なぜレビューを実施するのか，効果的なレビューとは何か！

1.1 ソフトウェア品質保証の現状と問題点

(1)　ソフト開発における市場環境と課題

現在，開発の周期は，半年，短いものは 3 カ月と短期開発となっています。一方，品質は以前よりさらに高い信頼性が求められています。このようにソフト開発を取り巻く環境は，低価格化，短納期開発，高信頼性要求といった 3 つの異なった要求を同時に実現していく必要があるため，かなり厳しいと言えます。

短納期開発のため仕様決定も遅れがち，または仕様が曖昧(あいまい)なまま作業を進めざるを得ないといったことも発生しています。そのため，安易な対応として開発プロセス，特に品質確保のためのプロセスを省略または簡易化するプロジェクトもしばしば見られます。例えば，リスク管理不在，レビュー省略または簡略，外注への丸投げ，プロトタイプレベル(不十分な設計/テスト)のまま出荷されるという事象があります。

リソース不足を補うために関連会社, 社外企業, 海外発注などへ作業を委託する際，分担部分の品質管理も含めて，容易にすべてを相手先にまかせてしまうといった丸投げを行う傾向もあります。その結果，受入または出荷後に品質問題が発生し，対応に追われることとなります。

障害発生時には，そのリカバリーを最優先に実施する必要があります。人海戦術等による応急処置は，短期的には 1 つの手であり，当面の処置となります。ただし，この対応が長く続く，もしくは散発的に発生している場合には，コスト問題や超過勤務に伴う従事者モラルの低下へと発展します。対応技術者は疲労し，勉強意欲の喪失や疲労による二次障害の発生リスクも高くなります。次の開発着手も遅れ，ますます短納期開発をせざるを得ないという悪循環に陥ってしまいます。

火を噴くまで対策を怠っていると悪循環に陥ってしまいます。この悪循環に陥らない，もしくは脱出するために，適切なレビューの実施とリスク管理ができる仕組みの構築が必要となります。開発プロセスを省略/簡略してしまう土壌として，ソフトの品質状況把握ができていない，もしくは不十分という「見える化」不足と，適切なタイミングで効果のあるレビューが十分にできていないという根本原因があります。

ソフト開発が悪循環に陥る原因は，仕様化や効果的なレビューの仕方といった技術的問題と，必要な開発プロセスを確実かつ十分に実施させるといった管理的問題が混在していると言えます(**図 1.1**)。

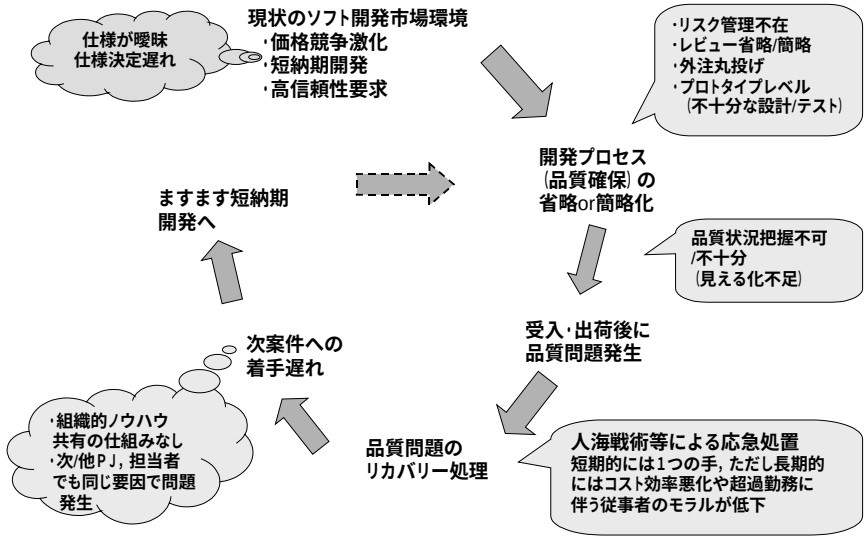

図 1.1 ソフト開発の悪循環

(2) 短期開発，コスト削減のための考え方

先に示したように，その場しのぎで開発を実施したとしても，納入後いつまでもバグや仕様変更への対応が必要となります。その結果，さらに他のシステム開発作業も

着手が遅れ，ますます短期開発になってしまうという悪循環が発生します。

これを防止するには，結局は 1 つ 1 つのシステムあるいは機能を手離れの良いものにするしかありません。そのため，ソフトウェアの品質を高める「品質保証活動」の充実化が必要となります。

