ソフトウェアテストと導入・移行

PMアソシエイツ株式会社
代表取締役
鈴木 安而 著

ソフトウェアの

トラブルを未然に防ぐために、

必要な知識を徹底習得！

SCC

ソフトウェアテストと導入・移行

ＰＭアソシエイツ株式会社
代表取締役
鈴木 安而 著

・本書で例として使用している「ある家電量販店のシステム」は、「平成 19 年秋期アプリケーションエンジニア試験午後 I 問 2」をもとに作成しました。
・PMBOK®は、米国およびその他の国で登録されている PMI®の登録商標です。
・その他、本書に記載されている会社名、製品名などは、一般に各社の商号、登録商標または商標です。
・本書では TM および®の記載は省略しました。

はじめに

筆者は『PMBOK ガイド』の第 4 版から第 6 版までの翻訳を手掛けて 10 余年、その間プロジェクトマネジメント関係の図書や記事を執筆してきました。あるとき不思議なご縁があって「アジャイルの教科書」を書いてくれと SCC の宮川傳法氏からお話をいただいたのが 2016 年の初めでした。そのときは、すでに欧米ではアジャイルが主流になっていたのですが、日本ではアジャイルへのネガティブな反応しか聞こえてきません。その原因を調べるために、欧米の会議に参加したり、実施している日本の企業に調査に行ったりして情報を収集したのです。そこで得た答えの一つが SCC から刊行された『図解でわかるアジャイル・プロジェクトマネジメント』でした。この書籍は、アジャイルの導入を進めるために、できるだけマネジメントの領域に注力して書いたものです。ところがソフトウェア開発の現場では、相変わらず「デスマーチ」がはびこっていて、品質上の問題は起こすし、成功しても高品質は開発要員の自己犠牲の上に成り立っているような状況です。

ちょうど『PMBOK ガイド第 5 版ソフトウェア拡張版』の翻訳が完了し発行した頃に、再び SCC からお話をいただいたのがこの本の執筆でした。

ソフトウェアのテストに関する教科書ということで、「品質は経営者の責任である」という ISO9000 シリーズの理念に基づき、テスト管理者育成を目的としました。そこで、ソフトウェア・テストのプロセスをシステム開発プロジェクトにおけるサブプロジェクトに位置付けました。

プロジェクトですから、本来は立ち上げから終結までのプロセスがあるのですが、今回は立ち上げを省略して計画から終結までを解説しています。計画ではスケジュールの作成に注力してプレシデンス・ダ

イアグラム法を採用し､かなり具体的で詳細なプロセスとしています。さらに品質尺度（メトリクス）を具体的に数値化し、測定可能にしました。進捗管理はアーンド･バリュー法を採用し､コストとスケジュールの予測を可能にしました。さらにこの本の最大の特徴は、システムの本番移行プロセスを事例に基づいて詳細に解説したことです。この部分は筆者の実体験によるプロセスを基にしていますので、すべてのソフトウェア開発に合致するとは限りませんが、考え方は通用すると思います。実際に自分がテストしたソフトウェア製品が現場でしっかりと稼働し、お客様のビジネスに役立っている様子を見ることは、非常に嬉しく楽しいものです。達成感と充実感にあふれ、次の仕事への意欲が湧き上がってくるものです。

読者の皆様には、ぜひこの本を座右に置かれて、日々のソフトウェア・テストに活かされることを期待しております。

最後に、刊行にあたってお世話になったSCCの宮川傳法（ひさのり）氏、現場でのテストケースについてご協力いただいた、鉄鋼メーカーの山中良文氏、電力会社の高木裕氏、元コンピューターメーカーで現在PMA社の講師でもある川俣賢一氏、その他さまざま形で応援していただいた方々へ感謝の気持ちを捧げます。

