンダー自身におけるプロジェクトの統合的管理能力・開発能力・評価能力などの劣化に起因したものです。

一方、下請け側の問題は、長年の元請け依存による消極的な受身の姿勢、すなわち言われた通りにしか実行できないという姿勢から脱却できないことに原因があります。

共に解決が困難な問題ですが、ベンダー側においては自身によるコントロールを取り戻すこと、すなわち統合プロジェクト管理の実行、要件定義能力の強化などが製品の利益および品質につながるということを再認識し、組織の筋肉質化を図る必要があります。一方、下請け側においては、ベンダー側が失った能力を補完すべくプロジェクトマネジメント能力・要件定義能力・設計能力の強化により、自律した開発組織として生まれ変わる必要があるでしょう。

6） 総合・運用テスト問題分析のまとめ

上記の問題分析をまとめると次のようになります。

①ベンダー側においては既に評価業務についての実行能力は大幅に低下していおり、また下請け評価チームの評価業務能力が低いため、適正な評価業務が遂行されていない。
②仕様検討工程・設計製造工程の不良および評価業務遂行能力の不足により十分なテスト計画の策定ができていない。
③仕様検討工程・設計製造工程の不良により十分なチェックリスト・評価用ドキュメントが作成できない。
④不具合に対する感覚が鈍いこと、不具合に対する実務能力の不足により、適切な不具合対応ができていない。
⑤ベンダー側においては、劣化した開発能力の再生を図るために、統合プロジェクト管理の実行および要件定義能力の強化が必要であり、下請け側においては、自律した開発組織に生まれ変わるために、プロジェクトマネジメント能力・要件定義能力・設計能力の強化が必要です。

第14章　本質を見極めるメソッド

1.　本質見極め問題のメソッド

メソッド＃1 【数学の考え方17か条】（秋山仁、東京理科大学教授）

①分類・整理しよう

10進法、12進法、60進法、2進法、これらの数の表し方は、それぞれ、10、12、60、2というまとまりに数を分類・整理して、表されている。
分類・整理することで対象物の特徴が明確になり、対象物間の相関関係も明確になる。

②図や表にしよう

図や表にすることで、それまで見えなかったことが見えてくるようになり、複雑な事象も単純な事象の組み合わせであることが判明する。

③簡単な模型をつくろう

図を書いたり、模型を作るなどして考えると、難しいことが考えやすくなったりする。ハードウェアにおけるモックアップやソフトウェアにおけるプロトタイプなどがそれである。

④基準をそろえよう

基準をそろえるとは、（1）ものさし・単位をそろえる、（2）基点を定める、ということであり、ばらばらの基準であったものが統一基準のもとに分かりやすくなる。

⑤数学（ロジック）の言葉で表そう

問題を方程式で表すということ。たとえば4y＝600。自然言語で長々と問題事象を記述しても見えなかったものが一目瞭然となる。

⑥小さな例で試してみよう

解けそうにもない複雑な問題は、条件を易しくした例で試して見れば意外と解ける場合も多い。

⑦難しい問題は分割しよう

難しい問題は、解けそうな大きさに分割して考えること。複数の要因が重なった問題は、要因ごとに分割して答えを積み上げることで全体としての解が得られる場合もある。

⑧必要条件で絞り込もう

必要条件とは、何かが成り立つために必要な条件のこと。複数の必要条件を出していくことで、問題を絞り込み、最終的に解を得る。

⑨特定の要素に注目しよう

データを整理し、それから何らかの傾向を読み取るためには、特定の要素に注目すること。特定の要素として、平均値・中央値・標準偏差などがある。

⑩視点を変えよう

だまし絵などに見られるように一見してある物に見えるものも、異なった角度から見た場合違うものに見える場合がある。しかし、その実体は同一である。一見して不明なものも違う視点で見れば、自明のものである場合がある。

⑪逆に考えてみよう

見えている姿が裏向きのため理解できなかったものが、それを反転しただけで理解できる姿が見えてくる場合もある。自分が持っている前提条件や**思い込み**を一たん否定したアプローチを取ってみることも有効だろう。

