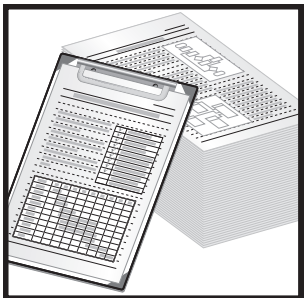


報告書 顧客との賢いコミュニケーションのために

コミュニケーションツールとしてのドキュメントの働きを考える

Text=株式会社オージス総研　正木威寛（PMP）MASAKI, Takehiro　山内奈央子 YAMAUCHI, Naoko

ドキュメントの役割を考えるとき、「お客様とのコミュニケーションを図る」ということがきわめて大切です。本稿ではこの観点から、コミュニケーション不全が起こすトラブルやそれらを防ぐための心構えを考察し、さらにそれらを助けるためのツールとしてドキュメントを紹介していきます。多くのサンプルも用意していますので、それぞれのプロジェクトに合わせてぜひ活用してください。

システム開発の現場では、開発を依頼したお客様と、システム開発を担当するエンジニアとの間でやりとり（コミュニケーション）をしながら開発が進められます。これには、要求定義書や設計書などの公式に納品される文書もあれば、開発の進捗や今抱えている問題など、開発の経過に関する文書やメール、あるいは口頭でのやりとりも含まれます。

どのようなやりとりであれ、お客様とのコミュニケーションでは、伝えるべきことを伝えているか（コミュニケーション内容）、それは正確に伝わっているか（コミュニケーション手段）、そしてメールや文書テンプレートなどのツール、この3つが重要です。もしどれかが不十分であれば、それがコミュニケーションギャップやミスコミュニケーションとなり、プロジェクトの深刻な問題となることがあります。

本稿では、上記3つに対応し、コミュニケーションのポイントを、

1. コミュニケーション内容を改善する考え方
2. コミュニケーション手段を改善するプラクティス
3. コミュニケーションに役立つかゆいところに手が届く文書

に分けて解説します。1と2についてを正木が、3については山内が述べます。

コミュニケーション内容を改善する考え方

コミュニケーション内容が合っていない、伝えるべきことが伝えられていない、といった「コミュニケーションギャップ」や「ミスコミュニケーション」は、コミュニケーションの当事者であるお客様とエンジニアの双方に原因が考えられます。ここでは、エンジニア側の問題に焦点を当てて、顧客志向やミッション志向の仕事を遂行するプロのエンジニアとしての仕事への取り組み方がコミュニケーションを改善させることを、いくつかのサンプルケースを引き合いにして説明します。

お客様から見える役割で振る舞う

あなたはある日、A社のシステム開発プロジェクトに加わることを上司から告げられました。プロジェクトにおける、あなたの役割は、プログラミングチームのメンバー（プログラマ）で、技術的かつ重要な判断はチームリーダーが行うのでそれに従うようにとのことでした。実際にチームに加わると、チームリーダーはあまり技術的なことはわからず、お客様は定例のミーティングでたびたび技術的な質問をあなたにしてきました。しかし、それはチームリーダーが答えてくれるものだとあなたは思い、よく「私にはわからないのでチームリーダーに訊いてください」と回答していました。あなたは自分のプロジェクトでの役割を十分に果たしていると思っていました。ところが何日か経ったある日、プロジェクトから外され、他のエンジニアに交代させられました。

これはなぜでしょうか？ あなたは、上司から指示された役割を果たしていたはずです。しかし、あなたがプロジェクトに加わる前にお客様に提出した提案書や契約書、プロジェクト計画書（以降、まとめて「契約」と呼ぶ）に、あなたの役割が技術的か

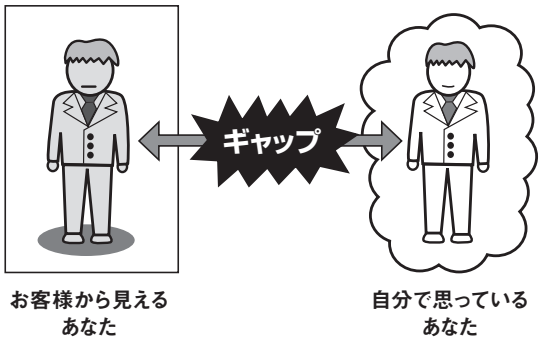


図1　お客様から見えるあなた、自分で思っているあなた

つ重要な判断を行う「アーキテクト」として記されていたらどうでしょう？ プロジェクトが取り組むべき技術的な事柄についてお客様が質問をしても、アーキテクトのあなたが技術的な見解や今後の方向性を述べることはありません。その結果お客様は、フラストレーションの溜まるコミュニケーションを強いられ、技術的な判断を何もできない半人前のアーキテクトとしてあなたを評価していたかもしれません。また、その結果引き起こされた問題は、あなたのせいになっているかもしれません。

エンジニアは、このようなトラブルに陥らないために、お客様から見えている役割を正しく認識し、振る舞うように心がけることが重要です（**図1**）。つまり、自社がお客様と交わした契約には必ず目を通し、そこに書かれた役割に基づいて仕事をすべきです。もし、プロジェクトに加わるときに上司から聞いた役割と契約上の役割が異なるのであれば、役割が変更になったことについて会社間で合意しているのかどうか上司に確認しなければなりません。契約をする営業担当者やプロジェクトマネジャーには、このような契約内容についてプロジェクトメンバーに積極的に周知しない人もいるのですが、これは自分達からミスコミュニケーションやトラブルを誘発しているようなものです（**図2**）。