その際, ともすれば「テストをこれまで以上に実施する」といった戦略もあります。しかし，もし作り込み品質が不十分であった場合，膨大なテスト工数が必要となる可能性があります。また，最終段階での品質確保では納期的に厳しいものがあります。その結果，品質確保が不十分のまま出荷せざるを得ない，もしくは納期遅延となってしまう場合が見受けられます。

そのため，上工程重視へのプロセスへと改善していく必要があります。さらに，発生したものを手直しするということだけではなく，起こした問題を組織として捉え，継続的にプロセスを改善し，作り込みの欠陥そのものを削減していくことを推進していかなければ，低価格化，短納期開発，高信頼性といった対応は困難だと言えます。

1.2 ソフトウェア開発におけるレビューの位置付け

1.2.1 レビューとは

ソフトウェア開発におけるレビューとは，ソフトウェア製品の設計品質およびそれを実現するために，ソフトウェアライフサイクルの各工程において客観的に知識や知恵を集めて評価し，改善点を提案し，次の工程に進み得る状態にあることを確認する組織活動の体系ということができます。

(1)　プロジェクトで実施するレビューの種類

プロジェクトで実施するレビューには，様々な種類のものがあります。プロジェクト管理を目的としたレビューではマネージメントレビュー，ステータスレビューやプロジェクト終了レビューなどがあります。

マネージメントレビューは, 上級管理者に情報を提供することにより, 製品のリリース，開発プロジェクトの継続(または中止)，提案の採用，プロジェクトスコープの変更，リソースの調整やコミットメントの変更を決定するなど，プロジェクトの重要案件を上位レベルの管理層と共にレビューし，課題を解決または正式決定することを目

的として実施します。

ステータスレビューは，マイルストーンに対する進捗，発生した問題の識別や計画時に設定したリスクの発生状況やリスクに対する対応策の実施と効果および今後の対応を，プロジェクトマネージャおよびメンバーで決定することを目的に実施します。

プロジェクト終了レビューは，プロジェクト完了時に，反省と今後のプロジェクトに対する教訓を得るなどの目的で実施します。

プロジェクト管理を目的に実施するレビューに対して，作業成果物そのものの品質に主眼を置いたレビューとして，開発段階前の要求仕様に関するユーザ折衝段階での契約レビュー，得意先に良い説明書を提供するための説明書原稿のレビュー，ソフトウェア開発でのレビューとして実施されているデザインレビュー(設計審査)やコードインスペクションなどがあります。

デザインレビューとは，そのソフトウェアに直接関与した設計の内容に対して，他の設計者や他の部門の関係者がレビューすることであり，コードインスペクションとは，プログラム製造担当者が机上デバックを終え，単体テストの実施が可能と判断した時点で，リスト上のソースコードを 1 ステップごとに読み進み，レビュー担当者がその良否を判断することと言えます。

CMMI[1]では，作業成果物の欠陥と改善の機会を探す目的で実施するレビューを「ピアレビュー」と呼んでいます。

このように，プロジェクトで実施するレビューは多くの種類がありますが，大きく分けると，プロジェクト管理を推進するためのレビューと直接作業成果物の品質を向上するためのレビューの 2 種類があります。

ソフトウェアのレビューは，ソフトウェアライフサイクルの各工程で，プロジェクトに合った，適切で効率的なやり方を選択し，実施することが必要です。一般的な手順に従い，その通りに実施するのも 1 つの方法ですが，それぞれのレビュー技法の特徴や考え方を理解し，そのポイントを押さえて自組織に合ったやり方にカスタマイズして導入していくことが効果のあるレビューとなります。

ソフトウェア開発時に用いるピアレビュー技法の種類や手順については，第 2 章で詳細に説明します。

---

[1] CMMI：Capability Maturity Model Integration(能力成熟度モデル統合)の略称。
Carnegie Mellon University,Software Engineering Institute の登録商標。

