SEC BOOKS

プロセス改善ナビゲーションガイド

～自律改善編～

SPINA^{3}CH（スピナッチキューブ）で進める自律改善

独立行政法人 **情報処理推進機構**　技術本部　ソフトウェア・エンジニアリング・センター

はじめに

ソフトウェアは、「人」により、「人」のために作られるものですから、ソフトウェアの開発環境を改善するためには、「人」の側面に目を向ける必要があることは自明でしょう。開発環境を改善するにあたり、「人」以外の「構成要素」そのもの（作るもの、方法論、プロセス、ツール等）を個別に扱うのではなく、それらに対して、「人」がどのように関わるのかの視点が必要になります。即ち、「人」が関わるがゆえに、開発環境は自由度の高い運用がなされ、改善のための解は唯一無二ということにはなりません。さらに重要なことは、「人」が関係する場合、「他者からの指示・指摘に基づくよりも、自らの気づきによる方がより効果的に実践できる」ということがあるのではないでしょうか。

SPINA^{3}CH（スピナッチキューブ）自律改善メソッドは、以上で述べたことを可能な限り自然な形で実践できることをめざして提供するものです。

2013 年 3 月　プロセス改善 WG

SPINA^{3}CH = Software Process Improvement with Navigation, Awareness, Analysis and Autonomy for CHallenge

Navigation: ソフトウェアエンジニアリングの適用と改善の進め方にガイドがある
Awareness: 気づきから始める
Analysis: 状況と課題を分析して進める
Autonomy: 自律的に改善を進める
Challenge: 業務上のチャレンジ意識が重要

目　次

商標等の取り扱いについて

- CMM®、CMMI® はアメリカ合衆国特許商標庁に登録されています。
- Microsoft、Excel は米国 Microsoft Corporation の米国およびその他の国における登録商標です。
- IBM は、世界の多くの国で登録された International Business Machines Corporation の商標です。
- SPEAK は、IPA（独立行政法人情報処理推進機構）の登録商標です。

- 上記にかかわらず、本ガイドに掲載されているシステム名、製品名等は、一般にその開発元の商標または登録商標です。本ガイドでは、本ガイドを製作する目的でのみそれら商品名、団体名を記載しており、編集者としては、その商標権を侵害する意思、目的のないことを申し述べておきます。

第１章　解説編

1．SPINA^{3}CH 自律改善メソッドとは

本手法は、ソフトウェア開発に関わる「仕事のやり方」を、現実に即して改善・向上させるチャレンジのためのツールを提供するものです。その「仕事のやり方」のことを、ここでは「業務（プロセス）」といいます。仕事の内容として行うべき業務（プロセス）の目的と、業務（プロセス）を通じて実現すべき成果をはっきりさせ、そのために必要な実現方法を着実に実施していくことを意味しています。

このツールには、以下のようないくつかの利用の場面が考えられます。一人でも取り組めることが特色の１つです。

- 開発チームの取り組みとして
- エンジニア個人の自己トレーニングや自己チェックとして
- 企業総体の改善ツールの１つとして

このツールでは、取り組まれる方々の自主的な努力と、手と頭を動かす作業に期待をしています。しかし、そうした現場の改善活動にも、先人の知恵や遺産といった手掛かりが必要です。そのために少しばかりのヒントをツール（道具）が提供するようになっています。また、改善のアイデアを出すだけでなく、実際の仕事に適用することを強く推奨する仕組みとなっています。