平成30年1月　筆者

目次　ソフトウェアのテストと導入・移行

第1章　ソフトウェア品質について

第1章 ソフトウェア品質について

ソフトウェアのテストについて理解を深めるためには、その品質についての定義や意味するところを理解することは非常に大切なことです。

1.1 品質とは

品質は、モノやサービスなどを評価する際の基準として「品質が良い」とか「品質が悪い」というように使われています。そもそも何を基準として良し悪しを決めているのでしょうか。「品質を向上させる」という場合に、具体的には何をどのように向上させればよいのでしょうか。そして、ソフトウェアの品質とは一体何なのでしょうか。これらの問いに答えるために、世の中で定義されている品質について見直してみます。

1.1.1 ISO9000 シリーズにおける定義

ISO（国際標準化機構）は、ISO9000 シリーズの中で品質を次のように定義しています。（日本規格協会）

- **品質**：本来備わっている特性の集まりが、要求事項を満たす程度
- **要求事項**：明示されている、通常、暗黙のうちに了解されている若しくは義務として要求されている、ニーズ又は期待

●**特性**：そのものを識別するための性質

●**品質特性**：要求事項に関連する、製品、プロセス、またはシステムに本来備わっている特性

➢ システムとは、相互に関連する又は相互に作用する要素の集まり。

ISO9000 シリーズは、ソフトウェアだけでなくすべての要素への定義です。ですからソフトウェアの品質について考える場合にはあまりに漠然としているので、さらに具体的な表現とする必要があります。これについては「1.2　ソフトウェア品質とは」で解説します。

1.1.2 基本的な品質用語

まず品質についての基本的な用語について解説します。

① 品質

ISO9000 シリーズにおける品質表現が一般的ですが、品質には主観的な要素が多いという特徴があります。なるべく客観的に表現したいものですが、例えば「いいもの」という場合にも個人差が出てきます。ハサミを例にとってみましょう。1 万円する高級なハサミは、きっと丈夫で切れ味もよさそうです。また 100 円で買ったハサミは、すぐにダメになりそうです。しかし今、ハサミがなくて困っていて一度しか使わないとしたら、どちらを選ぶでしょうか。確かに高級なハサミは「いいもの」ですが、今困っている状況では必要ありません。100 円のハサミも、その人の使用目的に見合っていれば「ちょうど良い品質」といえるのです。

②　等級（グレード）

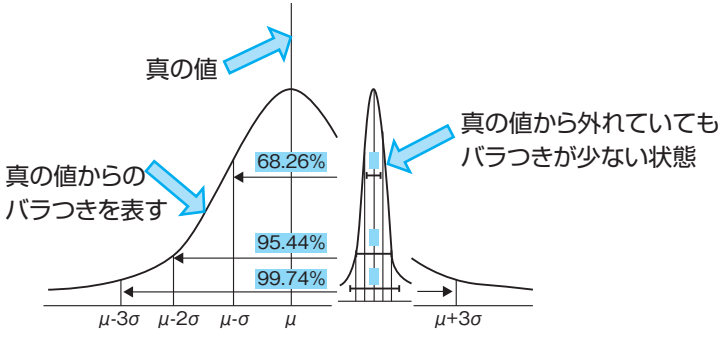
設計上で意図されるものであり、同一の用途であっても技術的特性が異なる成果物に定めた区分のことをいいます。よく「グレード」が高いとか低いとかいいますが、品質とは別に定義される用語です。品質と等級における評価の違いを例にとってみます。

等級と品質	評価	例
等級が低いが品質が良い	○	低機能ソフトウェアでバグが少ない
等級が高いが品質が悪い	×	高機能ソフトウェアでバグが多い
等級と品質の両方が良い	◎	高機能ソフトウェアでバグが少ない
等級と品質の両方が悪い	×	低機能ソフトウェアでバグが多い

製品設計では、投資に基づいたバランスをとることが大切です。

③　正確

品目の測定値が真の特性値に近くなるほど正確といい、真の値からの偏差をバラつきといいます。



「正確なものを作る」という場合には、バラつきの度合いを定義する必要があります。通常、ギリシャ文字のσ（シグマ）でバラつき度合いを表し、次のような確率で表現されます。

σ（シグマ）	達成確率	失敗確率
1	68.26%	31.74%
2	95.44%	4.56%
3	99.74%	0.26%
6	99.9999%	（3.4/100 万）× 100%