1.2.2 レビューの必要性

自分自身は作ったものに近づきすぎているため，自分で作った誤りや考慮不足となっている箇所が分からないといったことは，ソフトウェア開発に限らず日常的によくある事象です。漏れや勘違いを防止するためには，違う考え方をする人や様々な見方をする人が必要となります。人間はミスを犯す生き物であり，うまくいくとの期待感だけではリスクが大きいのです。特に，ソフトウェア製品は，ハードウェアのような物理的制約が少なく，その仕様を柔軟に設定することができます。そのため，レビューを十分に行い，顧客の要求に合致し，作りやすさ，保守性，拡張性に富んだ仕様や設計にしていく必要があります。

レビュー未実施の場合には，前工程で作り込まれた不良が後工程に持ち越され，手戻りが大きくなり，生産性を著しく低下させます。また，修正に伴いデグレードが発生し，品質そのものを悪化させることもあります。前工程でレビューに時間をかけると後工程の工数が減り，トータル的にはメリットがあると言われています。前工程のレビューで時間をかけないと，後工程でレビューをしても，なかなか開発工数は減少しないし，保守まで考えるといつまでも手が離れないシステムになってしまいます。開発工程と該当工数にかける関係を**図 1.2** に示します。

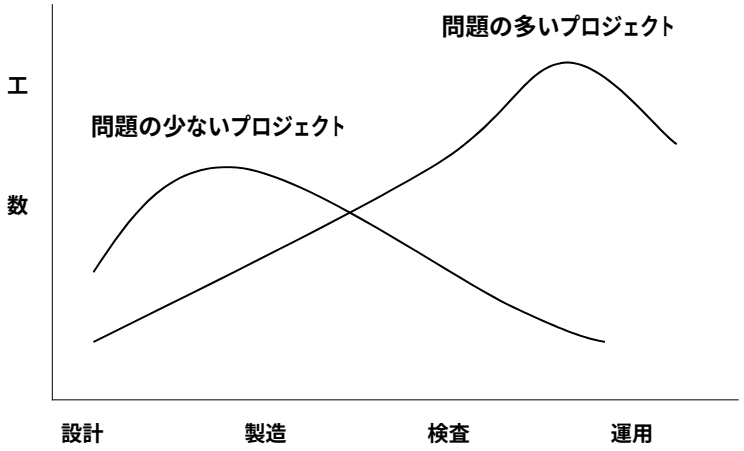

図 1.2 開発工程と工数との関係

しかし，十分なレビューとはどのようなものか，本当にこの図のようになるのかと疑問を持つ方もいることでしょう。レビューは行っていないわけではないが，テストで見つけるほうが効率的だし，レビューを行っても障害はほとんど減らない。あるいは，結局仕様変更が多すぎて，実施してもあまり効果が出たためしはない，頭で考えることと実態は違う，などの声も聞きます。

この図で特徴的な事項は，設計および製造にかける工数差の違いです。特に設計の後半から製造に至る工数の差が大きく異なる点です。レビューに関して言えば，最初の仕様を決定する段階は，問題の多い/少ないプロジェクトにかかわらず，ユーザとの関わりが強いこともあり，それなりにレビューが実施されます。しかし問題の多いプロジェクトは，その後の作業成果物に対するレビューが不十分になっていることが多いということです。特に，詳細設計やコーディングでの作業成果物あたりにくると差がはっきりとしてきます。

1.2.3 欠陥除去の段階とコストの関係

欠陥除去作業は，後になるほど修正コストが増大すると言われています。その主な原因は，修正の影響範囲が拡大し，原因究明や対策が複雑化することにあります。そのため，その原因追求と対策に時間がかかったり，再テスト/再統合などを含む後戻り作業も増大します。特に原因追求や修正に伴う関係者が増加した場合には，コストは顕著な差となって現れます。

単純障害で対応の範囲が狭い場合には，それほど大きな差は生じません。なぜなら，原因の特定も容易であり，テストの範囲も分かりやすいからです。しかし，複雑な障害や根本的な障害の場合には，原因追求や対応のために複数の人が関係することになり，大きな差となって現れます。

テストおよび運用時での障害は，同様の現象を再現させることから始まります。再現させるためには，そのための環境作り，再現手順の確認が必要となります。もし再現できない場合は，新たにトレースの機能やダンプ機能を作り込まなければならないかもしれません。その再現確認後から，問題箇所の特定作業となります。原因が判明しても対応すべき箇所が複数存在する場合には，同期を取った修正や統合およびテストが必要となります。工数をかけても，テストでは再現できない場合も多々あります。