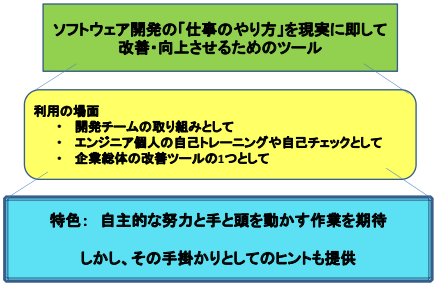


図 1-1　SPINA^{3}CH 自律改善メソッド

2.　本書の目的

本書の対象読者は次の皆さんです。

- 自分たちの開発をよりよくしたい人：たとえばプロジェクトのメンバとリーダ
- ソフトウェア開発の問題を何とかしたい人：たとえばプロセス改善の推進者

また、内容と使い方については、次のことを想定しています。

- SPINA^{3}CH 自律改善メソッドの利用ガイド

  われわれが想定した利用方法を提示します。それぞれの道具の詳細は、第 2 章で詳しく紹介します。
- 実際の状況を踏まえて、工夫して自分に合うものにしてもらう

  シート類の内容も含め、ここで提供するのは一種のテンプレートです。現場のニーズと問題意識に合わせて変更してもかまいません。ただし、そうした工夫をする際に、局所的な経験や思い付きに頼って自己満足に陥らないように、様々な文献やプロセス診断・改善の国際規格（ISO/IEC 15504、33000 シリーズ）や CMMI®[1]、SPEAK-IPA[2] のようなプロセス診断・改善モデルで示されている知識を参照・活用する姿勢が重要です。

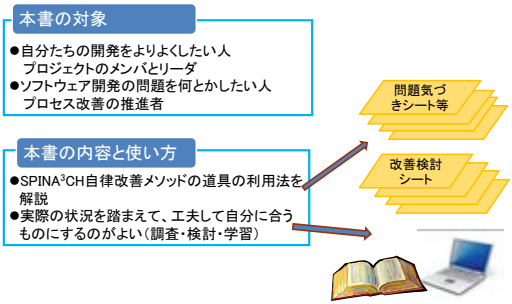


図 1-2　本書の目的

1　CMMI®：能力成熟度モデル統合、Capability Maturity Model Integration の略称。米国カーネギーメロン大学で開発され、現在、世界で最も広く用いられているプロセスアセスメントモデルの 1 つ。

2　SPEAK -IPA（Software Process Evaluation & Assessment Kit IPA）：ソフトウェアプロセスの供給者能力判定及びアセスメントキット－ IPA 版。アセスメントモデルの国際規格（ISO/IEC 15504）に準拠したアセスメントモデルとアセスメント手法。

3. 「仕事のやり方をよりよくする」ための道筋

3.1 基本的な留意事項

仕事のやり方をよりよくする上で、基本的な留意事項があります。

(1) プロフェッショナルな仕事のスタイルの追求

誇りを持てるプロフェッショナルな仕事のスタイルを求める必要があります。ビジネス的に言えば、他社に引けを取らない、他社より優れた成果や仕事のスタイルです。

(2) 技術とノウハウの蓄積と効果的な利用

この対応によって、プロの仕事が有効に行えます。所属する組織やチームの総合的な力量の確実な手掛かりとして、こうした蓄積が役立ちます。これらが顧客やエンドユーザに対する保証の手掛かりとなる場合もあります。

(3) 作業環境に適したテーラリング（修整）

実務での仕事のやり方は、プロジェクトの環境や規模の大小などの要素を見極めて、それらに適したものにする必要があります。ソフトウェア開発は、毎回決まりきった手順を適用すればＯＫというわけにはいかない面があります。プロジェクトの環境、規模の大小などの要素に応じた柔軟性を持ち、いわゆるテーラリング（修整）が的確に行える必要があります。このためにも技術やノウハウの蓄積が必要です。

3.2 推進上のポイント

次に、その「改善」推進上のポイントを強調しておきます。

(1) 事実に即して現状を多面的に把握する

記録などに残された事実情報を収集する、あるいは先輩・同僚・上司などの意見を聞くなど、現状をありのままに把握することにより、現状の問題をいろいろ

な観点から多面的に見ることが必要です。

(2) 自分で考える

現場の問題は、その状況に即して考えるのが一番です。

(3) 自らできる改善を行う

他人任せ、他人への責任転嫁では改善は永久にできません。改善の主体は自分です。

(4) 内外から学ぶ、調整する

現状のまま「考える」だけでは行き詰ってしまいます。解決したい問題や課題を解決するための IT の技術やソフトウェアエンジニアリングの技術を学ぶこと、あるいはこれまでに組織内で蓄積されたノウハウを整理すること、場合によっては、周囲の人々を説得したり、協力してもらったりすることも必要です。