達成なのか失敗なのかは、目的によって異なりますので注意が必要です。ちなみに 6 シグマは、100 万回のテストにおけるエラーの発生が 3.4 回以内であることを意味しています。また欠陥ゼロという表現は、理想ではありますが目標を表す用語であって、現実的ではありません。

④　精密

測定単位としてどの程度詳細に表現するのかという意味です。例えば、1cm 単位の目盛りの定規で測定すると、1cm 単位でしか表現できませんが、1mm 単位の定規で測定すれば 1mm 単位で表現できます。これを精密さの度合いともいいます。「精密なものを作る」という場合には、測定単位やツールを決めておかなければなりません。先ほどのバラつきを表した正規分布図で、真値から離れている集団がありますが、真値から離れていてもバラつきが少なければ、その中心値に対して精密であるといいます。

1.1.3 品質管理の考え方

一般に「品質管理」と表現する場合に、狭義と広義の場合があります。狭義の場合には、英語で「QC：Quality Control」と表現されます。QC 活動では、品質をある一定の基準値内に収めるように関連作業に働きかけます。要するにバラつきを最小限にする活動です。広義の場合には、英語で「QM：Quality Management」と表現されるので、

一般に「品質マネジメント」と訳されて、包括的な品質活動を表します。ここでは品質活動を「品質マネジメント」として解説いたします。

①　作業の品質

最終目的は「いいもの」を作ることですが、そのためには「作業」そのものの品質が高くなければなりません。特に人が直接関与する開発作業ではバラつきが大きくなりやすいものです。仕事の進め方をプロセスとして定義して愚直に進めることが大切です。このことを「正しく仕事をする」と表現しています。もし、プロセスに問題があると感じたならば改善を提案し、チームとして改善に取り込まなければなりません。成果物の品質は作業の品質によって左右されるのです。

②　成果物の品質

作業の結果として成果物が作成されることになりますが、納品の前にその成果物の品質を確認する必要があります。これがいわゆる「品質検査」です。成果物の特性によって検査方法はさまざまですが、目標とする品質基準に合格しなければ出荷できません。品質基準を定義する場合には、具体的で測定可能な尺度が必要です。「世界一の品質」といっても測定できなければ意味がありません。ある統計によれば、顧客が品質の目標を具体的に提示したプロジェクトは、提示しなかったプロジェクトより 2 倍ほど品質が良かったといわれています。

③　検査よりも予防

品質は計画されて達成されるべきであり、検査によって達成されるものではありません。なぜならば、検査はある品質基準に基づいた判断がなされて成果物の合格や不合格を決定しますが、だからといって品質が向上するわけではありません。例えば「検査を強化する」ということは、合格の範囲におけるバラつきの幅を狭めるだけですから、

不良在庫の山を作り上げることになります。品質を高めるためには、不良や全体のバラつき状況などを分析して、上流工程の設計や構築プロセスへフィードバックし､改善を求めることが重要です。要するに、開発工程の上流である設計等のプロセスにおいて予防処置を講じることで品質が向上するのです。品質は上流工程で対処しなければ向上しません。上流で作り込んでしまったバグや不具合を「技術的負債」と呼ぶことがあります。自分が作ってしまった負債（不具合）を次工程に持ち越さないことが重要です。この負債は、後になればなるほど大きくなる、ともいわれています。そのために被る損失は計り知れないものです。後述するリコールは、その１つです。

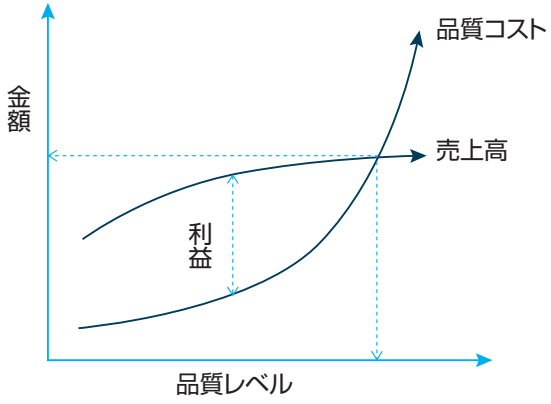
④　金メッキ

顧客の要求以上のものを提供することを「金メッキ」といいます。メッキが剥がれやすいという意味ではなく、顧客にとっては余計なモノを作ることをいいます。確かに同じコストならばいいモノは喜ばれるように考えがちですが、顧客は「そんなことをするならもっと早く安くしてくれればいいのに！」と思ったりします。ですから顧客の要求に見合った品質を提供することが大切です。そのためには、事前に品質要求事項をしっかりと把握する必要があります。