⑫操作は1つずつ片付けよう

難しい問題を考えるとき、いくつもの操作を同時に考えると混乱する。そのような場合は、1つずつ手順を踏んで考えよう。

⑬知っている事実を活用しよう

長方形の面積＝横x縦を知っていれば、その延長問題として三角形の面積＝底辺x高さ÷2が分かるように、すでに持っている答えを活用してより複雑な問題が解ける。

⑭規則性を探そう

例えば、バーコード。13桁の数の中にも規則性が隠れている。この数字は、商品に関する情報を表している。最初の2桁は国名、次の5桁がメーカー、さらに次の5桁が商品名を表している。そして、最後の1桁はチェックデジットと言い、バーコードの読み取りミスを防ぐ役割をしている。バーコードに書かれた数に注目してみよう。

（左から奇数桁目の数の和）＋3×（左から偶数桁目の数の和）＝10の倍数、となるように、チェックデジットは決められている。例えば「4983248006644」のバーコードを例に計算してみると、(4+8+2+8+0+6+4)+3×(9+3+4+0+6+4)=32+3×26=110 となり、10の倍数になっていることがわかる。つまり、バーコードを読み取ったときに10の倍数になっていなければ、どこかで読み取りミスをしている、ということがわかる。

⑮対応をつけて考えよう

「対応をつける」とは、何かと何かをペアにして考えるということです。例えば、目の前に乱雑に置いてある数百個のボールを、「だぶらず、漏らさず」数える場合、番号を書いたシールを一つずつ貼っていけばうまくいくでしょう。物の数え方の基本原則は「だぶらず、漏らさず」であり、そのためにボールとシールをペアにすることにより容易にその目的は達成されるでしょう。

⑯自然からヒントを得よう

同じ質量のもので表面積が最も小さい形は何かを考えた場合、自然界における水滴や溶けた水銀などから類推すれば球ということが自然と判明してくるでしょう。

⑰部分から全体を把握しよう

さまざまなデータから全体の傾向や状態を推測できる。例えば、「標本調査」や標本調査をするときの標本の選び方の「無作為抽出」や，標本調査によって母集団の大きさを推定する「比率の推定」などの手法がある。

2. 本質の見極め方に関する問題の解説

ものごとの本質は、普通目の前に現れた現象の背後に隠れており容易には見破ることができません。ものごとの本質の見極めを妨げる要因としては次のものが考えられます。

①思い込み
②ものごとの背景、目的、意味の理解不足
③事象分析力の低さ
④コミュニケーション能力の低さ
⑤役割認識の不足
⑥時間感覚の欠如
⑦心身の疲労。

これらの阻害要因を解消するためには次のような思考および行動が必要となります。

【本質を見極めるために必要な思考や行動】

①仕事や仕様の意味を知ること。
②仕事や仕様の背景を知ること。
③仕事や仕様の目的を知ること。
④仕事や仕様の事実の確認を根拠や証拠に基づいた文書（ドキュメント・ベース）で行うこと。
⑤複数の視点で物事を見て分析すること。

⑥口頭・文書ともに必ず誤りが含まれるという問題意識を持ち、“何故？”の疑問提起を行うこと。
⑦自己の役割・責任の範囲を知ること。
⑧仕事の許容時間・必要時間を把握しておくこと。
⑨心身の限界を知っておくこと。

第15章　人材育成のメソッド

1.　人材育成問題のメソッド

メソッド＃1 【仕事を任せる場合のポイント】

①上司においては、任せる仕事の全体像を完全に把握しておくこと。
　丸投げをしてはいけない。
②指示内容は口頭のみならず過不足のないドキュメントにより、漏れや誤解を防止すること。
③部下の能力や気質を常に把握しておき、少しずつ負荷を増していくことでその成長を図ること。
④部下が失敗しないように、毎日問題点のヒアリングおよび助言などのフォローを行うこと。
⑤大きな失敗を防ぐために、事前に適時のレスキューを計画しておくこと。