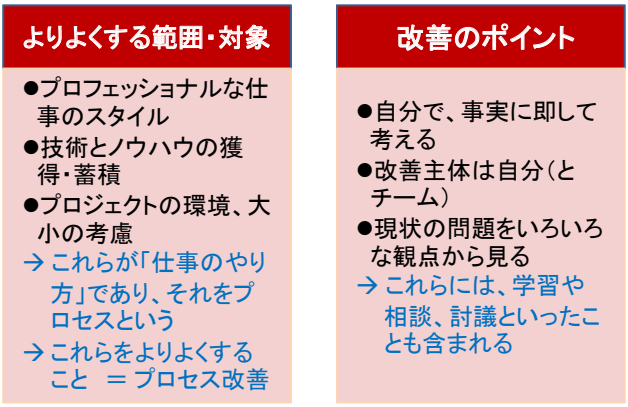


図 1-3　仕事のやり方をよりよくする

コラム　家計の改善にたとえると…

プロセス改善の一般的な議論では、枠組みとなるアプローチの議論しかしていません。実際の改善では、具体的な議論が必要です。それはちょうど、家計の状態を改善するために家計簿をつける際、帳面のつじつま合わせではだめで、「どの支出を節約するか」、「次に買う自家用車はどれを選ぶのがよいか」、「子供のアルバイトはどうするか」など、具体的な議論が必要であることと似ています。

4.　自律改善で用いる道具

SPINA^{3}CH 自律改善メソッドでは次のような段取りと道具を用います。

（1）開発の現場の「事実」と問題への「気づき」をとらえる

「問題気づきシート」を用いて自分の周囲の具体例に即して、問題状況をできるだけ幅広くとらえてみます。

（2）注目するポイントを見出す

抽出した問題の状況をよく見て因果関係を明確化し、本来の原因、重要な問題点、改善するとすればどこに注目するのが良いかといった点から改善候補を絞り込んでいきます。

（3）現実の業務（プロセス）の良いところと悪いところを考える

注目する業務（プロセス）について、現在、どのように取り組んでいるのかを冷静に書き留めてみます。同じ内容が、立場によっては良い点にも悪い点にもなり得ることにも注意が必要です。

（4）問題の原因や改善点を把握する

現状の仕事のやり方の改善すべき点を、絞り込みの結果（注目するポイント）も踏まえて、抽出していきます。

（5）良い事例や他の組織と比較する

改善方策を考えるにあたり、実際に行われてきた良い事例（「ベストプラクティス」と言ったりします）や、他の組織での実施例（ソリューション例）の情報を参考にするのもよいでしょう。

(6) ソフトウェア開発技法やマネジメント技法を学習する

また、理論的な研究、技法、先進的な事例（これもソリューション例）などを参考にするのがより効果的です。

(7) ツールなどを調査・評価する

ソリューション例の中には、ソフトウェア支援ツールや支援環境などを積極的に利用するものもあります。それらの有効性の評価もできるとよいでしょう。

(8) 改善計画としてまとめる

改善案を実際の改善計画としてとりまとめ、関係者で共有することが必要です。

(9) 知識の蓄積と整理を行う

作業に用いたシート類の記入内容は、業務（プロセス）の現状と改善着眼点を簡潔にまとめた貴重な情報です。これらの情報を蓄積・利用することも重要です。

SPINA^{3}CH自律改善メソッドで用いる段取りと道具

- 開発の現場の「事実」と問題への**「気づき」**をとらえる
- **注目するポイント**を見出す
- 現実の業務（プロセス）の良いところと悪いところを考える
- **問題の原因や改善点**を把握する
- 良い事例や他の組織と**比較**する
- ソフトウェア開発技法やマネジメント技法を**学習**する
- ツールなどを**調査・評価**する
- よく考え、その結果を**改善計画**としてまとめる
- 知識の蓄積と整理を行う

図 1-4　自律改善で用いる道具

5.　実際の改善への踏み出し

SPINA^{3}CH 自律改善メソッドでは、改善案を出すだけでなく、実際の改善に踏み出すことを重要視しています。

そのために、チームや組織、場合によっては部門長などの上司の合意を取り付けることも必要です。合意を得るには、事実に基づく現状の共有や理由づけ、世間相場などの調査も必要になるかもしれません。

あまり拙速な改善計画は、実現性が低く、マイナスの影響を残すだけに終わるかもしれません。着実な成果を継続して積み上げる等、長期的な見通しを持ちたいものです。パイロットプロジェクトを選び、小規模に始め、逐次改良して他部署に展開していくという考え方もあります。