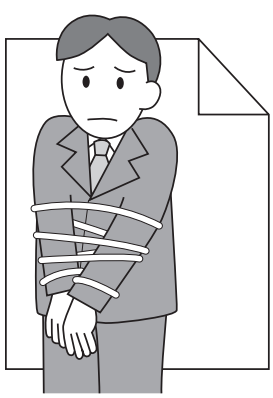


図2　ミスコミュニケーションが大きなトラブルに発展する

作業成果物の権利に気を配る

あなたは、B社から請け負ったシステム開発プロジェクトに加わりました。役割はソフトウェア開発チームのチームリーダーで、プロジェクト計画書の体制図にもそのことは書いてありました。ある日、お客様から設計者やプログラムに著作権表記（Copyright）が無いので、B社の著作権表記を追加するように相談がありました。あなたは、B社から依頼された案件ですのでそれに同意しました。ところが、それからしばらく経ってB社と自社との間に著作権に関する深刻なトラブルが起きました。あなたは上司に呼び出され、著作権表記をB社にしたことを責められました。結局、プロジェクトから外され、懲戒処分を受けました。

これはなぜでしょう？ 自分の役割もプロジェクト計画書で確認していますし、B社がお金を出している開発なのでB社の著作権表記を追加しただけです。じつは、これも契約を無視した仕事への取り組み方が原因です。契約を無視して仕事をすると、たんなる現場のミスコミュニケーションのトラブルでなく、ひどいときには企業間の裁判に発展することもあります。

典型的なのが、このような著作権に関するトラブルです。著作権法は無方式主義、つまり著作物を作成した時点で自動的に著作者である開発側の企業にその権利が発生します。したがって、お客様と開発側の企業との間の契約書に「著作権はお客様の企業にあること」が特に明記されていなければ、設計書やプログラムなどの著作権は、開発側の企業（つまり、あなたの企業）にあります。開発している企業に対価を払っているからといって、お客様にそのソフトウェアの著作権があるのではないのです（**図3**）。

プロジェクトメンバー（場合によってはプロジェクトマネジャーまでも）が契約書に目を通さずに仕事をしているのを目にすることがありますが、このような開発の現場ではお客様と開発側の担当者どうしの勝手な合意によって、設計書やプログラムにお客様の企業の著作権表記を書いてしまうことが多いようです。しかし、開発側の企業が今後の知的財産とするつもりで、安価に開発を請け負っていたとしたらどうでしょう？ 気づいたときにはプログラムはお客様の知的財産になっており、最悪の場合は裁判に発展し、あなたも責任を問われます。別のケースでは、お客様に著作権があるとしたある案件のプログラムを、別のエンジニアが契約内容を確認せずに「作業効率が良いから」という理由だけで別の案件に使ってしまい、問題になったということもあります。

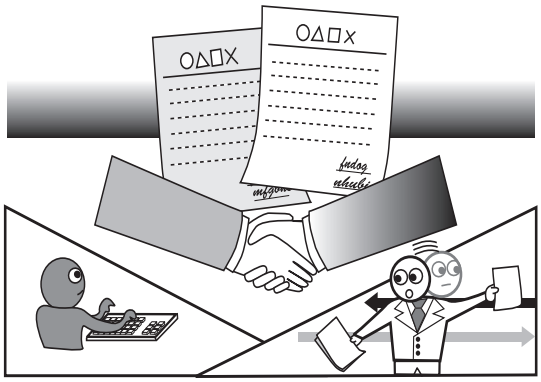


図3　契約に基づいて仕事をする

プロフェッショナルに仕事をする

あなたはある日、C社のシステム開発プロジェクトに加わることを上司から告げられました。あなたのプロジェクトでの役割はテストチームのリーダーですが、このような役割は初めてなので不安でした。ところが、上司は「請負契約ではなく準委任契約なので、お客様に指示されるとおりに作業すれば問題ない」と言い、ひとまず安心してプロジェクトに参画しました。その日からテスト設計やテストケース作成の手順をお客様に聞きながら、忠実に作業をしていました。ところがある日、お客様からあなたの上司へ、あなたはテストチームのリーダーとして失格だという苦情が寄せられました。プロジェクト計画書の自分の役割も、契約書の著作権も確認しておいたので、何も思い当たることがありません。最終的には、あなたはプロジェクトから外され、他のエンジニアに交代させられました。

なぜでしょう? あなたは役割を果たしていたし、著作権も確認しています。しかし、ここではプロとしての仕事への取り組み方に問題があります。IT業界の契約では、お客様が成果物に対して対価を払う請負契約と、作業に対して対価を払う準委任契約[注1]が用いられています。準委任契約は、弁護士の弁護や医者の治療行為などと同じ契約と思っていただければイメージしやすいでしょう。つまり、プロフェッショナルのための契約形態なのです。

図4　プロフェッショナル

残念ながら、IT業界には「準委任契約」と、雇い主に言われたとおりに労働を提供する「派遣契約」の区別がついていない人も多いようです。そのような人は、「作業成果物に責任が無いから楽だ」とか「お客様の言われたことをやりさえすればよいので楽だ」と言います。しかし、準委任契約が保証しないのは、弁護士が裁判に勝つことを保証しない、あるいは医者が患者の病気が完治することを保証しないのと同じように、先の例ではテスト結果は保証しないということです。一方、弁護士が依頼人に弁護の方法を聞きながら弁護することがないことや、医者が患者に治療方法を訊きながら治療することがないように、先の例では、テスト要求の収集、テスト設計やテストケースの作成を自身で遂行できなければいけません。つまり、弁護士は弁護という行為に責任を持って遂行しますし、医者が治療行為を責任を持つのと同様に、エンジニアは担当した役割とその作業を、責任を持って遂行できなければならないのです(**図4**)。

テストエンジニアであれば、テスト要求をインタビューすることはあっても、テスト設計やテストケース作成の進め方や技法には口出しはさせないぐらいの気迫があってもよいでしょう。

このプロフェッショナルとしての態度や規範は「プロフェッショナル倫理」と呼ばれます。弁護士や医者だけでなくIT業界にもプロフェッショナルを認定する団体があります。たとえばISACA[注2]のCISA(Certified Information Systems Auditor:公認システム監査人)やPMI[注3]のPMP(Project Management Professional)などです。このような団体やSIを請け負う一部の企業では、独自にプロフェッショナル倫理の規定を策定しています。プロフェッショナル倫理規定が自社にあっても無くても、プロとしての心構えを持って仕事に取り組めば、お客様とのコミュニケーションにおいて冒頭のようなトラブルは起きません。