⑤　限界分析

品質基準を達成するためのコスト増分と品質を向上させることによる利益との合致点を探すための分析のことです。芸術の世界では製作者の意思によって納得のいくまで「いいもの」を追求するでしょうが、ビジネスの世界では、投資対効果という状況を考慮して品質基準を決める必要があります。「世界一の品質」という目標を掲げて品質を追求することも悪くはありませんが、そのためにどの程度のコストをかけられるのかについて理解しておく必要があります。ビジネスではコ

ストをかければかけるほど製品価格が上がりますので、売れ行きに影響します。確かに高価でも高品質のものを購入する顧客が存在することは確かですが、自組織が目指す市場にとって何が最適なのかについて戦略的な判断が求められます。



図：限界分析の概念

⑥　品質への責任

品質活動は品質担当者だけが責任を負うものではありません。組織全体が品質に関係する責任を持ちます。プロジェクトでは最終的にプロジェクト・マネジャーにすべての責任がありますが、チーム・メンバーにも仕事上の責任があるのです。自分が担当した作業には責任感を持ってあたらなければなりません。また上級マネジメントは、たとえ現場の問題であったとしても、組織全体としての品質に最終的な責任を負うことになります。品質マネジメントの基本サイクルとして「PDCA」を世に広めたW・エドワーズ・デミング（William Edwards Deming）は、「品質問題の85%はマネジメント上の問題である」と言っています。ちなみに、デミングの意向を踏まえた米国を中心とした考え方が近年変化しています。例えば、

「PDCA（Plan-Do-Check-Action）」⇒「PDSA（Plan-Do-Study-Action）」

へ変更になっています。これは、「Check = 点検」よりも「Study = 分析」として確認作業を強化した考え方です。

1.2 ソフトウェア品質とは

1.2.1 ISO9126 における定義

本書「付録 品質基準決定」ではソフトウェア製品の品質定義として JIS X0129 を採用していますが、別途ソフトウェア製品の評価に関する標準があります。それは JIS X0133 で、JIS X0129 を包含するように規定されました。ソフトウェアの品質特性としては同じ定義なので割愛しますが、他の要素として、利用時の品質特性モデルについて、次のように追加項目を定義しています。

- **有効性**（effectiveness）
- **生産性**（productivity）
- **安全性**（safety）
- **満足性**（satisfaction）

これらの項目は利用者側の目で見たもので、利用者としてオペレーターやプログラマーも含む広い概念になっています。

1.2.2 IEEE (Institute of Electrical and Electronics Engineers)による定義

IEEE（電気電子技術者学会）コンピューター・ソサエティの IEEE 610 では、ソフトウェア品質を次のように定義しています。

- システム、コンポーネント、あるいはプロセスが、特定の要求事項を満たす程度
- システム、コンポーネント、あるいはプロセスが、顧客またはユーザーのニーズや期待を満たす程度

表現に少しの違いがありますが、ISO9000 シリーズにおける定義と大きな違いはありません。また ISO25000 シリーズに基づいて制定された SQuaRE が JIS X0129 および JIS X0133 を包含していますが、本書における品質定義には大きな影響を与えないと判断して割愛します。

1.2.3 ユーザー視点の品質

さまざまな品質の定義について紹介してきましたが、顧客満足を得るためには「顧客の視点」が重要です。開発者が、その視点でどんなにいいことを並べ立てても、顧客は別の視点を持っていることが多いので、満足に至らないケースが増えています。品質の用語で解説したように、等級（グレード）という用語は品質とは別物としていますが、ユーザーにとっては区別がありません。ユーザーから見たソフトウェア品質についての例を挙げてみます。

