項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
1	中核SEの離脱	業務設計の中核となっていた協力会社のSEが、結合テストの直前に入院してプロジェクトを離脱。 SEが担当していた部分の詳細設計書やソースコードがPCにもサーバーにもない	業務設計チームのリーダーとして、チームの進捗報告を定期的に行っており、単体テスト工程までは特に進捗上の問題も無く報告されていたし、進捗状況に関する質問にも的確に答えていたので、この中核SEを信頼して成果物の確認をしていなかった。 また、発注元とはプロジェクト全体の契約形態が請負契約か作業委託契約かで揉めて（下請けと孫請けの契約）おり、チームの報告を受けてその内容を確認する責任者がどちらかあいまいだった	・作業委託でも作業者の管理責任は受託側にあると認識。応援部隊を編成し、当該リーダーが実施する予定の作業を短期間で完了させた ・リーダーに社員を配置するとともにドキュメント・サーバーによる文書の一元管理体制へ移行した。この部分の結合テストは、後半に実施するよう再スケジュールした	・人員の投入（増員） ・構成管理（ドキュメント） ・再スケジュール	・構成管理のルールを決めること ・作業分担を明確にすること
2	仕様凍結の遅延	1カ月で仕様確定、1カ月で開発、1カ月でテストという3カ月工程で予定されていた案件が、仕様確定のための1カ月間、需要先と打合せができず、設計検討も含めて実質2カ月に工程が短縮された	アプリケーション開発を受注し、開発終了が3カ月後で確定していた。アプリケーションを販売する会社へのヒアリングで仕様を確定するはずが、販売先からの受注遅延により仕様の打合せを始めたのが2カ月目に入ってからとなり、開発期間を1カ月短縮せざるを得なかった	・小規模なシステムだったため、運用イメージを基に1カ月でプロトタイプを一式用意した ・2カ月目には客先で、プロトタイプを基にパーツ単位で仕様確定に持ち込み、追加変更削除の依頼に対応しながら開発と仕様確定を同時進行させた	・再スケジュール（リリース時期） ・機能削減交渉	・フィージビリティ・スタディ（実行可能性の検証） ・契約によって顧客を巻き込む（仕様確定するまで顧客と共同作業） ・分割契約にする ・委任契約にする
3	特定機能の進捗遅れ、品質劣化	機能変更のプロジェクトにおいて、ある重要な機能の業務ルールがドキュメント化されておらず、そのソースコードを修正できない（品質が悪化する）。当時の担当SEは他部門へ異動していた	ソースコードがスパゲティ状態（構造化されていない）で、保守できない。プログラムを作り直そうにも、業務ルールがドキュメント化されていない	・当時のSEを1カ月間引き戻し、新メンバーとペアでコーディング、テストさせる ・業務ルールは都度ドキュメント化（ソースコードの作り直しは実施せず） ・当該機能は2次リリースへ延期	・再スケジュール ・分割リリース ・業務を知っている人材を確保 ・共同作業によるスキル・トランスファ ・プログラムの影響範囲調査	・ドキュメント化の徹底 ・レビューの徹底 ・コーディング規約の策定と順守
4	検収時期に一括納品できず	契約書には仮納品の記載はないが、顧客から事前検証や研修のため、3カ月前に仮納品するように言われていた。だが、進捗が遅れてすべての業務を一括して納めることができなかった	顧客との約束がプロジェクト全員に共有されていない上、プロジェクト・マネージャは業務単位で完成した順にシステムを提供することで検収や研修に対応できると勝手に考えていた	・プロジェクトの進捗状況を顧客に説明し、顧客の検収、研修を開発状況に合わせて順次、実施してもらえるように依頼した。 ・顧客の上長に対して文面でのお詫びを行うとともに、経験が豊富で実績あるプロジェクト・マネージャを投入し、顧客交渉に当たらせた ・開発体制を強化するとともに品質管理チームを発足させ、遅れや品質の問題解決に当たらせた	・分割リリース ・プロジェクト・マネージャ交代 ・問題点の解決管理 ・品質測定	・顧客とのコミュニケーション ・作業分担の明確化 ・プロジェクト内部の情報共有
5	パッケージとカスタマイズのソースコード範囲が不明確	納品時に顧客へソースコードを開示することになっていたが、その範囲が不明確になった	パッケージ製品に対するカスタマイズとして開発を受注した。パッケージ製品のソースコードは公開不可で、カスタマイズ部分については顧客に開示する契約であったが、顧客要求に対応するために、カスタマイズがパッケージのコア機能にまで入ってしまい、顧客への開示範囲について不明確になってしまった	手を入れた部分がカスタマイズであると定義して顧客と調整し、プログラム単位で顧客に開示。コアソースの流出を防いだ	・顧客調整（契約内容）	・事前の契約確認（コア部分への修正に対する開示） ・プログラムの構造と権利範囲の明確化（著作権とコードの所有権）
6	テスト工程で設計仕様変更が多発	総合テスト工程に入り、画面設計、業務仕様、出力帳票などで設計変更する個所が多発し、納期遅れにつながった	・Webベースの新製品の開発で、新しいプロダクトの技術的な特徴、性能、制約条件などを正確・的確に把握するのが困難だった ・業務仕様においては、ユーザー要件の検討項目を膨らませ過ぎて仕様凍結が遅れ、仕様変更がテスト工程まで発生した	・プロトタイプの作成後にプロダクトの特性を見極め、計画値との差異を検証して対策を検討 ・顧客要求事項に関して、コスト、納期、品質面を考慮したシステム化対象範囲を明確にし合意する ・仕様凍結時期を限定する	・バグか仕様変更かの切り分け ・対応の優先順位付け ・仕様凍結時期の明確化	・分割契約 ・仕様凍結時期の明確化 ・十分な事前調査（制約、機能など）

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
7	開発側の運用イメージ不足と仕様説明不足	本番稼働後、Webメール・クライアントで文字化けが発生し、使いものにならないという指摘を受けた	メール・ヘッダーやエンコード手法はメール・クライアントごとに異なる場合がある。作成する機能の前提条件で、ある種のヘッダーと文字コードに限定した仕様としていたが、顧客は各種外部機関からメールを受信するため、提示している仕様前提ではメール・クライアントとして使いものにならなかった。仕様書は承認されていたが、実際には顧客はその意味が理解できていなかった	・顧客も仕様を承認している以上、仕様変更として対策を実施。対策期間は、旧システムとの並行運用時 ・開発担当が入手可能なフリーのメール・ソフトの各バージョンで機種依存文字を含むメールを送受信し、顧客が使用する想定範囲で動作するよう、一通りの実装と動作確認を実施した	・バグか仕様変更かの切り分け ・旧システムとの並行運用 ・バリエーション・テストによる問題点の把握	・現状システムの調査の徹底 ・現状運用の調査の徹底 ・十分な受け入れテストの実施
8	顧客の機能要件説明能力不足と開発側の顧客ニーズ把握ミス	顧客による動作確認段階に入ってから、仕様に対する苦情が多発した	実装した機能については仕様書に載せて説明していたが、実装していない機能については説明していなかった。顧客は旧システムの各種機能を踏まえた仕様になることを想定していたため、大幅な意識のずれが発生していた	・顧客も仕様を承認している以上、仕様変更として対策を実施 ・旧機能のうち、便利だと思っていた機能を利用者からヒアリングし、コストと開発期間で合意に達したものをすべて実装した	・仕様変更交渉の実施 ・追加実装機能の優先順位付け	・役割分担の明確化(仕様確定者)
9	業務設計の中心となる設計リーダーが倒れて復帰できない	業務を熟知し、プロジェクトの中心となっていた設計リーダーが過労で倒れた	あらゆる問題が1人の設計リーダーに回されていた。その設計リーダーは、部下への仕事の振り分け方があまりうまくなかった	・業務設計の中心となる設計リーダーがいなくなった場合は、そのリーダーが仕事を抱え込むタイプで状況がブラックボックスになっているか、あるいは仕事を共有するタイプで部下もある程度分かっているかによって対応が変わる ・前者であれば、設計リーダーがいなければ仕事を進められないため、根本的にスケジュールや体制を見直す ・後者であれば、サブリーダーをリーダーにしてチーム全体を上位スライドさせ、さらに人員強化で体制を組み直す	・再スケジュール ・人員投入	・工程ごとに成果物と課題管理 ・業務勉強会の実施 ・ユーザとの打ち合わせに開発者を出席させる ・徹底した勤務管理 ・作業分担の工夫