注1　委任契約は、弁護士など法律にかかわる事務だけに使われ、準委任契約もそれ以外は基本的に同じです。業務委託契約やSES契約(システム・エンジニアリング・サービス契約)も準委任契約に準じます。

注2　http://www.isaca.gr.jp/

注3　http://www.pmi-tokyo.org/

コミュニケーション手段を改善するプラクティス

ここまでは、コミュニケーションの当事者の片方である我々エンジニアの態度を中心に、改善ポイントを述べました。ここからは、もう片方であるお客様とのコミュニケーション手段を改善するために、実際に筆者らが実践しているプラクティスを紹介します。

コミュニケーションチャネルを明確にする

「コミュニケーションチャネル」と言うとメールや電話やミーティングなどの媒体や場を表しますが、ここではその相手も含めて考えます（**図5**）。前節では、エンジニアはお客様から見える役割を取るべきであることを述べましたが、コミュニケーションの相手であるお客様の役割やその手段を明確にしておかなければ、これもミスコミュニケーションの原因となります。なぜなら、お客様は開発のプロではないので、開発側から積極的にお客様の役割を確認していかないと、曖昧なまま進行してしまうケースが多いからです。

このようなケースでお客様側に何人かの関係者がいる場合、

- 要求定義書や設計書などの作業成果物を承認する役割なのかと思っていたら、たんなるオブザーバーだった
- 打ち合わせや状況報告にたまにしか参加しないのでオブザーバーだと思っていたら、承認者だった

というようなことがあります。このようなことが起きると、予定どおりに承認されてオンスケジュールで進んでいたはずのプロジェクトに、突然「真の承認者」が現われて、手戻りやスケジュール遅延といった事態が生じます。プロジェクトのキックオフミーティングでは、少なくともコミュニケーションに関する次の事柄は確認しておきたいものです。

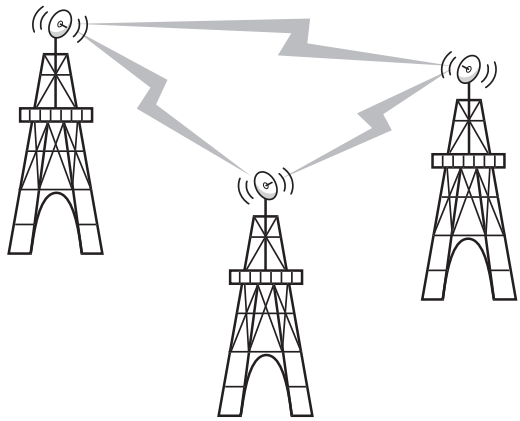


図5 コミュニケーションチャネル

- ソフトウェア要求のヒアリングなど各工程の情報源やその手配を行うお客様の窓口
- ユーザーインターフェイス仕様／設計書／プログラムなど各工程の作業成果物の承認者
- 納品時の成果物一式の検収者
- プロジェクトの状況報告などプロジェクトマネジメントに対する承認者
- それぞれの伝達手段（口頭／メール／会議 etc.）や承認方法（目視／レビュー etc.）

またひとつの作業成果物の承認者が複数のお客様で構成される場合は、全員の合意が必要なのか、参加者だけでよいのかも確認しましょう。全社の場合は、1人でも参加できなければスケジュールの遅延につながるので、そのことをお客様に理解していただく必要があります。

「共通のことば」に合意する

お客様がまったくITに知識が無ければ開発に関する用語は開発側のことばで結構なのですが、ユーザー企業の情報システム部門や情報システム子会社などITに関して知識がある場合は、コミュニケーションするときの「ことば」に気をつける必

選択肢	ことばの基準	採用の条件	まずやるべきこと
選択肢1	お客様の基準にしている開発プロセス	お客様の開発プロセスがすでに文書化されていること	開発側の開発プロセスとの対応関係を合意しておく
選択肢2	開発側の開発プロセス	開発側の開発プロセスがすでに文書化されていること	お客様に開発側の開発プロセスを説明し理解していただく
選択肢3	ISO12207[注]や市販の開発プロセス	双方が入手あるいは購入できること	双方が開発プロセスとの対応関係を合意しておく

表1 ことばの基準の選択肢

注：ISO／IEC 12207:1995 ソフトウェアライフサイクルプロセス

要があります。なぜなら、開発側が作業スケジュールの進捗や作業成果物のレビューをするときに用いる「ことば」の意味が異なることが非常に多いからです。

たとえば、「要件定義」「外部設計」「実装」「統合テスト」など、一見コミュニケーションできているつもりでも、お客様が基準にしている開発プロセスと、開発者側の開発プロセスが異なれば、作業内容やゴールあるいは作業成果物が多かれ少なかれ異なります。したがって、プロジェクトの最初の作業として、**表1**（前頁）のようにお客様の基準にしている開発プロセスと自分達の開発プロセスをすり合わせる必要があります。筆者が過去に関係したプロジェクトでも、ことばの意味の相違から順調だったはずのプロジェクトが突然承認を得られなくなったり、お客様の段取りが開発側の予想していたものと異なっていたために、追加作業によるスケジュール遅延や言われのない不信感を買ってしまったというケースを見ています。

プロジェクトのイベントは明確に

コミュニケーションチャネルで述べたプロジェクトのイベント、要求のヒアリングや作業成果物の中間レビュー／承認レビューは、事前にお客様やエンジニアとスケジュールを取り決めておいても、開催するときには忘れられていることがあります。このようなプロジェクトでは、ソフトウェア要求をヒアリングしているはずが、いつのまにか技術的な実現方法の話題になっていたり、承認レビューのはずがコメントをするのみの中間レビューになってしまったりで、ヒアリングやレビューをだらだらと繰り返します。また、いつまでも承認が下りずスケジュールの遅延を起こすか、時間切れで承認してはいけないレベルのものを承認してしまい、後に大きな問題になってきます。