（自分自身が困難な仕事に直面した場合）

上記のことは、自分自身が他の上司から困難な仕事をまかせられた場合についても全く同様のことが言えるということを理解しておく必要があります。すなわち、困難な仕事を目の前にした場合、失敗を大いに恐れてください。そして失敗を防ぐためのあらゆる事前準備と対策を実行してください。自分に足りない知恵や労力は他から調達することを考えてください。毎日、自分の抱える問題点の把握と解消に努めてください。万一、失敗に至りそうになった場合は間に合うタイミングでの支援要請が必要です。

メソッド＃2 【社員育成の場】

①チーム内における日次情報共有会議
②開発着手前における仕様の説明時
③仕様調査時における不明点および疑問点に関するQ＆Aのやりとり時
④レビュー時（設計、コードレビュー、単体テスト、結合テスト、総合テスト）
⑤不具合修正時
⑥プロジェクト終了時の振り返り会議

上記のすべてのプロセスを確実に実行していれば、自動的にメンバーの教育が実行されることになります。

メソッド＃3 【自助努力による人材教育に必要なこと】

①まずリーダー自身のスキルアップを図ること。
　優れたリーダーが存在しなければ、メンバーの育成は困難です。優れたリーダーは下記の②および③を率先して実行する中で養成されて行きます。

②開発行為の全てが教育の場であると考えること。
　日次情報共有会議、仕様検討時、仕様説明時、すべてのレビュー時、不具合修正時、振り返り会議時の優れた行為は全ての人材の最も良い教育の場となります。

③開発工程すべてにわたる改善活動の実施を行うこと。
　現状の各工程における問題の改善活動は開発の効率化、利益の向上、時間の創出の効果を生み出すと同時に、全メンバーの最も効果的な実地教育の場となります。

メソッド＃4 【作業指示のやり方およびフォローの仕方】

①作業指示者においては、事前に指示内容を完全に理解しておくこと。
②作業指示は口頭のみならず必ずドキュメント資料により行うこと。
③作業指示時に、相手の理解度をチェックすること。
④日次情報共有会議にて、メンバーの疑問点・問題点を聞き出し、適切な指導や助言を行うこと。

2. 部下育成問題の解説

部下育成の目的は、自分で考え自分で行動できる自律性のある部下を育てることにあると言えます。すなわちものごとを良く認識し、その判断に基づき有効な手段を用いて、効果的な行動をとることができる人材を育てるということです。部下の育成は、部下の問題であると言うよりも、その部下を育成する上司の問題であると言った方が良いでしょう。自律性のない上司には自律性のある部下の育成はできないでしょう。

提起された各問題を、自律性の構成要素である、認識の問題、手段の問題および行動の問題の三つの視点で見ると上司における人材育成の問題点は以下のようになります。

（注. 件数データは本問題に関する過去の相談データから抽出したもの。）

1） 認識の問題

人材育成に関する上司における認識の問題点は次の通りです。

①役割認識の不足 1件
②困難な仕事に対する恐怖感・不安感に対する誤認識 2件
③仕事の意味・本質・内容の把握力不足 2件
④改善活動の必要性認識不足 1件
⑤感情本位な仕事への取り組み方 1件
⑥失敗に関する理解不足 1件
⑦人員増＝業務品質向上という誤解 1件

2） 手段の問題

人材育成に関する上司における手段の問題は次の通りです。

①プロジェクト管理・遂行に関する実務知識・知恵の不足 2件
②適切なドキュメントの不足（ドキュメントベースの仕事の不足） 3件
③時間不足 2件
④チーム編成の知恵不足 1件