10	大幅な工数増加と工程遅延	見積もりと比較して、実際には大幅に工数が増加し、工程遅延となった	当初の見積もり(実績値とほぼ一致していた)では顧客の予算に合わず、顧客から圧縮を求められた。対象業務の知見に乏しく、受注しないと先がないとの危機感も加わり、顧客の「簡単にできる」という言葉に反論できず受け入れてしまった	・根拠の乏しい顧客の工数圧縮要請に安易に迎合しない ・類似プロジェクトを調査し、知見を十分活用する	・人員投入(工程回復を図る) ・顧客との価格調整	・過去の類似プロジェクトを調査して見積もりの妥当性を評価
11	大幅な工数増加による採算悪化	受注範囲のうち、初経験領域での大幅工数増によりプロジェクトの採算が大きく悪化した	初経験領域の受注に際して、体制の構築に手間取り、具体的な作業内容の検討に十分な時間を割けなかった。その状態で見積もりを出し、請負契約をしたため、作業範囲が肥大化しても追加費用の交渉ができなかった	・初経験領域での受注に関しては、受注範囲の明確化に十分な時間を割く ・仕様や作業範囲の不透明な場合、請負契約は極力避けて、委任契約といったリスクヘッジできる契約形態にする	・再スケジュール ・顧客との価格再交渉 ・人員投入	・初経験に対するリスク管理を十分行う ・作業範囲が分かるまでは委任契約にする ・不透明要素が高い場合は請負契約はしない ・初経験の場合は委任契約にする ・システムの重要度を考えた契約にする ・顧客とのリスクの共有とそれに見合った契約にする
12	大幅な工数増加と工程遅延	顧客からの納期要求、機能の追加・変更要求に対して反論できず、一方的に受け入れざるを得なかった結果、工数増加と工程遅延を招いた	仕様がはっきりしない状態で、過小見積もりしたため、プロジェクト・マネージャ、管理要員、チームリーダーの人数が足りなかった。そのため、十分な仕様変更管理ができなかった	・サブのプロジェクト・マネージャを投入し、管理要員、チームリーダーを追加した ・仕様の承認ルール、仕様追加・変更スキーム、費用精算ルールなど、顧客と必要なルールを取り決めた ・障害管理、リリース管理、構成管理など、プロジェクト・チームが順守すべき規約・標準類を整備し、順守の徹底を図った	・プロジェクト・マネージャ支援要員の投入 ・手順、ルールの確立	・手順、ルールを事前に定義しておく ・顧客とのルール、手順の合意 ・仕様確定の顧客との合意

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
13	納期に間に合わない	初めての業務で、当初想定していたより業務が複雑だった。そのうえ業務の有識者も少なく、約束した納期に間に合わないことが判明した	開発が進むにつれ、部門によるローカルなルールで業務を運営していることが明らかになり、当初の見積もりを大幅に超える開発規模になった。また、業務の有識者が少なく、業務を共通、統合、集約する提案を顧客にできず、顧客の要望も断れなかった	業務が想定以上に複雑で多岐にわたるため、運用が回る最小の業務に絞り込んで開発し、その他の業務は順次優先順位を付けて開発することにした	・顧客に協力を依頼 ・有識者の確保 ・人員投入 ・分割リリース ・再スケジュール	・初経験に対するリスク管理を十分行う ・仕様打ち合わせ会議に、業務をよく知っている業務部門の人に出席してもらう ・分割契約 ・顧客との協力体制の確立
14	仕様があいまいなままシステムを提供	初めての業務だったので、顧客が承認した詳細設計に基づいて開発したが、サービス開始後に不具合が多発した	部門で業務仕様に差があり、そのバリエーションは当初想定した以上だった。詳細設計書を承認した顧客も気づいていたが、途中で顧客側の担当者が変ってしまった。開発側も要員追加ができず、仕様を膨らまさないように、詳細設計通りで本当にいいのか、あえて確認しなかった	・部門ごとに業務仕様が異なることを顧客にも認識してもらい、仕様をすべて調査するよりも、運用で不具合を見つけ出すことにした。不具合を発見したら仕様を確定し、システムに反映した ・不具合に対して、運用支援で対応するチームと仕様を確定させてシステムを修復するチームを作り、本運用で発生した不具合に応じて順次修正することとした	・使ってみて不具合をたたき出す ・ドキュメントを残す ・修正作業の体制整備	・事前調査を十分に行って事前に仕様を明確にする
15	プロジェクト・マネージャが多忙	プロジェクト・マネージャの健康維持が課題になった（倒れてしまうと代役がいない状態）	プロジェクト・マネージャの能力、保有情報が突出し、過負荷となった	プロジェクト・マネージャのまわりに人材を増やし、軽微な雑用から、使いはしり、そのほか代理可能な業務を積極的に引き受け、プロジェクト・マネージャの負荷軽減を図った	・プロジェクト・マネージャ支援要員の追加	・勉強会の実施 ・作業分担の明確化 ・役割分担の明確化
16	共同受注案件での開発分担不明確による混乱	公募案件を共同提案で受注後、元請けとの分担が不明確となり混乱した	受注優先で共同体制を組むが、受注後の役割分担、予算配分の詰めがなく、受注後混乱した	受注優先であっても、受注後の役割分担を文書で明確にし、合意しておく。予算配分、責任分担を明確にしておく	・交渉（役割、費用）	・役割分担の明確化 ・予算配分と責任分担の明確化
17	要件定義が不十分で、テスト工程になってリリース必須機能の漏れが発覚	要件定義が不十分で、テスト工程になってからリリース必須機能がないことが発覚し、当初予定していた日のリリースが難しい状況となった	過去に経験したプロジェクトと類似だと判断したが、実際はそうではなかったという場合や、現行機能について十分な理解をしないまま要件定義を実施する場合に発生しやすい。要件定義が不十分であり、後工程で要求がたくさん上がり、工数が膨らんでしまったことにつながるためである	・開発体制を増強する。自社メンバーだけではなく、協力会社側にもメンバー増強を依頼する ・スコープを限定して一部機能をリリースする	・体制強化と人員投入 ・分割リリースの実施 ・顧客との仕様の再確認	・ドキュメント作成 ・工程区切りの評価 ・顧客によるレビュー ・事前調査を徹底する（過去の類似プロジェクトと本当に類似かどうか）
18	テストの進め方が分からず、テスト計画もできない	総合テストに入ったが、いままで小規模の開発しかやっていないメンバーばかりで大規模開発での総合テストのやり方を全く理解していなかった	経験不足、知識不足	・テストの進め方が分からない場合は、社内の類似の別プロジェクトのものを流用する ・大規模な総合テストを実施したことのある経験者を連れてきて計画を立てる	・有識者の確保 ・ルール、手順の入手（経験者による判断が必要）	・組織プロセスの定義（プロジェクト・マネージャの任命方法、要員、体制の確立など）
19	稼働1カ月半前でも結合・総合テスト計画が不十分	稼働1カ月半前になって、事業責任者（プロジェクト・マネージャの上長）から、「プロジェクトが危なそう」というアラームが出た。単体テスト中なのに、結合テスト、総合テストの計画が立っていなかった	元請け会社のプロジェクト・マネジメント不足、要員不足。開発委託先（3社）の責任範囲は、社内で行う単体テストまでであり、結合テスト（社間テスト）以降は、元請け会社が取りまとめる責任を負う	・委託先を含めて、メンバー全員を1カ所に集結 ・元請け会社の体制強化。委託先のキーパーソンを元請けの一員としてテスト計画を立案 ・毎朝、プロジェクト・マネージャと実務責任者のミーティングを実施 ・1次リリースと2次リリースに分割（一部機能はリリースを延期）	・メンバーの集結と一体感の醸成 ・PMOの投入 ・テスト計画とテスト環境整備 ・再スケジュール ・プロジェクト体制の強化	・計画重視型のプロジェクト推進 ・プロジェクト管理体制の整備 ・工程区切りでの評価