●安全性

➢ このソフトウェアによって他に危害を及ぼさないか

●セキュリティ

➢ 脆弱性はないか

●信頼性

➢ 安心して使えるか

●**耐故障性**

➢ 問題があっても自己回復できるか

●**ディペンダビリティ**

➢ 他のソフトウェアやハードウェアとの依存関係はどうか

●**スケーラビリティ**

➢ 将来の機能拡張をサポートできるか

●**パフォーマンス**

➢ コスト・パフォーマンスはよいか

●**学習のしやすさ**

➢ 簡単に理解できるか

●**ユーザビリティ**

➢ 使いやすいか

●**エラー・メッセージの解読のしやすさ**

➢ 出されるメッセージはわかりやすいか

●**アベイラビリティ**

➢ 必要なときに簡単に使えるのか

●**アクセス可能性**

➢ 簡単にアクセスできるか

●**効率性**

➢　レスポンスは早いか

●**相互運用性**

➢　複数のシステム間で使えるか

●**堅牢性**

➢　いろいろな使い方をしても大丈夫か

これらの特性の多くはユーザーにとって「当たり前の品質」であって、要求事項の中に特定されることが少ないものです。要するに暗黙知であり、また測定しにくい内容なので、仕様書の中に明記されないことが多いということです。要求を聞き出す場合にも、専門用語ではなくユーザーの使う言葉で表現し理解するように努力する必要があります。

1.2.4　品質評価との関係

品質の評価は、通常、品質要求事項に基づいた品質尺度によって測定された結果からなされるので、顧客の暗黙知は評価から外れてしまいます。ですから品質評価基準を設定する場合には、ISO などの基準だけにとらわれることなく、ユーザーの真の要求あるいは隠れた要求を引き出す努力をしなければなりません。

人間というものは、自分の考えを明確に形式化することが不得手なものです。ですから、ユーザーの頭の中にある「思い（暗黙知）」を引き出して文書化することが大切なのです。顧客が作った要件定義を鵜呑みにすると思わぬトラブルに遭遇することが多いものです。その裏に潜んだ本音を理解することが重要です。

1.3 品質に関するコスト

品質目標を達成するためにはそれ相応の予算が必要となります。決して精神力だけで達成できるものではありません。品質に関するコストには、プロジェクト期間中に発生するものとプロジェクト後に発生するものがあります。また品質基準に適合させるための「適合のコスト」と基準を外れた場合に必要となる「不適合のコスト」があります。それについて解説します。

1.3.1 適合のコスト

●**予防コスト**（高品質製品の構築）

① トレーニング

➢ 作業員のスキルアップによってミスを減らすために、全員を最低限必要なスキル・レベルに引き上げます。

② プロセスの文書化

➢ 工程を文書化してミスを減らすために、標準化やテンプレートを準備します。

③ 装置

➢ 性能のよい装置を使いバラつきを減らすために、メンテナンスを十分に行い、場合によっては新機種にリプレースします。

④ 正しく作業をするための時間

➢ 無理な作業時間でなく、確実に実行できるスケジュール

として作業員の健康を保ち、モチベーションの向上を図ります。

●評価コスト（品質査定）

① 試験

➢ 十分なテスト時間を確保することによって品質が向上し、かえって総コストの削減に寄与できます。

② 破壊試験損失

➢ 破壊試験のためのコストを設定する。
ソフトウェアの場合にはあまり見られませんが、耐久テストなどによる損失を見込みます。

③ 検査

➢ 検査のためのコストを設定する。
第三者検査のために外部検査官を招聘することがあります。

1.3.2 不適合のコスト

●内部不良コスト（プロジェクトで発見された不良）

① リワーク

➢ デバッグや再作業のためのコストを事前に見積もることは困難ですが、経験則から予備費を確保します。

② 廃棄

➢ 成果物を廃棄することによって被る損失が発生することがあります。ソフトウェアでもせっかく完成したにもかかわらず顧客の受け入れがなされずに廃棄することがあります。