このようなトラブルを回避するために、プロジェクトのイベントをとり仕切る担当者は、

- そのイベントが始まるときに、必ずこの集まりが何の集まりなのかを参加者に告げる
- そのイベントが終わるときには、次回はどのようなイベントか確認する

といったことに気をつける必要があります（**図6**）。

図6 これから何をするのか、次は何をするのか

たとえばイベントの開始時に、「今回はソフトウェア要求のヒアリングを行います」「今回は、前回のヒアリングを元に作成してきたソフトウェア要求仕様書の中間レビューを行います」「今回は、前回の中間レビューの指摘事項を反映したソフトウェア要求仕様書の承認レビューを行います」などの宣言をします。あるいはイベントの終了時に「次回はソフトウェア要求のヒアリングの続きを行います」「次回はソフトウェア要求仕様書に本日のヒアリングの結果をまとめてきますのでその中間レビューを行います」「次回はソフトウェア要求仕様書の最終確認として承認レビューを行います」といったひと言を述べ、参加者全員に同意を得るのです。

簡単なプラクティスですが、お客様をはじめとする関係者が、自分達がいま開発ライフサイクルのどこにいて、このイベントは何のために行うのか、それによって次は開発ライフサイクルのどこへ移るのか、といったことを再認識でき、コミュニケーションの質や参加意識が高まります。

プロジェクトに小さな締め切りを作る

上流の作業であればあるほど人間的／概念的になるため、曖昧さや不完全さはどうしても排除できません。また、お客様の業務要求やソフトウェア要求などを開発者がヒアリングして仕様化する場合、本当に伝えるべき要求を伝えたのか、開発者の分析内容が正しいのか、お客様自身もわからないことが少なくありません。

このような工程、たとえばソフトウェア要求仕様書の承認レビューなどは、先述のプラクティスをもってしても、お客様からダメ出しが繰り返され、要求定義者は作業遅延を起こします。たんなる遅延で済めばよいのですが、結局は大きな作業遅延がプロジェクト全体の遅延につながったり、プロジェクト終盤の激務

を引き起こしたりすることが多くあります。これを回避するには、いつまでもたんなる作業遅延として扱うのではなく、ある時点でプロジェクトの問題としてプロジェクトマネジャーが対応することです。そこでお薦めなのが、タイムボックス開発です（**図7**）。

タイムボックス開発では、一般的な開発で2か月±1か月、アジャイルなどの小規模開発では2週間程度のタイムボックス（時間の箱）の連続としてプロジェクトを捉えます。このタイムボックスの終わりには、必ず締め切りが設定され、延長は認められません。つまり、ソフトウェア要求仕様書を1か月のタイムボックスで完成させると決めたら、担当のお客様とエンジニアはなにがなんでも完成させなければいけません。完成と言っているのは、内容に矛盾のないことを指しますが、「もっと詳細な図や絵が欲しい」、「要求がまだあるかもしれない」ということは許容されます。

図や絵などのドキュメントのグレードや、あるのか無いのかわからない要求については、締め切りに間に合うようであれば検討し、間に合わなければプロジェクトマネジャーと資金を提供するお客様がプロジェクトマネジメントとしての判断を下します。つまり、それらに対して「次のタイムボックスでも、さらに検討を続ける」「机上で続けるのはやめて、プロトタイプを作成し、要求を評価する」「実際に開発して再評価する」「この際、曖昧な要求に対する投資はやめる」などの戦略を立て直します。こうすることにより、手遅れになるまで担当者に委ねておくのではなく、問題を早く顕在化させて、プロジェクトとして利害関係者全員で共有し、プロジェクトとしての対処を促進します。

指摘を管理する

タイムボックス開発では、やり残した事柄をタイムボックスの終わり（締め切り）の時点でプロジェクトマネジャーに明示できなければいけません。中間レビューなどで挙げられた指摘事項や担当エンジニアが自発的に気づいた改良点は、対応したのか未対応なのかをタイムボックスの実行中にも管理しておく必要があります。

レビューと言うと、結果が議事録に記録されることを想像される読者も多いと思います。しかし、タイムボックスの最中に何度も行われた中間レビューの議事録の指摘事項をすべて参照し、タイムボックスの終盤や後続のタイムボックスで対応するのは難しいものです。

この場合、指摘事項を確実に管理するには、バックログのリスト（課題表）を作成する方法が有効です。筆者らは、作業成果物に対する指摘事項は、議事録ではなく作業成果物のバックログリストに登録し、優先度や対応の有無まで管理しています。議事録には今後のレビューの日取りなどの作業成果物への指摘以外を記録し、バックログリストは議事録に添付して関係者へ配布することで、議事録作成の手間を軽減し、指摘も管理できるようにしています。

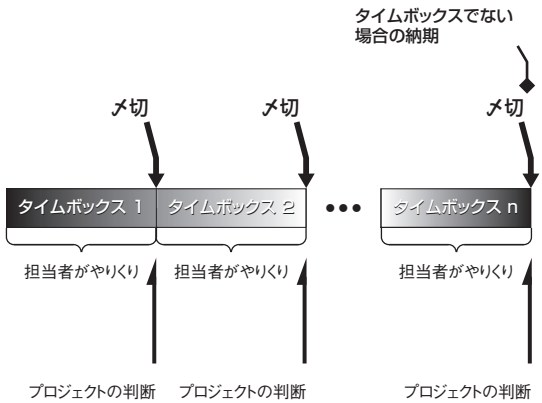


図7 タイムボックス開発

かゆいところに手が届く文書

以上のように、お客様とのコミュニケーションにおける考え方の基本とプラクティスをまとめてみました。ここからは、お客様とコミュニケーションを図るツールとしてのドキュメントを考えていきます。