20	大幅な工数増加による採算悪化	移行作業に関して、大幅な工数増によりプロジェクトの採算が大きく悪化した	移行作業に関して作業量を絞りきれないまま、顧客の要請に応じて請負契約で受注。実際には工数が予定より大幅に増加した。 受注に際して前提条件を付けたものの、明確ではなかったため、追加費用の請求に応じてもらえなかった	仕様や作業範囲の不透明な場合、請負契約は極力避けて、委任契約といったリスクヘッジできる契約形態にする	・顧客との価格再交渉 ・トップ会談での価格調整	・リスクに応じた契約形態 ・リスクの高いことを相手に伝えて、理解してもらえない場合は受けない

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
21	共通ライブラリのスケジュール遅れ、品質不良	アプリケーションの開発と並行して、共通ライブラリを開発していたが、共通ライブラリの開発が遅れ、アプリケーションの開発スケジュールにも大きな影響が出た	クリティカルパスである共通ライブラリの開発スケジュールの管理不良と、機能要件の定義不足(アプリケーションと共通ライブラリの機能分担の責任があいまい)	・共通ライブラリのマニュアル整備(機能要件の定義を追記)と開発体制の強化 ・アプリケーションの単体テストはスタブで実施	・人員投入 ・設計の見直し ・テスト実施方法の工夫	・ハイスキル要員の確保 ・クリティカルパスを意識した進捗管理 ・工程管理基準の定義
22	見積もり時の前提想定不足	顧客先での立会いや運用サポートのSE作業が大量に発生し、想定以上のコストがかかった	契約時に作業場所や作業内容が明確化されていなかった。そのため、交通費は見積もりに入っていないが見積もり範囲外と宣言できなかった。SE作業で依頼される範囲が不明確なため、依頼された作業量が予定コストを超過しても、見積もり範囲外と宣言できなかった	顧客調整により本契約範囲内での作業範囲を明確化し、依頼を受け続ける体制から作業終了目標をはっきりとさせるようにした	・顧客調整 ・作業範囲の明確化	・契約作業範囲の明確化 ・他プロジェクトの見積もり例の参照
23	メーカー提供プロダクトの不具合でシステムが動かない	メーカー提供のプロダクトを大規模に利用したが、メモリー・リークが多発して実用に耐えない	プロトタイプを作成して、技術的な見極めを完了したと思っていたが、開発で本格的に利用してみるとメモリー使用量が徐々に上がっていき、最後はオーバーフローしてしまった。大量データ登録での確認不足が原因だった	この状況における対策は、何としても解決するか、早くあきらめて代替機能を探すことに重点を置くかのどちらか。見込みがない、あるいは解決できる自信がない場合は早くあきらめることが肝心。パッケージの調査をして、対応方針を検討し、顧客と調整する	・ベンダーを巻き込む ・パッケージの選び直し ・スクラッチで作り直す ・スケジュール再調整 ・運用回避検討(日次のシステム再起動など)	・保守契約を結んでおく(保証の範囲を明確にする) ・パッケージの調査を事前にしっかりやっておく(実績を調査する、機能の網羅性、事前のテスト) ・パッケージが顧客からの指定の場合は、責任範囲を明確にする ・事前に実績のあるパッケージを調査・準備しておく ・パッケージの使用経験者を確保する ・パッケージ提供会社の調査(安定性)
24	システムの品質が悪い	スケジュールに追われ、低品質のものをリリースし、障害対応でドタバタする、という負のループから抜け出せない	品質の作り込み(ドキュメント作成、レビュー)や十分なテストよりも、スケジュール/コストを優先していた。開発担当者が、当該システムで求められる品質に対する認識の甘さがあった	・品質管理を中心に管理要員を追加投入 ・リリース・スケジュール・ルール、障害対応ルールなどを整備し、その順守を周知徹底した	・品質分析の実施(弱点部位の特定) ・人員投入	・品質要求の明確化 ・テストの十分な実施 ・リリース品質基準の明確化 ・品質に関して顧客と共通認識を持つ
25	システムの品質が悪い	結合テスト時、単体テストレベルのバグが多数発生。結合テストを中断・延期し、工程遅延を招いた	追加機能を開発していた際、メンテナンス性の向上を図るため、既存プログラムの修正も実施。その修正部分のテストを十分実施しないまま結合テストを実施した	・プログラムの単体テストを再実施 ・第三者によるソースコード・レビューを実施	・テストの再実施 ・ソースコード・レビューの実施	・機能追加ルールの明確化 ・機能追加のリスクを顧客と合意
26	大量印刷時間が長い	クライアント/サーバー・システムで開発した業務パッケージをWeb化した。プリント・サーバーの印刷処理方式を自社で構築したが、総合テストの大量印刷処理で想定を超える処理時間がかかった	Webアプリケーション・サーバーとプリント・サーバー間のデータ受け渡し処理で、ミドルウェアの共通資源が不足し、処理待ちが発生していた	ミドルウェアの共通資源の設定を見直すとともに、新たに大量印刷の処理方式を構築し、印刷量の多い業務は新しい方式に変更した	・類似個所の見直し ・処理方式の見直し	・事前調査(技術調査) ・事前の負荷テストの実施 ・性能要求の検討と明確化
27	ユーザーの研修で急に処理が遅くなる	ユーザーの業務運用研修で、同じ業務処理を同時に行うと処理時間が大幅に遅くなる	認証サーバー、DBサーバーへの処理が集中したうえ、共通資源の排他制御の不具合、SQL文の使い方の誤りによる無駄なDBアクセスが多かった	・共通資源の排他制御方式を見直し、トランザクションが集中する業務プログラムを優先して修正 ・DBアクセス・ログを解析し、特に無駄な処理を行っている業務プログラムのSQL文を修正	・専門家を連れてくる(方式の見直し)	・事前調査(技術調査) ・有識者による設計レビュー ・べからず集の作成
28	発注時の丸投げによって失敗	業務知識・構築実績があると思われた協力会社に開発を委託したが、進捗・品質に問題が発生した	協力会社のマネジメント力の不足による。また、協力会社を信頼して全面委託する場合、協力会社からの見積もりの検証や品質チェックが不足していて問題になることもありうる。業務システムの特性によっては、見積もり規模も予想より大きくなることもあり、見積もりチェックなどはきちんと実施すべきである	・協力会社の開発体制を強化する ・顧客からの要件がまだ増える場合には、顧客との調整でスコープを整理する ・協力会社に対するプロジェクト・マネージャ支援、作業支援を通して、プロジェクトの一体感を醸成する	・協力会社の体制強化 ・プロジェクト・マネージャ支援要員の追加 ・作業分担の見直し	・協力会社の事前の調査・評価 ・協力会社のプロジェクト管理の徹底 ・協力会社とのコミュニケーション計画の合意(進捗報告の義務など)

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
29	パッケージのバグフィックスに時間がかかる	パッケージの機能のうち、初めて利用した機能の品質が悪く、そのバグ修正に時間を要した	パッケージの機能のうち、初めて利用した機能の品質が悪かった。海外のパッケージ・ベンダーだったこともあり、バグ対応に時間がかかった。また、バグ修正により機能がデグレードしていることもあり、受け入れテストを十分に実施する必要があった	・パッケージの中で初めて利用する機能は、その品質を十分確認しておく ・バグがあることを前提としたテスト・スケジュールを立てる	・迂回策の検討 ・代替製品の調査 ・アドオンによるカスタマイズ対応	・テストスケジュールの作成 ・パッケージのサポート体制の確認 ・サポートに対する覚え書きをとる ・事前調査(初利用パッケージの機能・性能) ・リスクの顧客との共有
30	大量データ処理方式の設計ミス	総合テストに向け、テスト環境とテスト計画を策定したが、アプリケーションの開発リーダーから性能問題に関する懸念があがった	現行システムの業務集中と処理能力の改善報告を受け、アプリケーションの開発リーダーから新システムの処理方式に問題提起があった。方式開発リーダーは業務量に応じた性能設計を行い事前にテスト環境で確認していたが、業務運用に伴うトランザクションの傾向には疎く、想定外であった	・処理方式を見直す ・業務処理の集中を避けるための運用制限をかけるべく、本番環境を用いて性能テストを実施し、顧客の承認を得た	・類似個所の見直し ・運用制限 ・処理方式の見直し	・事前調査(運用特性)