一方，レビューは，欠陥箇所を直接指摘することができます。また，異常系やタイミングに起因するバグは，一般的に正常系と比べて再現が難しいものです。テストで

再現させようとするとコストも高くつきます。すべての異常系をテストで網羅することは難しい作業となります。テストでは発見することが困難な障害が，運用後に大問題になることも多々あります。そのため，複数の人が知恵を絞り，考えられる事項が漏れていないかを確認する必要があるのです。

これらのことから，レビューで欠陥を抽出することが効率性を高める 1 つの手段であると言えます。

1.2.4 レビューとテストとの違い

レビューは静的文書とモデルを評価します。一方，テストは成果物の動作を実際に評価します。テストでの動作確認を行う際，いくつかの問題点があります。例えば，障害修正後に次の障害が顕在化するような障害の発見があります。この場合，1 つの障害を修正しては，次の障害に出くわし，さらに修正を加えるといった，一種のもぐら叩き状態になります。一方，レビューの場合には，一度に全部を見ることができるため，次の障害修正を待つまでもなく，一度に発見し修正できるため効率的です。

表 1.1 に，レビューとテストでの主な違いを記載します。

表 1.1 レビューとテストとの違い

	レビュー	テスト
確認範囲	**文書，モデルやコードの静的な評価のみ。**	**想定環境下での動作を実際に評価できる。**
障害発見タイミング	**間接的な一症状ではなく，問題点を直接指摘。そのため，問題点に対する改善を即検討し，実施することが可能。**	**障害，つまりシステムが期待通りの挙動を示さないという実行結果の 1 つに出くわして初めて，原因となる欠陥を究明。原因究明の後にその対応を検討し，実施する。**
潜在障害対応効率	**テストでは検出が遅れる可能性のある欠陥も一度に確認し，問題を顕在化することが可能。問題の顕在化時期が早い場合には，修正が比較的楽であるため，対応が容易。**	**障害を再現させるためのデータや環境を用意する必要あり。また，テストで大きな変更必要事項を発見しても，時間的な問題や作業担当者の心理的な問題で，通常は抜本的な対策は取り難く，暫定対応になりがち。その結果，対応が遅れる。**

これらのことから, 静的または動的コードアナライザ, テスト, カバレッジツール, デバッカなどを利用してバグを摘出するにしても，その前段階でレビューを確実に実施し，開発全体の生産性向上を推進していく必要があります。

1.2.5 ソフトウェア開発とレビューとの関係

レビューを行うとうまくいくのであれば，なぜレビューをきちんと実施せずに問題が発生するプロジェクトが後を絶たないのでしょうか？

レビューを躊躇(ちゅうちょ)する人たちは，レビューをすると工数が取られ，コストが増し，プロジェクトの進捗が遅れるということをよく聞きます。しかし，コストを増し，進捗を遅らせるのは，見落としていたバグによる手戻りによるものです。手戻りは開発工数の 40〜60%を占めると言われています。レビューがプロジェクトを遅らせるのではなく，欠陥が遅らせるという認識が必要です。

レビューが時間の無駄と言えるのは，作業成果物がレビューで発見できる欠陥を含まない場合のみです。逆に言えば，欠陥を摘出しないレビューは無駄と言えます。レビュー手順の確立と維持，チームの訓練，レビュー実施にかかるコストが，手戻りの減少と顧客満足の向上による対応コストを下回れば, 利益が生まれることになります。

ここで必要なことは, 効率的で効果があるレビューとはどのようなものかを認識し，実行することです。例えば，レビューと説明会を混在しないことです。よく見かける事象は，レビューと称して一方的な説明会に留まってしまい，それでレビューを実施したと思い込んでいる人がかなりいるということです。説明会を実施することは必要ですし，場合によっては，そこで重要な欠陥を抽出することもあります。しかし，一般的には多くの欠陥の抽出は望めません。また，レビューは欠陥摘出のためと理解しつつも，実際には指摘事項解決のための議論の場となっていることも多々見受けられます。

このように，一見レビューを実施しているように見えても，欠陥摘出という目的から外れた作業中心になっている場合には，大きな効果は期待できません。

適切なレビューを行えば，先に述べたように効率的な作業を期待することができます。レビューは製品に潜む欠陥のすべてを見つけはしませんが，欠陥摘出のための重要なツールの 1 つとなります。