システム開発のなかでソフトウェアの仕様が記述される公式な文書はもちろん重要ですが、そのほかにもさまざまな文書が作成されます。そのなかにはこまごまとしたものもありますが、それらを適切に作成していなかったばかりに、お客様とのコミュニケーションが思わぬところでうまくいかなくなるケースも考えられます。この節では、先の2つの節の内容に基づき、コミュニケーションの「かゆいところに手が届く文書」として以下に焦点を当て、具体的なサンプルを挙げながら説明します。

- 質問票
- 調査報告書
- レビュー記録
- バックログリスト
- 障害報告書

質問票

質問票はその名のごとく、相手に質問したいことを書き、回答してもらうための文書です。

システム開発の作業を進めるうえで、お客様に質問しなければならない場合があります。たとえば、ソフトウェア要求仕様書の曖昧な点について詳細を確認する場合や、要求に矛盾点を発見したため、どちらを優先させたいのかを確認する場合などです。お客様と私たちエンジニアがすぐ近くの席にいて、文書以外のコミュニケーションが容易な場合には、些細な質問をわざわざ質問票でやりとりする必要はないかもしれません。しかし、お客様との距離が離れている場合や、質問事項を文書で受け取り、まとめて回答するほうがよいと判断される場合、「言った言わない」の問題を避けたい場合には、質問票を使ってやりとりするとよいでしょう。また、大規模プロジェクトになると、お客様だけではなく、同じシステム開発プロジェクトのチーム間でも質問票がやりとりされることがあります。

サンプル1に、質問票の例を示しています。イメージしやすいように、フォーマットに具体的内容を記入しました。補足説明が必要な項目については、図中に番号を付けて説明を記述しています。また、サンプル1の吹き出し番号について、項目の意味や記述する際のポイントを以下に説明します。

①分類：
質問内容を分類します。分類は、ビジネスルール、ユーザーインターフェイス、環境など、プロジェクトであらかじめ決めておきます。

②件名：
質問の内容を端的に表現した名前を付けます。コミュニケーションしやすくするために、名前は重要です。質問票のステータスを一覧で管理する場合にはこの件名を用いるため、どのような質問内容なのか推測しやすい名前を付けると、一覧を見ただけで内容が思い出しやすくなります。質問票の一覧については、サンプル2を参照してください。

③回答希望日：
質問する相手に、いつまでに回答してほしいのか伝えられます。回答する側も、急ぎなのかどうかを判断できるため、コミュニケーションしやすくなります。

④質問内容：
質問事項は、1つの質問票に1つにするほうが件名も付けやすく、管理しやすくなります。

サンプル2は、質問票の一覧表です。個々の質問票が、現在どのようなステータスになっているのかを一覧形式で参照するようになっています。誰がどのタイミングで一覧表に登録し、ステータスを更新するのかはプロジェクトで決めておく必要があります。質問票の運用に特別なソフトウェアを用いない場合は、質問票を取りまとめる担当者が更新する、もしくは、それぞれの質問者が更新するといった運用方法があります。

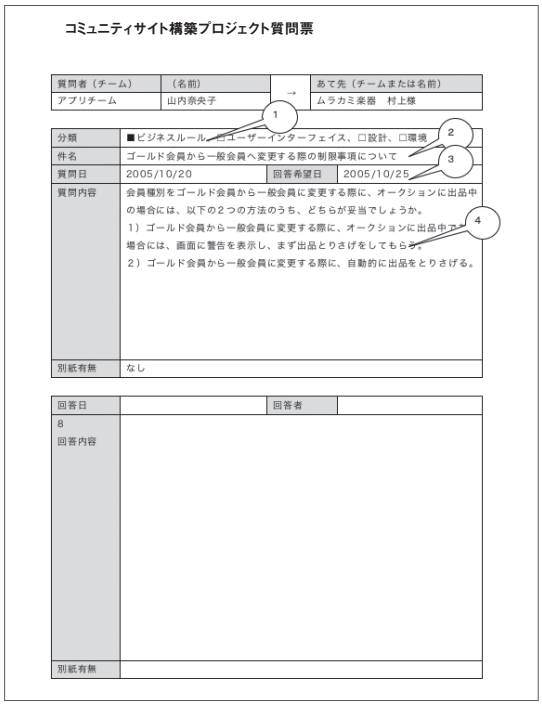
調査報告書

システム開発の初期などに、利用しようとする技術が本当に使えるものかどうかを調査することがあります。たとえば、「外部システムと接続するためのフレームワークが本当に使えるかどうか確認しておきたい」といったことです。また、購入を検討しているテストの自動化ツールに対し「必要な機能が揃っていそうだけど、トライアル版を使って操作性も含めて確認しておこう」といった調査は、日常よく行われていると思います。調査報告書とは、その調査結果や結果に対する考察をまとめたものです。

調査報告書を書く際には、これから書こうとしている調査報告書が、システム開発のプロジェクトの中でどのような役割を担うのかを知っておくことが重要です。たとえば、あるフレームワークがプロジェクトで利用できそうか調べる際に、求められていることは何でしょうか。それは「利用できる」または「利用できない」という判断です。したがって、たとえ詳細に調査を行い、結果をきちんとまとめたとしても、考察として「だからどうなのか?」といった指針が示されていなければ、お客様の満足が得られにくく、注意が必要です。

調査報告書の目次のサンプルと、記載する内容の説明を、以下に示します。

- 調査の目的：
  何のために調査を行うのかを説明します。
- 調査の方法：
  どのようにして調査をしようとしているのかといった戦略、その妥当性も含めて説明します。
- 調査環境：
  PCの型番やメモリ容量、利用するソフトウェアのバージョンなど、読み手が同じ環境で調査を再現できるレベルまで詳細に記述します。
- 調査手順：
  調査を行うための作業手順や調査項目を説明します。
- 調査結果：
  手順に従って測定したデータです。量が多くなってしまう場合には、別紙にします。わかりやすくするために図や表を用いることも有効です。画面のハードコピーを用いる場合は、どこが結果の部分なのかを色で囲むなど、読み手にわかりやすいように工夫します。