31	あいまいな仕様のまま実装し、テスト工程で要件不一致が多発	開発委託先からの納品を受けて結合テストを開始したが、要求機能を満たしていなかった	要件(業務ルール)のドキュメント化が不十分で、プロジェクト・マネージャ(元請け会社)の頭にしかなかった。開発委託先へのレビュー不足	・テスト手順書をディジョン・テーブル形式で記載 ・そのテスト手順書に基づいて、開発委託先に再度プログラミングと単体テストを依頼 ・1次リリースと2次リリースに分割	・業務ルールのドキュメント化 ・テスト手順のドキュメント化 ・再テストの実施 ・コードレビュー ・再スケジュール	・仕様のドキュメント化
32	外字印字の不良	文書を印字するシステムで、本番稼働後に、文字が抜けて空欄のある帳票が印刷されていた	学術系の文字が外字として登録されていた。開発したシステムからこれらの外字を印刷するためには、運用環境でも印刷用に外字の設定が必要だった。しかし、外字登録の範囲や必要性を開発側が把握しておらず、外字のコードをそのままプリンタに送ったため、問題が発生した	外字設定を実施後、空欄が発生していないか印刷結果に対して全件目視確認を実施した	・全文字確認テストの実施 ・顧客の要求仕様の明確化	・事前調査(現行調査)
33	運用中の業務停止	運用開始後にシステムが異常終了してしまった	テスト環境と本番環境が異なるため、本番環境への移行時にパラメータを修正する必要があった。本番環境での動作確認はできないため、パラメータ・ファイルの差分を抽出して確認した。ただし、「正しい差分が出ていること」はチェックしたが、「差分が出ないということは反映もれ」というチェックを怠ったため、テスト環境のパラメータが残ってしまった	全ソースコードのパラメータ・リストを出力し、本番環境とテスト環境の差異を把握している有識者2人で目視確認した	・設定状況の見直し ・有識者の参画 ・構成管理者を立てる	・有識者によるレビュー ・テスト計画のレビュー ・リリース手順の見直し
34	指揮・命令系統が不明確	プロジェクト内の指揮・命令系統にあいまいな部分があり、プロジェクト・マネージャによるリソース調整がうまくできなかった	このプロジェクトは、専任のAチームと他のプロジェクトを掛け持つBチームで構成していた。Bチームに対するプロジェクト・マネージャの権限があいまいであったため、プロジェクト・マネージャはBチームに対するリソース調整ができなかった	プロジェクト推進における責任分担、指揮・命令系統の明確化	・体制俯瞰図の作成と見直し ・作業分担の明確化 ・責任の明確化 ・チーム間の作業の調整(優先順位の判断、プロジェクト・マネージャやその上司との意識合わせ)	・プロジェクト計画時から責任分担と作業分担を明確にする ・作業負荷の明確化 ・プロジェクトのスコープを明確化
35	プロジェクト編成の中でプロジェクト・マネージャ的な人間が2人存在する形になり混乱	営業活動から参画していた全体的視野を持つプロジェクト・マネージャのもとに、開発部隊を率いた実力派の開発リーダーが参加。その実力ゆえにプロジェクト・マネージャが二重に存在するような状態となった	社内で、開発の統制に関する仕切りが不十分だった	開発チーム立ち上げ時の社内統制に細心の注意を払う。上級管理職、幹部の監視と振る舞いが重要	・責任分担を明確にする	・プロジェクト計画時から責任分担と作業分担を明確にする
36	他業種から転進した元請け会社が未熟で混乱	他業種から転進した元請け会社のもとで2次請けするが、元請けの作業標準が確立しておらず、元請けを教育しながらの開発となった	他業種からの転進の場合、元請け会社としての技量は容易に獲得できず、未熟なスキルのまま元請け業務を推進していた	・元請け会社、2次請け会社の機能分担を明確にし、長期的な企業連携による技量向上を図る ・2次請けは元請けにソフト作業標準を教え、基本的な手順について合意を形成する	・作業分担の明確化 ・開発標準の共有化	・開発標準に関する合意

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
37	DBMSの技術的な問題で対応できない	DBMSに不慣れな開発要員だけでアプリケーションを開発。DB操作のレスポンスが遅すぎて、システムとして使い物にならなかった	ITアーキテクト要員の不足。本来なら利用技術が確定したところで必要な技術者をアサインし、技術グループを構成するところだが、それをしないで進めた場合に陥りやすい問題	・時期的に遅くなっても専門の技術者を連れてこないと解決しない ・技術者にまかせきりにしないで、アプリケーション開発要員の数人を同じチームにして技術グループを構成する	・技術者の確保 ・開発要員への教育 ・再設計 ・再構築	・有識者の参画 ・要員のスキルレビュー ・要員教育
38	結合テストで品質不良が発生	Webアプリケーション・サーバー上の結合テストに着手したが、プログラム設計の誤りによるバグが多発した	詳細設計工程で業務仕様が増加したため、プログラム設計工程から要員を多量に追加した。これらの要員は、対象業務も、そのプロジェクトの開発環境も初めての経験だったうえ、進捗の遅れも重なり、オブジェクト指向によるプログラム設計や新処理方式に関する事前研修が不十分だった。研修不足を補うために有識者のレビューを実施することもなく、途中参加の要員にプログラム設計を任せていた	・結合テストを中止し、対象業務の経験者を投入 ・レビューを通じて発見した誤りを全員に周知して、各自がプログラム設計を見直し、全プログラムを改修した	・再スケジュール ・経験者の投入 ・ソースコード・レビュー ・追加要員への教育	・研修の実施 ・コーディング規約の策定
39	受注後の要件詳細化、基本設計体制に不備	メンバーの経験が設計後の開発ばかりだったため、要件定義や設計能力に欠けていた。設計工程から遅延し始めた	設計後の開発ばかり受注していたため、要件定義、設計能力が育たなかった。設計待ちの姿勢となった	短期には、必要なスキルを備えた要員を探し、投入する	・経験者の投入	・設計者の育成 ・上流設計技術者の採用
40	大幅な工数増加と工程遅延	人的リソースを十分に確保できていない状態だったが、要員不足・工期不足にも「なんとかなる」といった甘い判断でプロジェクトを進めていた	プロジェクトに必要なスキルを持った人的リソースが、他のプロジェクトに取られてしまった。その補充要員はスキル不足で、従事可能期間もかみ合わず、結果的に数少ないキーパーソンに負荷が集中。機能横断的なコーディネート、設計不備などのチェック、要件拡大に対する適切な対応がタイムリにできなかった	・サブプロジェクト・マネージャの投入 ・管理要員の投入 ・開発要員の投入 ・残タスク（機能開発、障害対応、仕様変更対応など）の洗い出しと、そのタスクの実施スケジュールの作成、担当者のアサイン、顧客との調整を実施 ・日次での進捗管理	・組織マネージャへのエスカレーション ・人員投入 ・残タスクの洗い出し ・再スケジュール ・日次での進捗管理 ・顧客調整（プロジェクト間の全体最適）	・リソースの十分な確保 ・リスク管理 ・プロジェクト・マネジメントの徹底
41	業務担当リーダーの離脱	業務には詳しいがマネジメントでは不安があった業務担当のリーダーが、総合テストの直前にプロジェクトを離脱	総合テストを控え、業務に一番詳しい人を業務担当リーダーとしてマネジメント・チームに加えるよう指示があった。業務の有識者はいたが、協力会社の人でもあり、マネジメントには不安があった。プロジェクト・マネージャは別の人をアサインするよう提案したが、方針ということで断られた	・マネジメントに長けた経験者をチームに投入 ・業務の有識者がいなくなったので、残ったメンバーとの面談を通じ、業務仕様を任せられる人を選定 ・顧客に協力を依頼し、業務仕様のレビューを行いながら人材育成する	・経験者の投入 ・業務有識者の投入 ・業務担当リーダーの育成 ・顧客への協力依頼	・マネジメント能力のあるプロジェクト・マネージャの任命