3） 行動の問題

人材育成に関する上司における行動の問題は次の通りです。

①行動選択の誤り 2件
②コミュニケーション能力の不足 8件

4） メンバーにおける行動の問題

全24題の問題の内、メンバーにおける問題は下記の3件のみです。これらの問題は上司側の問題でもあります。

①メンバーにおける自律性の欠如 1件
②メンバーにおける知的行動習慣の不足 1件
③メンバーにおける学習意欲の弱さ 1件

上記の問題の内、非常に件数が多かった、コミュニケーション能力の不足（8件）および適切なドキュメントの不足（3 件）については、リーダーおよび上司における重要課題であると認識する必要が

あります。全体として言えることは、人材育成の役割を担うリーダーや上司においては、自分自身における認識問題、手段問題、行動問題の解決、すなわち自律性の獲得を行うことが、直ちにメンバーの育成につながるという認識をもつ必要があります。

3. ノウハウ継承問題の解説

開発組織におけるノウハウの継承は、その個人および組織のスキルアップの最も基本的かつ基幹的な活動です。ノウハウの継承が行われない組織には成長も発展も望めないでしょう。

ノウハウの継承の有効性を妨げている継承のやり方の問題点は以下の通りです。

（注. 件数データおよび%は本問題に関する過去の相談データから抽出したもの。）

①ドキュメントによる継承が行われていない、もしくは不足している 21件、51%

主な問題点は次の通りです。

・ノウハウの継承が口頭のみでしか行われていないこと。
・仕事に関するコミュニケーションが口頭に依存しすぎていること。
・ドキュメントが必要になる前に用意されないこと。
・ドキュメントが常時更新されていないこと。
・同一種類のドキュメントが共通フォーマット化されていないこと。
・ガイドライン等による体系化がされていないこと。
・衆知を結集しノウハウ集を検索可能にするためのデータベースが存在しないこと。

②ノウハウについての情報共有認識の不足 7件、17%

主な問題点は次の通りです。

・自分の知りえた有用な情報を自己防衛的姿勢により他者に伝達しないこと。
・他の組織に対するノウハウの横展開が行われていないこと。

③業務知識および能力不足 5件、12%

主な問題点は次の通りです。

・業務知識の不足
・プロジェクトマネジメントの役割知識不足
・業務の緊急度・優先度の理解不足
・仕事の目的・意図・背景の理解不足

④ラップアップ（振り返り）の未実行 2件、5%

ラップアップ（振り返り）実行の必要性・義務感の不足。

⑤その他 6件、15%

支援・継承のかいのない相手に対する拒否感。

（注. 件数および%データはノウハウ継承に関する過去の相談データから抽出したもの。）

ノウハウ継承の最大の阻害要因はドキュメントによる継承が行われていないということにあります。このことはコミュニケーションにおける正確な情報の伝達における阻害要因でもあり、組織内で重要な情報を伝達するためには文書・書類などのドキュメントが絶対的に必要であるという認識が欠如していることを意味しています。さらにこの認識欠如の原因は、ドキュメントを作成する時間がないことおよび、他者との相互理解や相互関係をめんどうなことだと感じてしまう束縛感を嫌う姿勢がもたらしていると言えます。

時間不足の問題は開発問題の全てにおける最大の問題です。

おわりに

妥当な開発期間・開発費の獲得を大前提とし、開発対象物の明確化によって初めて適切な開発が可能になります。適切な開発は、妥当性に支えられた合理的なアプローチによって実現され、改善活動によって継続性を与えることが可能となります。永続性をもつ開発力は、失敗を最小化し、高品質製品を顧客のもとに届け、結果として我々の目標とするQCDを達成させるでしょう。

これらの優れた開発活動を支えるものは、言うまでもなく、個人および組織における健全なヒューマンスキルに他ならず、これらの人間能力は、その自律性を基本的な土台とし、妥当性と合理性の双方の力によって、柔軟な思考・行動を生み出し、個人ないしは組織間における相互義務および相互扶助の行動を促進するでしょう。最後に、これらの魅力的かつ人間的な思考・行動によって獲得された有形・無形の心的・物的な資産を、将来の自他に譲る行為によって、その個人ないしは組織に永続的な繁栄をもたらすことができると思われます。