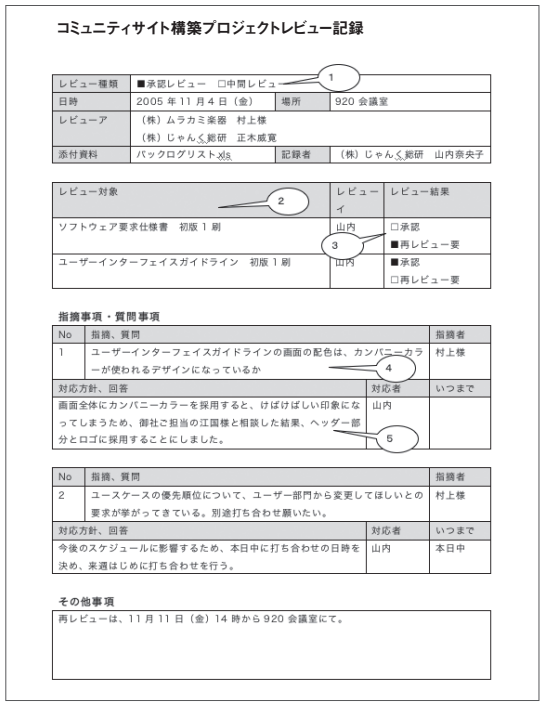
コミュニティサイト構築プロジェクト質問票

質問者（チーム）	（名前）	→	あて先（チームまたは名前）
アプリチーム	山内奈央子		ムラカミ楽器　村上様

分類	■ビジネスルール、□ユーザーインターフェイス、□設計、□環境		
件名	ゴールド会員から一般会員へ変更する際の制限事項について		
質問日	2005/10/20	回答希望日	2005/10/25
質問内容	会員種別をゴールド会員から一般会員に変更する際に、オークションに出品中の場合には、以下の2つの方法のうち、どちらが妥当でしょうか。 1）ゴールド会員から一般会員に変更する際に、オークションに出品中の場合には、画面に警告を表示し、まず出品とりさげをしてもらう。 2）ゴールド会員から一般会員に変更する際に、自動的に出品をとりさげる。		
別紙有無	なし		

回答日		回答者	
回答内容			
別紙有無			

サンプル1　質問票



コミュニティサイト構築プロジェクトレビュー記録

レビュー種別	■承認レビュー　□中間レビュー		
日時	2005 年 11 月 4 日（金）	場所	920 会議室
レビューア	（株）ムラカミ楽器　村上様 （株）じゃんく郷研　正木威寛		
添付資料	バックログリスト.xls	記録者	（株）じゃんく郷研　山内奈央子

レビュー対象	レビューイ	レビュー結果
ソフトウェア要求仕様書　初版 1 刷	山内	□承認 ■再レビュー要
ユーザーインターフェイスガイドライン　初版 1 刷	山内	■承認 □再レビュー要

指摘事項・質問事項

No	指摘、質問	指摘者
1	ユーザーインターフェイスガイドラインの画面の配色は、カンパニーカラーが使われるデザインになっているか	村上様

対応方針、回答	対応者	いつまで
画面全体にカンパニーカラーを採用すると、けばけばしい印象になってしまうため、御社ご担当の江国様と相談した結果、ヘッダー部分とロゴに採用することにしました。	山内	

No	指摘、質問	指摘者
2	ユースケースの優先順位について、ユーザー部門から変更してほしいとの要求が挙がってきている。別途打ち合わせ願いたい。	村上様

対応方針、回答	対応者	いつまで
今後のスケジュールに影響するため、本日中に打ち合わせの日時を決め、来週はじめに打ち合わせを行う。	山内	本日中

その他事項

再レビューは、11 月 11 日（金）14 時から 920 会議室にて。

サンプル3　レビュー記録

コミュニティサイト構築プロジェクト

No	分類	質問名	質問者		あて先		質問日	回答日	ステータス
1	ビジネスルール	ゴールド会員から一般会員へ変更する際の制限事項について	アプリチーム	山内	ムラカミ楽器	村上	2005.10.20		回答待ち
2	設計	全社標準のテーブル命名規則における例外事項について	アプリチーム	正木	SAチーム	高木	2005.10.21	2005.10.21	回答済
3	ビジネスルール	複数ID取得者のチェック機能の条件項目について	アプリチーム	山内	ムラカミ楽器	村上	2005.10.21		回答待ち

サンプル2　質問票一覧

コミュニティサイト構築プロジェクト

No.	分類	バックログ内容	登録日	成果物	登録者	指摘場所	優先度	対応完了日	対応者	対応方針、対応内容など	済
1	課題	「人気コミュニティランキング」のランキング基準が未決定。	2005.11.2	ビジネスルール	山内	承認レビュー	低			11/10のミーティングまでに確定する予定。	
2	改善	リモートホスト別のアクセスログは、サイトへの足跡などアクセス数の多さを見ることが多いため、ホストの名前順だと見つけづらい。リモートホスト別のアクセスログは、アクセス数順の方がわかりやすいのではないか。	2005.11.3	アクセスログ画面仕様	正木	承認レビュー	並			アクセス数順に変更する方針とする。	
3	不具合	ログイン失敗時のメッセージだけ「ユーザーIDとパスワードを確認せよ」と命令口調になっているのはおかしい。	2005.11.4	ログイン画面仕様	山内	中間レビュー	高	2005.11.5	山内	「ユーザーIDとパスワードを再度確認してください」に変更した。2005/11/5 承認済み。	済

～（後略）～

サンプル4　バックログリスト

- ●考察：
  調査結果から何がわかったのかを記載します。調査結果だけではなく、調査目的に対してどのようなことが言えるのかを分析し、説明することで、調査報告書によって読み手に何が言いたいのかを伝えられます。