42	協力会社のリーダー、中核要員がくるくる変わる	プロジェクトの立ち上げ時に協力会社の要員が交代する	好況時、人材が出払っているにもかかわらず、協力会社が営業的に受注を優先し、場つなぎの人材を当てて、急場をしのいでいた。本格的な体制が組めるようになったのは、プロジェクトがかなり進んだ段階だった	・人材派遣型契約の抑制 ・ドキュメント主義型の開発スタイルをとる	・本格体制の構築 ・再スケジュール	・仕事に応じた体制の構築
43	協力会社のスキルが低すぎてまったく役にたたない	協力会社を使って設計をさせたが、まともに設計ができない。内容のレベルが低い	協力会社の要員のスキル不足、経験不足	プロジェクトの状況によるが、本来は協力会社を替える必要がある。ただし、ポテンシャルがあれば、教育することで良い協力会社に変わることができる。協力会社に限らず、スキルを持っている要員を集められればよいが、実際はスキルのない要員も多いので、結局は教育をした方が得策なこともある	・協力会社要員への教育実施 ・設計レビューの実施 ・業務有識者の投入	・協力会社の能力の調査 ・協力会社へのガイドラインの作成

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
44	プロジェクト・マネージャ、リーダーと実務担当者の関係が悪化してコミュニケーションも作業も進まない	次期システムの構築に当たり、実務担当者は旧システムの保守から携わっているため業務知識が豊富である。一方、プロジェクト・マネージャとリーダーは別のプロジェクトから引き抜かれた要員である。やり方の違いなどで意見が合わず、コミュニケーションも悪化している	やり方、進め方、コミュニケーションのとり方のミスマッチがあった。問題の発見が遅れた	・この問題はプロジェクト責任者でないと対応できない ・プロジェクト・マネージャか担当者かどちらに問題があるのかを明確にして問題のある方を是正する	・関係者からの状況ヒアリング（個別面談） ・議論の場を作る（合宿の実施など） ・実務担当者からの説明会実施 ・お互いの仕事の状況を情報共有 ・課題の情報共有	・作業方針を示す ・会議体の設定 ・キックオフの実施（プロジェクト・マネージャが変わったらキックオフをし直す） ・プロジェクト・マネージャのメンバリングとチームビルディングの違いを認識する
45	役割分担が不明確	アプリケーション開発担当チームと運用担当チームのコミュニケーションが不足して、手戻りが発生。作業負荷が大幅に増えて、メンバーが疲弊した	アプリケーション開発担当チームと運用担当チームが連携してプロジェクトを進めるのは初めてで、双方の役割分担が不明確だった。両者間の情報伝達も悪く、特にアプリケーション開発担当チームから運用担当チームへの情報伝達の遅れや度重なる仕様変更により、運用設計の作業負荷が増大した	・それぞれの分担と責任範囲、連携方法を明確にする ・各チームの定例ミーティングに双方のリーダー、キーパーソンが出席して情報交換する	・メンバーを集めてミーティングをする ・チームビルディング ・ルール作り	・コミュニケーション・ルールの策定 ・作業計画の情報共有
46	複雑な開発組織内で、口頭による依頼、周知が多用され混乱	開発組織が複雑で、作業標準が統一されておらず、口頭による連絡が多用されて大混乱となった	複雑な組織構成のなかで、コミュニケーションの文書化がなく、コミュニケーションの管理が不在となった	・必ず文書でコミュニケーション（依頼、回答、周知など）をとる ・この文書（1件1ページ）に通番を振り、通番管理をする ・これら文書の管理者を置く。関係者の進捗管理会議で、この文書管理を実施する	・問題点一覧の作成（情報集約） ・コミュニケーション・ルールの策定 ・メーリングリストの作成 ・伝達事項の文書化	・コミュニケーションルールの策定 ・問題点管理ルールの確立
47	事務方との連携が悪く、開発グループが客先で孤立	2次請けしたプロジェクトで、元請け会社内に詰めている開発グループの労働環境が悪化し、疲弊した	2次請け会社の事務方が開発現場の状況を把握できず、労務的に放置状態となっていた	・事務方を開発現場に連れてゆき、状況を把握させ、会社として対応可能な施策をすべて打たせた ・ホテルの手配、タクシー券の配布、補食の手配 ・事務的サポート要員の手配、通信環境の改善、予算措置など ・随時、現場の状況を把握する体制を確立した	・作業環境の改善 ・PMOの派遣 ・事務支援要員の投入	・事前の作業環境計画
48	総合テストが品質不良で進まない	結合テストまでは先行していたチームにおいて、総合テストで品質不良が判明した	結合テストまでの進捗が一番早かったチームから総合テストに入ることになった。そのチームの業務担当リーダーは詳細設計以降の仕様変更・追加を把握しており、テスト範囲を限定しながら総合テストを進め、並行して仕様変更･追加することを考えていた。しかし、その考えが顧客に伝わっておらず、総合テストは顧客の都合に合わせてスケジュールされていた。仕様変更・追加への未対応部分がすべて品質不良となった	・総合テストの開始を遅らせ、仕様変更・追加の開発を先に実施した ・業務担当リーダーは総合テストの実施に当たり別の作業場所に移ったので、新たに業務知識のあるサブリーダーを選び、仕様を整理させた ・サブリーダーは定期的にマネージャとコミュニケーションをとり、業務仕様の確認作業を進めた	・再スケジュール（テスト開始時期） ・業務担当リーダーの育成	・顧客との進捗状況の共有 ・テスト計画の顧客との合意
49	仕様書と実装の差異に顧客がクレーム	基本設計書に基づいて作成した画面を顧客に確認してもらったところ、画面内の微妙な差異について指摘を受け、ドット単位で修正することになった	画面仕様書はプロトタイプを基に作成したが、顧客はそれが完成イメージであるとして認識していた。実装後にある程度の差異が発生することを事前に説明していなかったため、顧客との調整に失敗した	仕様書に完全一致するよう、全画面を再修正した	・要求にあわせて修正実施	・厳密性を要求される画面・帳票イメージの顧客確認

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
50	顧客のプロジェクトマネージャが交代してしまった	設計が完了し、これから開発に入るという段階で、顧客側のプロジェクト・マネージャが突然会社を辞めてしまい、プロジェクトの運営に支障を来たした	事情はどうあれ、顧客側のプロジェクト・マネージャが交代することはあり得ることである。そのときの対策を速やかに打たないとプロジェクトの運営に支障を来たす	・プロジェクト・マネージャが社内でどの程度状況や進捗を報告しているかを確認する ・新しいプロジェクト・マネージャを配属してもらうよう依頼する ・新しいプロジェクト・マネージャが配属になったら、すぐに集中ミーティングを実施して、プロジェクトの概況、進捗状況、予算と実績、体制、リスクなどを共有する ・新しいプロジェクト・マネージャは最初に情報を提供してくれる者を信用する傾向があるので、情報を提供する側は早さと正確さを心がける	・新しいプロジェクト・マネージャの指名の依頼 ・プロジェクト・マネージャの方針を新プロジェクト・マネージャに伝える ・キックオフの実施 ・議論の場を作る(合宿の実施など) ・実務担当者からの説明会実施 ・お互いの仕事の状況を情報共有 ・課題の情報共有	・リスク管理 (情報の文書化、顧客側のサブリーダの確保) ・契約書にプロジェクトマネージャの交代に関する制約事項を記載する ・キーパーソン・プロジェクトマネージャの交代時には事前に連絡をもらうようにしておく
51	協力会社が設計を完了した時点で次工程の開発を断ってきた	基本設計を委託していた協力会社が設計完了間近になって、「基本設計までは請け負うが、次の工程からは撤退したい」と通告してきた。意思はかなり固い	協力会社には、プロジェクト・マネジメントへの不安、設計内容を実現することへの不安、次工程で要員増加できない不安などを抱えていた	・まずは引き止め交渉をする。それが無理となったら、次の協力会社への引き継ぎを確約させる ・その上で、コストも含めた引き継ぎ策を協力会社と決める(通常、引き継ぎにかかるコストは支払わない約束を取り付ける)	・引き留め工作 ・引き継ぎ作業の確約と契約	・マネジメントの強化 ・協力会社との課題共有 ・協力会社の能力に応じた範囲での発注