●外部不良コスト（納品後に顧客で発見された不良）

① 責任

➢ 契約上の責務にかかるコストには、ペナルティや損害賠償などがあります。

② 保証

➢ 保証のためにかかるコストには、保証期間中に発生する障害対応コストなどがあります。

③ ビジネス機会損失

➢ 評判を落とすことによって被る損失は、具体的な金額とはなりませんが長期的には大きな損失となります。

➢ その他、リコールなどによって被る損失などがあります。

これらのコストのうち、適合のコストと不適合のコストのうちの内部不良コストはプロジェクト予算でまかなうことになりますが、外部不良コストはプロジェクト終結後、つまり納品後に発生するコストなので、別途、会社として引当金を用意する必要があります。また、リコールなどは多大な損失になるケースが多く、場合によっては会社の存続にかかわることになりかねません。ですから適合のコストのための予算を十分に確保して、いかに外部不良コストを最小化するのかに

ついて考慮しなければなりません。開発コストを抑えることはビジネスの要求として当然のことですが、行き過ぎた削減は外部不良コストの増大につながります。

品質マネジメントでよく使われる方程式に、次のものがあります。

予防コスト＜トラブル対応コスト

実際には「100 万円かけたら、いくらバグがなくなるか」という問いには具体的な数値で答えられませんが、過去の経験や統計的な分析で証明されていることです。

1.4 低品質の影響

品質が低いということは期待された品質を達成できていないということなので、次のような、さまざま問題を引き起こします。

1.4.1 リコール

品質コストで述べたように、リコールは外部不良コストの中でも最大の影響を企業に及ぼします。補償のためのコスト以外にも法律上の制裁を受けたり、評判を落とすことから売り上げの減少を招いたり、会社の存続にかかわる問題です。リコール対象となる製品は製造物責任法（PL 法）によって定義されています。ソフトウェア製品は、ハードウェアに組み込まれた、いわゆる組み込みソフトが対象となります。一般的なソフトウェア・パッケージは対象となりません。つまり、組み込みソフトには相応の品質が求められることになりますので、品質管理のために十分な投資をしなければなりません。

PL 法の条文を紹介します。

●第１条　目的

製造物の欠陥により人の生命、身体又は財産に係る被害が生じた場合における製造業者等の損害賠償の責任について定めることにより、被害者の保護を図り、もって国民生活の安定向上と国民経済の健全な発展に寄与する

●第２条　第１項

製造物とは製造又は加工された動産をいう

●第２条　第２項

欠陥とは当該製造物の特性、その通常予見される使用形態、その製造業者等が当該製造物を引き渡した時期その他の当該製造物に係る事情を考慮して、当該製造物が通常有すべき安全性を欠いていることをいう

法律の表現はなかなか馴染みにくいものですが、この条文は知っておくとよいでしょう。またテストに関連する法律には民法があって、そこではいわゆる保証期間に関する条文があります。(2017年5月改正)

1.4.2 企業のイメージダウン

たとえ PL 法対象でなくても、不具合が多いソフトウェアを使うことによるシステムダウンなどによって、会社としての信頼を失うことがあります。システム開発に続いて保守というビジネスがあり、その先にはシステム改訂プロジェクトや新規システム構築プロジェクトなどのビジネスが待ち受けていることが多いのですが、信頼を失ったま

ますとその顧客におけるビジネスを継続することが困難になります。それどころかいったん貼られたレッテルを剥がすためには多大なエネルギーを必要とします。

1.5 無駄を省く

1.5.1 リーンという考え方

リーンという用語は無駄を省くという意味に使われていますが、そもそもメアリー・ポッペンディック（Mary Poppendieck）らが提唱している手法で、トヨタ生産方式を手本に、ソフトウェア開発を成功させるための原則を基にして具体的な実務慣行を生み出しています。その考え方の基本として「作業工程には７つの無駄がある」とされています。それを次に紹介します。