- ●参考文献：
  調査の際に利用した参考文献を記載します。読み手が、どの文献なのか探せるように、以下の項目を記載します。本の場合には、文献名／出版年／出版社／版刷／著者を記載します。インターネット上の文献についてはURLを記載します。

レビュー記録

公式なレビューを行った際に、そのレビューで決まったことや、指摘事項、レビューの結果を記録しておきます。レビュー記録を作成し、レビューの出席者へ配布することによって、認識が合っているかどうかを確認でき、コミュニケーションの助けとなります（**サンプル3**）。後述のバックログリストと併用する場合には、成果物に対する個々の指摘事項はバックログリストに記載し、

レビュー記録の添付資料とします。一方、レビュー記録には、承認／却下の結果や、質問事項、今後のレビューの日程などを記載します。

サンプル3の吹き出し番号について、項目の意味や記述する際のポイントを以下に説明します。

①レビュー種類：
承認レビューなのか、中間レビューなのか、といったレビューの種類を記載します。

②レビュー対象：
成果物の名称に加えて、版刷（またはバージョン）も含めて記載しておくと、後から見たときにレビューの対象が明確になります。

③レビュー結果：
承認レビューの場合には、対象の成果物が承認されたのか、再度レビューが必要なのか、はっきりわかるようにしておきます。

④指摘／質問：
レビューアからの指摘事項や質問事項を記載します。バックログリストを併用している場合には、個々の指摘事項はバックログリストに記載し、レビュー記録には記載しません。バックログリストを併用しない場合には、この欄に指摘事項を記載します。

⑤対応方針／回答：
指摘事項に対し、どのような対応をとることになったのかを記載します。質問に対しては、回答を記載します。バックログリストを併用している場合には、個々の指摘事項への対応はバックログリストに記載し、レビュー記録には記載しません。バックログリストを併用しない場合には、この欄に対応方針を記載します。
また、レビューのなかで具体的な結論が出ない場合もありますが、その場合は誰が再検討するのか、といった方針を記載しておきます。

バックログリスト

レビューでの指摘事項や、タイムボックス開発での積み残し作業（タイムボックス開発については前のセクションを参照してください）を管理するための文書です。作業の内容だけではなく、優先度や対応方針、対応した結果などを登録し、現在挙げられている課題がどのようなステータスなのかを管理します。プロジェクト内部に周知するためにも、またお客様に説明する際にも有効に活用できます。

サンプル4（前頁）に、バックログリストの例を示しています。吹き出し番号ついて、項目の意味や記述する際のポイントを以下に説明します。

①分類：
バックログの内容を分類します。サンプルでは、どのように解決すべきかわかっていない事項（課題）、対応方針までわかっている事項（改善）、誤っている事項（不具合）の3種類に分けています。「不具合」の場合の優先度は「高」にするなど、優先度を決定する際の判断基準にすることもできます。

②バックログ内容：
レビューでの指摘事項や、積み残し作業、改善すべき点などを記載します。改善すべき点を記載する場合は、改善案だけを記載するのではなく、経緯や問題点も含めて記載しておくことが望ましいです。なぜなら改善案はバックログを登録した時点での案でしかないからです。経緯や問題点は、実際に改善する時点で本当にその案が妥当かどうか検討する際の良い材料となります。

③優先度：
高／並／低のいずれかに分類しています。

④対応方針／対応内容など：
バックログに対して、どのように対応する予定なのか、またはどのように対応したのかについて書いておきます。

障害報告書

システムに起きた障害について、どのような障害がいつ起きたのかを記録し、報告する文書です。障害報告書は、システム開発中のテストで発生した障害を報告する場合と、本番リリース後に発生した障害を報告する場合の2つのケースで活用でき、システムの品質の良し悪しを判断する材料になります。どちらのケースにおいても、報告された障害は、対応状況を管理するためにExcelにまとめられたり、障害管理のためのソフトウェアで管理されたりします。したがって、障害報告書を誰が受けつけて、どのように管理していくのか事前に決めておく必要があります。

サンプル5は、システム開発中のテストで発生した障害を報告するための障害報告書です。

「テスト」とひと言で言っても、プロジェクトによって定義の仕方はさまざまですが、一般的に単体テストで障害報告を書くことはあまりないと思いますので、単体を統合したビルドに対するテストを想定しています。また、サンプルでは障害報告のフローとして、障害の報告→原因の調査→修正担当の割り当てや優先度の決定→修正とテスト、という流れを想定し、すべてを1つの文書にまとめています。

サンプル5の吹き出し番号について、項目の意味や記述する際のポイントを以下に説明します。

①障害ID：
障害報告の管理方法に従って採番されるIDです。

②障害名：
発生した障害の内容を端的に表現した名前を付けます。コミュニケーションをしやすくするために名前は重要です。

③テストレベル：
統合テストやシステムテストなど、プロジェクトで定義されたテスト名を記載します。

コミュニティサイト構築プロジェクト

ID	障害名	報告日	報告者	重要度	ステータス
IT-0034	会員削除画面で、管理者ユーザーが会員を削除できない	2005.12.12	山内	低	対応済み
IT-0035	会員検索画面で、ヒット数が0件の場合、画面のレイアウトがくずれる	2005.12.13	山内	中	調査済み
IT-0036	月末のマスタデータ洗いがえ処理時に、異常終了	2005.12.14	正木	ー	未対応