52	業務有識者不在での総合テスト実施	本番環境で総合テストを実施する前に担当者が交代した。急いで別の担当者をアサインしたが、顧客との調整やテスト・シナリオの作成に想定以上のコストを費やすこととなった	総合テスト開始前に業務有識者が離職し、新しい担当者が業務知識や顧客対応の面で先任者のレベルに追いついていなかった	プロジェクト・マネージャと関連業務を担当している有識者を招集して臨時に対応した。1人で実施する予定だった作業に人数をかけたので、コストは人数分増加したが、顧客に対する影響はほぼ解消できた	・次のキーパーソンを育てる ・再スケジュール ・ドキュメントがあるかのチェック ・有識者を連れてくる	・キーパーソンを複数作っておく ・ドキュメント化する(仕様書だけでなく、判断基準、業務ノウハウなどを明記する) ・受託時によく検討する(業務に詳しい人が複数人いること)
53	契約前に作業完了	短期の小規模システム開発において、契約未締結のまま開発を開始し、工期完了まで実施してしまった	開発計画と見積もりに対し、顧客から承認の連絡を得て開発作業を開始した。月内に内示をもらったが、正式発注は工程期間内に実施されなかった	顧客側担当者の上長を含めて交渉の機会を設けてもらい、正式発注手続きを早急に実施しなければ、システムを納品できない旨を伝えた。翌月、正式発注された	・正式発注手続きを強く求める	・発注をもらってから開発に着手
54	ベンダ提供の業務パッケージの品質不良による運用の中止	パッケージ・ソフトをカスタマイズしてシステムを稼働させたが、品質と性能の問題が生じ、稼働後すぐに運用を中止することになった	パッケージの仕様として提供されるものと、カスタマイズして開発するものとの判別を、パッケージ・ベンダー任せにしていた。これが原因でカスタマイズの開発期間を十分に取れないまま、納期を向かえてしまった。また、プロジェクト・マネージャが顧客やパッケージ・ベンダーとの仕様調整に悩んで離脱。交代したプロジェクト・マネージャは兼務だったため、プロジェクトの状況や問題を十分に把握できないまま、パッケージを導入して稼働させてしまった	・パッケージ・ベンダーや業務有識者による状況判断の結果、品質・性能を改善するために稼働を半年遅らせることにした ・パッケージ・ベンダーとの開発分担を見直し、開発体制を立て直した	・現状調査の実施 ・再スケジュール ・顧客交渉 ・役割分担と開発体制の見直し	・事前調査 ・負荷テスト ・パッケージベンダとの責任分担の明確化 ・第三者による進捗状況管理
55	営業時に営業SEが決めたブラウザが実用に耐えなかった	受注活動の中で営業SEがブラウザの製品を決めて提案し、そのままシステム構成に組み込まれたが、テスト段階で実用に耐えないことが判明した	営業SEは、製品の実用性を確認する立場になく、確認する組織が社内に存在しなかった。システム開発部門は、受注後、与えられた条件でシステム構築するだけだった	・受注後、開発部門でシステム・アーキテクチャを再検討し、実現性の面から再評価、再設計する ・営業活動から引き継いだ条件を、無条件に引き継がない ・そのためのコストを見積もりに盛り込む	・利用可能な代替製品の調達	・事前調査
56	営業時に営業SEが決めたRDB製品の最新版を自社ミドルウェア製品がサポートしていなかった	受注活動の中で、営業SEがRDB製品の最新版の利用を決めて提案し、受注後、そのままシステム設計が進んだ。しかし、これと連動する自社ミドルウェア製品がRDBの最新版をサポートしておらず、その対応に時間と費用がかかることが判明した	営業SEは自社ミドルウェア製品の開発計画を正確に把握していなかった。RDBの最新版のサポートに要する費用、期間に関する正確な認識を持っていなかった	・営業SEが自社製品の開発に関する正確な情報を保持できるような社内体制を確立する ・営業活動のシステム提案を社内で評価できる体制、そのために必要なコストをプールする体制を作る	・調達品に関する顧客との調整	・事前調査(パッケージ、社内製品の整合性など) ・営業社員への自社パッケージの教育 ・提案内容のシステム担当者によるレビュー

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
57	パッケージ・ソフトの利用によりブラックボックス化が問題に	過去に実績のあるパッケージ・ソフトを導入したが、その品質が著しく悪かった。パッケージ提供会社は「問題ない」と回答してくるが、実際の状況とかけ離れた内容だった	パッケージ・ソフトは、内部に踏み込んだテストで品質を確認できないため、パッケージ提供会社からの回答をうのみにしている場合に問題になりやすい	パッケージ提供会社に品質管理体制やサポート体制の強化を依頼する。テスト支援などを依頼することもある	・パッケージベンダを巻き込む ・アドオンによる回避 ・代替製品の調達	・パッケージ会社との保証契約
58	結合テストで重複障害やデグレートが多発	結合テストに入り、バグを発見して修復しても、再度同じバグが発生したり、インタフェース・ミスが多発したりする	バグ修復を複数のプログラムに対して実施した場合に、修正履歴、プログラムの世代管理が厳密に行われていなかった。未修正のプログラムを使ったり、担当者が勝手にオブジェクトだけを入れ替えたりしたため、結合テストの業務プログラムがどのような構成になっているか把握できていなかった	結合テスト環境を運営する専任者を配置し、履歴、世代、構成管理ルールを改善。責任者の承認を得てテスト環境を提供するように改善した	・構成管理ルールの策定と実行 ・構成管理者の配置	・構成管理ルールの策定 ・構成管理者の配置 ・構成管理ツールの決定
59	自社のプロジェクト・マネージャが体調不良で倒れた	自社のプロジェクト・マネージャが体調不良で倒れてしまった	長時間残業、責任の集中、精神的疲労などプロジェクト・マネージャは常のストレスとの戦いにあり、それが原因で体調不良に陥った	プロジェクト・マネージャの交代はプロジェクト全体に不信感を広げるので早期の対応が必要である。リーダーの中から暫定プロジェクト・マネージャを任命する方法もあるが、失敗するとこのプロジェクト・マネージャも離脱して将棋倒しのような現象が起こり得る。プロジェクト責任者が暫定的にプロジェクト・マネージャを兼任するのがよい	・新しいプロジェクト・マネージャの投入(上位者、内部)	・上位者によるプロジェクト・マネージャへのケア ・会社の勤労管理の整備
60	顧客側の担当者が能力・経験不足でまったく役にたたない	顧客側の担当者とまともなコミュニケーションをとれず、何を言っても十分に対応してもらえない。プロジェクトも停滞状態に陥った	顧客側の担当者のコミュニケーション能力がなかった。システムも業務も理解していなかった	システムを開発する上で重要な情報を顧客側の担当者から得られない場合、それに起因するリスクを一覧にまとめ、顧客側の責任者と相談する。プロジェクトとしてのリスクを客観的かつ誠実に説明して改善を求める	・リスク一覧の作成 ・リスクを顧客に説明する ・第三者による顧客窓口の変更依頼	・コミュニケーション・ルールの計画
61	テスト工数を小さく見積もってしまった	パッケージ・ソフトを導入するので、実際の開発工数は少ないと思い、機能工数の積み上げで見積もってしまった	パッケージ・ソフトの導入は、開発工数を抑えられるが、妥当性検証やインタフェース・テストなどのテスト工数がかかる。それを見積もっていなかった	・有識者を加え、不足するテストを洗い出し、必要な期間、要員、テスト環境などのリソース投入を見積もる ・実現可能なテスト計画を再作成し、ステークホルダーに変更承認を得る ・プロジェクトの目標を達成でない場合には、品質、コスト、納期の見直しも含め、ステークホルダー間で調整する	・残タスクの洗い出し ・再スケジュール ・顧客との費用の交渉 ・要員投入	・見積りの妥当性検証 ・パッケージの調査