① 作りすぎ:ソフトウェア開発では、スコープ･クリープといって計画にはない要素を作り出してしまうことです。

② 手待ち：ソフトウェア開発では、スケジュール上、作業を始めなければいけないのに前工程からのアウトプットが出てこないので待っている状態のことです。

③ 運搬：ソフトウェア開発では、出来上がったソフトウェア・モジュールを一定の場所に保管しなければならないのに、余計な時間がかかってしまうことなどです。

④ 加工：ソフトウェア開発では、変更による手戻りや追加機能に相当します。

⑤ 在庫：ソフトウェア開発では、せっかく作成したソフトウェア・モジュールが使われないでPCの中に眠っている状態などです。

⑥ 動作：ソフトウェア開発では、ソフトウェア・モジュールの動作の無駄な動きや冗長性が相当します。

⑦ 不良：ソフトウェア開発では、ソフトウェア・モジュールそのものに含まれたバグの存在が相当します。

これらの無駄をいかに省くか、という視点でのプロセス設計が行われなければなりません。この考え方は、昨今のアジャイル型プロジェクトによく採用されていますが、あらゆる仕事に適用することができます。今までやってきたから改善は不要、などと考えていると無駄を省くことができないばかりか、ビジネス・チャンスを失うことにもなりかねません。

1.5.2 カンバン方式による工程の改善

この方式は、デビッド・J・アンダーソン（David J. Anderson）によってトヨタの工程管理研究から考え出されたものです。トヨタの大野耐一氏が進めた現場主義による工程管理から、米国式のトップダウン型マネジメント方式に変換したワークフロー改善手法です。現場における可視化のために看板を活用し、一定のペースに基づく開発期間を設定し、コストを考慮した優先順位付けによって、仕掛品（WIP: Work-In-Progress）の最小化を図ります。

：タスクを記述した付箋紙

ユーザーストーリー	作業待	作業中	テスト中	完了
1				
2				
3				
4				

作業の進捗に合わせて付箋紙を移動していく

この方式は報告の簡素化も達成できますし、全員が状況を現場で簡単に把握できるので動機付けや意識付けに役立ちます。よく、状況を示した文書をサーバーに保管しておいて後で確認するというプロセスを構築することがありますが、サーバーに保管された情報は、多忙な作業員には遠い存在になってしまうことがあります。現場のための情報は身近に置かないと役に立ちません。しかしながら、ビジネス上必須の情報は、正式のものとして作成し保管されなければなりません。そのための情報管理と現場での情報管理とを分けて考えると管理が容易になります。

第 1 章の理解度テスト

1．品質活動について正しく表現しているものは次のうちどれか

a）品質活動とは顧客要求に正しく応えることであり、顧客によって文書化された機能要件を忠実に達成することである。

b）品質活動とは欠陥のない製品を作ることであり、ISO などに定義された品質基準を達成するように活動することである。

c）品質活動とは顧客にとって価値のある製品を作ることであり、その価値をよく理解し顧客要求を達成することである。

d）品質活動とは継続的な改善を実行することであり、最高レベルの品質を達成するように努力することである。

2．デミングサイクルを正しく述べているものは次のうちどれか

a）アジャイル型プロジェクトを進めるための開発アプローチの１つである。

b）Plan − Do − Check − Action の改善サイクルの別名である。

c）80対20の法則を改善活動に応用したアプローチの１つである。

d）欠陥ゼロを目標として改善活動を繰り返すことである。

3．品質と等級は異なる特性である。次のうち等級を表しているものはどれか

a）シックスシグマを目標として活動しているが、まだ３シグマ状態であるという程度を表している

b）製品のデザインをトップデザイナーに依頼して、若い人にうけるようにした。

c）テスト中にバグが摘出できなくなってきたので、テスト方法を変えてさらにバグを摘出した。

d）顧客の品質要求があやふやだったので、具体例を示して数値化した。

4．品質が向上すると全体のコスト削減に寄与するといわれている。その理由は何か

a）品質が良くなると不適合のためのコストが少なくなるから

b）品質が良くなると適合のためのコストが少なくなるから

c）品質が良くなると開発者の動機付けに役立つから

d）品質が良くなるとテスト関係者の動機付けに役立つから