サンプル6　障害報告書(テスト用)一覧

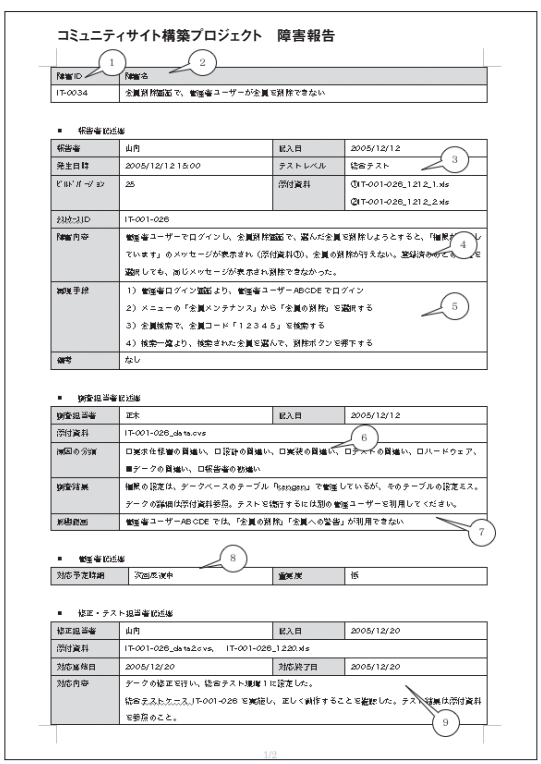
④障害内容：
どのような障害が発生したのかを説明します。障害がどのような条件で発生したのか、曖昧にならないように説明します。たとえば、「画面がぐちゃぐちゃになった」という表現ではよくわかりません。レイアウトの崩れ方の特徴や、出力されたエラーメッセージを記載するなど、読み手がわかりやすいように心がけます。

⑤再現手段：
どのような手順で作業すると、この障害が再現するのかを説明します。障害対応の担当者にとって重要な情報となります。

⑥原因の分類：
原因がどこにあったのかを分類します。あらかじめプロジェクトで選択肢を用意しておくと便利です。たとえば、要求仕様書の間違い、設計の間違い、など各工程における主な成果物で分類する方法があります。

⑦影響範囲：
この障害によって、他の機能や、今後のテストにどのような影響を及ぼすのかを記載します。

⑧対応予定時期：
障害報告と調査結果を受け、障害への対応時期を記載します。日付がはっきりしている場合には、日付で記入します。「現在の反復内」「次回以降の反復」など、幅を持たせて記載する場合もあります。



コミュニティサイト構築プロジェクト　障害報告

障害ID	障害名
IT-0034	会員削除画面で、管理者ユーザーが会員を削除できない

■　障害報告記述編

報告者	山内	記入日	2005/12/12
発生日時	2005/12/12 15:00	テストレベル	結合テスト
ビルドバージョン	25	添付資料	①IT-001-026_1212_1.xls ②IT-001-026_1212_2.xls
対応つけID	IT-001-026		
障害内容	管理者ユーザーでログインし、会員削除画面で、選んだ会員を削除しようとすると、「削除されています」のメッセージが表示され（添付資料①）、会員の削除が行えない。登録済みの[illegible]を選択しても、同じメッセージが表示され削除できなかった。		
再現手順	1）管理者ログイン画面より、管理者ユーザーABCDEでログイン 2）メニューの「会員メンテナンス」から「会員の削除」を選択する 3）会員検索で、会員コード「12345」を検索する 4）検索一覧より、検索された会員を選んで、削除ボタンを押下する		
備考	なし		

■　調査担当者記述編

調査担当者	正木	記入日	2005/12/12
添付資料	IT-001-026_data.cvs		
原因の分類	□要求仕様書の間違い、□設計の間違い、□実装の間違い、□テストの間違い、□ハードウェア、■データの間違い、□標準書の誤植い		
調査結果	権限の設定は、データベースのテーブル「kengen」で管理しているが、そのテーブルの設定ミス。データの詳細は添付資料参照。テストを続行するには別の管理ユーザーを利用してください。		
影響範囲	管理者ユーザーABCDEでは、「会員の削除」「会員への警告」が利用できない		

■　管理者記述編

対応予定時期	次回反復中	重要度	低

■　修正・テスト担当者記述編

修正担当者	山内	記入日	2005/12/20
添付資料	IT-001-026_data2.cvs、IT-001-026_1220.xls		
対応着手日	2005/12/20	対応終了日	2005/12/20
対応内容	データの修正を行い、結合テスト環境1に設定した。 結合テストケースIT-001-026を実施し、正しく動作することを確認した。テスト詳細は添付資料を参照のこと。		

1/2

サンプル5　障害報告書(テスト用)

⑨対応内容：
障害に対して、どのような対応を行ったのか説明します。修正内容だけではなく、どのようなテストを行ったのかも説明してください。修正担当者とテストを行う担当者が別である場合は、それぞれの記入欄を持つフォーマットがよいでしょう。

サンプル6は、障害報告書の一覧表です。個々の障害報告書が、現在どのようなステータスになっているのかを一覧形式で参照するようになっています。誰がどのタイミングで一覧表に登録し、ステータスを更新するのかはプロジェクトで決めておく必要があります。障害報告書の運用に特別なソフトウェアを利用せず、このサンプルのような単純なExcelシートを利用する場合は、障害報告書の管理担当者が更新する、もしくは、それぞれの障害報告書を更新する担当者が更新するといった運用方法があります。

以上、本稿では、コミュニケーション改善に向けての姿勢、改善ポイント、さまざまな文書フォーマット例、という3つについて説明してきました。システム開発のプロジェクトでは、人とのコミュニケーションが必要かつ重要な役割を果たします。本稿が開発の現場において、読者の方々の一助となれば幸いです。

なお、本記事の各種サンプルや関連データは「開発の現場」のWebページ(http://www.shoeisha.com/mag/kaihatsu/)からダウンロードできますので、ぜひご活用ください。

PROFILE

正木威寛(まさき たけひろ)
若い頃にレジの組み込みソフトの開発を通して、小売業界ではあたりまえの顧客志向の発想法を徹底的に叩き込まれる。現在はプロセスコンサルタントとして顧客志向かつ開発者志向のプロセス策定・導入に奮闘している。

山内奈央子(やまうち なおこ)
金融系や人事給与系などさまざまなシステム開発に従事。現在は、開発プロセス提供サービスである「オージステーラリングサービス」のプロセスエンジニアとして日々奮闘中。仕事は楽しく♪がモットー。