62	開発チームへの指示系統が複数	開発チーム(2次請け会社)への指示系統が2つになっており、現場が混乱した。作業遅れと作業品質の低下を引き起こしていた	元請け会社が2次請け会社に開発を委託して、さらにその下に孫受けの開発ベンダーがいた。元請け会社が孫請けの開発ベンダーに直接指示を出していたため混乱が生じた。元請け会社の指示は、現場レベルの作業手順に落とし込まれていなかった	元請け、2次請け、孫請けの3社による確認の場を設け、「階層を飛び越した指示は出さない、受けない」を改めて合意事項として確認する	・指示系統の明確化と徹底	・コミュニケーション・ルールの計画 ・ステークホルダー俯瞰図の作成と合意
63	個別チームの状況を把握していないため、総合テストに入っていいのかわからない。	結合テストから総合テストに進もうとする段階で、「総合テストに入ってよいものかどうか判断できない」と悩んだ	プロジェクト全体の統括機能が弱く、開発重視の体制になっていたため、キーパーソンがサブシステムの開発リーダーに回されていた。これにより、総合テストに進むべきか判断できる人材がテスト・チームにいなかった	・チームごとに「品質評価報告書」(結合テスト結果の総括)を提出させる。これにより、そのチーム(機能、サブシステム)が総合テストにいつ入れるか判断する ・その結果を基に、必要があれば結合テストの組立て(テスト順序など)を見直す。本来のテスト計画通りに結合テストができない可能性があるため。 ・総合テストに進むチームのリーダーに全体統括の機能を担当してもらう	・品質状況の把握 ・総合テスト計画の策定 ・総合テストのための体制作り	・工程区切りの評価 ・品質評価基準の策定 ・品質報告の徹底とレビュー ・総合テストの品質管理ルールの策定 ・体制計画

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
64	キーパーソンが疲弊している	危機的状況にあるプロジェクトで、プロジェクト推進のために作業がキーパーソンに集中した。キーパーソンは疲弊しきってしまい、作業効率が上がらず、さらにプロジェクトが進まなくなってしまった。それが原因で、さらにキーパーソンに負荷が増えるという悪循環が発生した	どうしても負荷がキーパーソンに集中してしまう一方で、キーパーソン以外の人材が育っていなかった。キーパーソンが倒れたらプロジェクトは破綻してしまう	・キーパーソンのタスクを整理する要員を付けるとともに、キーパーソンの「キー」である部分を代行できる人員を育成する ・タスク整理要員の主な任務は、集中するタスクの交通整理と重要度/期限の管理。キーパーソンは与えられたタスクを順にこなすだけの作業に専念できる ・キーパーソン代行要員の主な任務は、対象となる「キー」部分を習得し、少なくとも問題の1次切り分けと解決に必要な情報を収集できること。キーパーソンはある程度整理された情報の中から答えを出していく作業に集中できる	・キーパーソンの負荷分散	・キーパーソンに対する労務管理 ・キーパーソンに負荷がかからないような作業分担 ・後継者の育成
65	協力会社が多階層になり責任範囲があいまいに	規模が非常に大きいプロジェクトで、協力会社から納品されるプログラムやシステムの品質が悪かった。協力会社間の責任範囲があいまいで、解決のための方向性が見えない	大規模な案件で起こりやすい。顧客から発注した元請け会社の下に、2次請け会社、孫請け会社と多階層になり、各社の責任範囲が不明確になってしまった。これに起因して、品質の低下、状況把握の悪さが表面化。プロジェクトの規模に対して、統合管理チームが手薄だった	・中間層の協力会社が機能を果たしていないことが多いので、その場合は例外的に、会社・階層を問わない指示体制を確立する ・協力会社が一つの場所に集まっていれば、上記の対策を進めやすい ・中間の協力会社が孫請けの開発状況を把握できないのなら、中間の協力会社から実質的に指揮権を取り上げる(責任は持ってもらうが、管理できていない点を主張する) ・協力会社が場所的に離れている場合は、上位の協力会社とのコミュニケーションをよくする。最後は緊急集合	・作業者を集める ・指示・命令系統の整理 ・統合管理チームの構築	・協力会社(中間層)に対する管理徹底
66	実質的なマネジメントを協力会社の開発グループが実施	元請け会社が2次請け会社に発注しているが、2次請けの持つ製品がシステムの主要部分を構成しているために、実質的なプロジェクト・マネジメントを2次請け会社が担当した。その負荷が重く、肝心の開発作業が遅れてしまった	2次請け会社がプロジェクト・マネジメントと開発作業を一緒にやることで、オーバーフローしてしまった	契約時の作業分担を明確にする必要がある。特に、メーカーは2次請けの協力会社にシステム開発を丸投げするケースが増えてきおり、メーカーは「プロジェクト・マネジメント担当だが何もしない」、協力会社は「すべてを請け負い、開発の責任もすべて持つ」、顧客は「困ったら問題を協力会社に押し付ける」、という関係はシステム開発を台なしにしてしまう	・作業分担の明確化 ・作業の再配分	・作業分担表の作成 ・会議体の設定
67	総合テストの進捗が悪い	上流工程の進捗はそこそこ良かったが、下流工程で総合テストが遅れだした	総合テストをあまりスキルの高くないエンジニアが実施していた	・要員の交替 ・テスト作業の均質化を図るため、テスト・ツールや標準化されたテスト手法を採用 ・テスト仕様書を見直す。スキルの低い要員でも仕様書通りに作業すればよいという環境を作る	・テスト要員投入 ・ルールの策定 ・ツールの標準化 ・教育 ・スキルに応じた役割・作業分担	・テスト計画の立案 ・総合テストには顧客に参画を依頼する ・ハイスキルな要員を総合テストにあてる

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
68	テスト作業に無駄が多い	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	・システム設定情報が誤っていた ・テスト用データベース、テスト・データなどのテスト環境の品質不良 ・テスト手順や障害発生時の情報取得・解析手順、復旧手順、構成管理といったテスト作業の品質不良	・テスト環境数を増やす。テストごとに設定情報の変更やデータベース/テスト・データの入れ替えをしなくて済むようにする。あるいは、その頻度を少なくすることにより、テスト作業の品質不良による影響を少なくする ・複数のテスト環境の統括管理責任者を置き、次のことを管理させる −テスト環境の利用計画作成/スケジュール管理 −テスト環境の切り替え計画作成/スケジュール管理 −設定情報を含めたテスト環境の構成管理 −テスト環境切り替え手順書作成と周知徹底 ・障害管理責任者を置き、次のことを管理させる。 −障害対応フローおよび手順書の作成と周知徹底（障害登録、チェックアウト、チェックイン、プログラム修正・テスト、テスト環境、プログラム・リリース、報告までの一連のフローを含む。また、テスト実施者、プログラム修正者、ライブラリ管理者、障害管理者の誰がやるかを明確にする） −修正プログラムのリリース・スケジュール作成と実績のフォロー ・テスト実施時の作業フロー、手順書の作成と周知徹底 ・上記を実施することによる工程遅延と全体工程内での挽回策を顧客に説明し、承認を得る	・テスト環境の整備 ・テスト環境の統括責任者配置 ・障害管理責任者の配置 ・テストの作業フロー・手順書の作成と徹底 ・顧客への挽回策の説明	・テスト計画の立案と有識者レビュー（テストのプロ、業務有識者、顧客）
69	サービス開始の可否が判断できない	進捗が大幅に遅れており、サービス開始の可否が判断できない	本社の情報システム部門が、支社の情報システム部門に仕様策定を丸投げしていたために、本社の情報システム部門が仕様や進捗状況を把握していなかった	・プロジェクト・マネージャ、キーパーソンのヒアリング、プロジェクト進捗状況の実情把握、問題摘出 ・サービス開始時期の到達予測 ・プロジェクトの再構築、再開による到達度を予測 ・上記とほぼ平行して、顧客の最低必要サービス項目の抽出、スケジュール再設定 ・サービス項目、サービス開始時期を再設定し、顧客、プロジェクト・マネージャ、メンバーによる打ち合わせ体制を見直す	・現状把握 ・プロジェクト・マネージャ支援要員の投入 ・再スケジュール ・体制見直し ・リリース機能の顧客との合意	・プロジェクト計画の策定 ・プロジェクト監査の仕組みづくり
70	総合線表がない	個々の開発チームの線表はあるが、全体を俯瞰する線表がない。そのため、システム全体としての進捗はどうなっているのか把握できない	プロジェクト・マネジメント能力の不足	・プロジェクト・マネージャの交代あるいはプロジェクト・マネージャ支援者の配置 ・システム統合チームの設置 ・顧客のサービス必要時期の再確認 ・システム統合への移行スケジュール、体制、作業項目（WBS）の策定とその実施	・プロジェクト・マネージャ支援要員の投入 ・プロジェクト・マネージャの交代 ・顧客との必要機能の調整 ・再スケジュール ・体制の見直し	・プロジェクト計画の策定 ・プロジェクト監査の仕組みづくり ・工程区切り監査
71	顧客側と開発側のプロジェクトの一体感欠如	顧客側と開発側の間にプロジェクトとしての一体感がない	開発側のプロジェクト体制が手薄	・プロジェクト推進会議を設置する。顧客側、開発側双方のキーパーソンと有識者による会議体を設定し、重要課題の検討など、プロジェクトの方向付けや助言を行う ・開発側体制を見直す。業務知識を持ったメンバーの投入やPMOメンバーによるマネジメント支援を要請する ・段階的に機能をリリースするなど、スケジュールを見直す。スケジュールの見直しは顧客側キーパーソンを含めて検討し、同意を得るとともに一体となって対応する意識を高める ・進捗会議には顧客側キーパーソンの参加を求め、情報共有を促進する	・会議体の策定による一体感の醸成 ・プロジェクト・マネージャ支援要員の投入	・チームビルディング ・コミュニケーション・ルールの計画 ・非公式なコミュニケーション手段の形成

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策（教訓）
72	オープンシステムのプロジェクト経験がない	オープン系システムの構築経験がなかったため、システム品質の低下や、進捗遅延を招いた	オープン系システムの構築、特にマルチベンダー/マルチプロダクトによるオープン系分散処理システムには、多数の製品ベンダーが絡むため、高度な技術力とマネジメント力が要求される。しかし、統括部門に経験がなく、要員もいなかったために、取りまとめの能力がなかった	・至急、オープン系システムの経験者、専門家をプロジェクトに動員できるようにする ・オープン系製品群を統合利用したことがある経験者の助言を受けられるようにする ・自社内に人材が不足していれば、社外に人材を求める ・使用する個別製品の経験者を確保する。製品ベンダー（開発元、販売元）の顧客サポート部門との関係を構築する。あるいは、開発チームをオープンシステムの経験者に変更する	・有識者の投入 ・製品ベンダーとのサポート体制を構築	・リスク管理の強化 ・チームビルディング
73	要求仕様が固まらないまま開発に突入	要求仕様が固まらず、要求仕様書がないまま開発が進み、プロジェクトが破綻した	プロジェクト・マネジメント能力の不足。最近の小規模プロジェクトでは、このように要求仕様書が確定しないまま開発に入るのは当たり前になっている。後付けで要求仕様書を作成するケースが多い	・本来、要求仕様書を作らずに開発を進めるべきではない。開発を一度止めて、改めて要件定義を確定させるタスクを設ける。その時点までの設計書を活用して、できるだけ期間短縮を図る ・プロトタイピングを実施して主要機能の画面、帳票などを簡易に作成し、プロトタイプで実質的に要求仕様を確認する ・開発規模が小さい場合（7人以内ぐらい）は、XP（Extreme Programming）手法に変更して対応する	・開発を一度止める ・要求仕様の確定とドキュメント化 ・プロトタイプの作成	・開発プロセスの構築（開発計画書に記載） ・顧客に開発プロセスを合意してもらう
74	プロジェクト全体に疲弊感が漂っている	プロジェクト全体に疲弊感が漂っており、現場の様子を見ていてもモチベーションの低下が見られる	疲弊感やモチベーション低下の原因はさまざまだが、出口の見えないプロジェクトや、先の見えないテストに明け暮れる日々がモチベーションを低下させる	・チームメンバーとのコミュニケーションを通じて疲弊感やモチベーション低下の原因を探る ・プロジェクトの目標と現状を再確認し、目的達成に向け、作業の優先順位付けや最適化を行い、実行可能な計画に見直す ・再度、クリティカルパスやマイルストーンをメンバーで共通認識する（ゴールを再確認する）	・プロジェクトの目標の再確認 ・チームメンバーとのコミュニケーション ・作業の優先順位付け ・計画の見直し ・ゴールの設定	・労務管理の徹底 ・チームビルディングの実行
75	外部調達のパッケージ製品の品質が悪くて使えない	外部から調達したパッケージ製品の品質が悪い	外部からのパッケージ製品の調達に関しては、そのパッケージがどのようなサービス・レベルをターゲットに作られているのかを事前に検証しておく必要がある。この事例では、ミッションクリティカル性を想定していないパッケージ製品だったことにより問題が生じた	・パッケージ製品利用時には、パッケージ製品をサポートする会社と必ずサポート契約を結び、逃げられない体制を作る 海外の会社に対しては、製品使用の制限事項や用途の制限について事前に調査する ・パッケージを使うには、パッケージに合うよう顧客のビジネス・プロセスを変更する必要があることを十分顧客に認識してもらう	・サポート契約の締結 ・パッケージの調査 ・顧客との情報共有	・事前調査（サービスレベル） ・パッケージ・ベンダーとの契約事項の確認

項番	事象タイトル	事象	原因	対策	対処療法	再発防止策(教訓)
76	プログラムの品質が悪い	プログラム品質が悪く、テストを実施してもバグだらけだった	仕様に問題がある場合もあるが、たいていはプログラミング能力の不足やレビューの不足が原因	・プログラマごと少なくとも1本のソースコード・レビューをして、スキル不足のプログラマを識別する。さらに、そのプログラムの問題点を抽出し、修正方針、方法を決める。レビューの負担が重い場合は、第三者を投入してレビューする。スタイルチェックは静的ソースコード解析ツールを使う ・スキル不足のプログラマが作成したプログラムをすべて修正する。修正作業は、同一チーム内のスキルのあるプログラマが分担する ・スキル不足のプログラマをプログラミングから外し、その後の障害対応のためのプログラム修正も同一チーム内のスキルのあるプログラマが分担してもらう。外したプログラマには、可能であれば、修正プログラムのリリース前修正確認、リグレッション・テストを実施してもらう ・ソースコード・レビュー後の修正プログラムについて結合テストを再度実施し、不具合を修正して、品質を上げる ・上記を実施することによる工程遅延と全体工程内での挽回策を顧客に説明し、承認を得る	・コードレビュー、ピア(peer)レビューの実施 ・プログラム修正とテストの再実施 ・スキル不足のプログラマの識別	・プログラマのスキルチェック ・レビューの励行 ・ドキュメントの記述方法の充実 ・コーディング規約の策定と徹底
77	パフォーマンス設計の不備	レスポンスの確保や無応答状態をなくすための方式設計を全くしておらず、トラフィック量やトラフィック・パターンも把握していなかった。このため、総合テストに入るころに性能問題が表面化した	多くの技術者が、「性能上の問題が起こったらハードを増やせばいい」と安易に考えていた。しかし、性能問題は設計時点で考慮すべきだった	・性能問題の状況把握 ・多くの場合、パフォーマンス・チューニングの専門家を投入 ・性能支援チームの構築 ・当初性能モデル、トラフィック条件と現時点での差異を顧客とともに確認 ・システムに求められる性能要求条件の再設定 ・性能ボトルネックの検出と改善 ・性能チューニング・性能確認テスト	・性能問題の専門化の投入 ・性能要件の確認 ・ボトルネックの抽出と分析 ・対処方法の検討と実施 ・顧客との性能要件の合意	・有識者による方式設計レビュー ・顧客への性能要件の確認