また、具体的な改善を遂行するために、めざす成果を達成するには現在の業務（プロセス）のどの部分をどのように変えるのかを明確にし、そのために必要なリソース（ハードウェア･ソフトウェアの設備環境や要員／稼働時間）を見積もり、業務（プロセス）の変更手順や時期も明確にしていく必要があります。

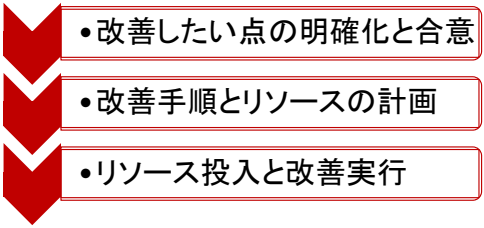


図 1-5　改善への踏み出し

6.　持続的な改善

6.1　改善のサイクル

改善のまわし方についてもいろいろな考え方や、現場の状況に即した実施方法があります。

身近な改善を積み重ねていく方針の場合には、数ヶ月のサイクルで、小さくても具体的な改善点を目に見えるものにしていくと効果的でしょう。それに対して、部門全体の制度的なやり方を変えるような場合や、新しい技術的手段を導入するような場合には､1～2年といったサイクルを想定するのが現実的かもしれません。

このような見極めには経験的な知識が役に立ちますが、サイクルが長期的になる場合にも、数ヶ月単位の節目で改善効果を確認して、以降の対応をより効果的にするなど、確実に成果を獲得するための工夫が必要です。

6.2 改善結果の評価

改善の結果は、明示的に「評価」したいものです。改善は必ずしも、ビジネス的な好業績に直結しているとは限りません。じわじわと能力を上げていく改善もあります。また、新しい手法の導入のために、一時的に直接のパフォーマンスが下がることもよく見られます。

こうした諸要素を踏まえなければいけませんが、改善は客観的な結果を求めて行うべきです。もし、一見成果が上がっていないように見える局面に直面したら、「なぜそうなのか」をよく考えましょう。それは、新しい改善への素晴らしい出発点となるかもしれません。もし、まったく新しい手法や状況にチャレンジしているのなら、小刻みに中間評価をしながら実施方法を修正していくのも、よい考え方かもしれません。

図 1-6 持続的な改善

7.　SPEAK-IPA や CMMI® との関係

SPINA^{3}CH 自律改善メソッドは、SPEAK-IPA やカーネギーメロン大学の CMMI® などのアセスメントモデルとの関係を意識して構築しました。

SPEAK-IPA や CMMI® などを活用するモデルベースのアプローチでは、ライフサイクル全体への包括的なアプローチが期待されます。一方、SPINA^{3}CH 自律改善メソッドでは、欲しい成果を獲得するために効果がある領域はどこかを特定し、適切な改善手段によるピンポイントのアプローチを行います。つまり、欲しい成果を獲得するための有効な改善領域に“フォーカス”します。

そして、適切な改善手段を検討する際には、アセスメントモデルと連携した道具（改善検討ワークシート）を活用します。アセスメントモデルが持つ多面的な視点を活用して、本質的な「抜け」を補うことが、欲しい成果の獲得に結びつく改善手段となることをめざしています。

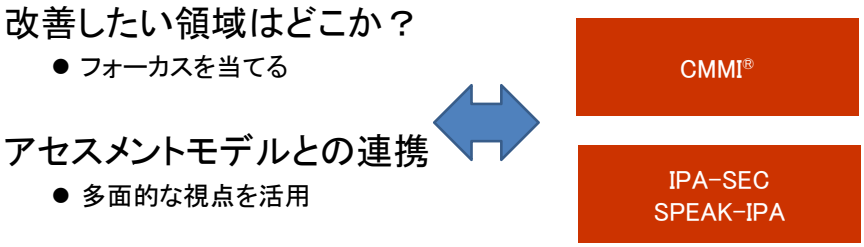


図 1-7　SPEAK-IPA や CMMI® との関係

8. SPINA³CH 自律改善メソッドの特色

8.1 課題ベース改善とモデルベース改善

SPINA³CH 自律改善メソッドの特色を大づかみに言えば、「課題ベース改善」と「モデルベース改善」との融合をめざすものと言えます。

課題ベース改善とは、仕事の遂行上の具体的なトラブルや困った事柄に着目することから始めて、そうした事態が発生する仕事のやり方の改善をひとつひとつ求めるもの、あるいは、現在よりも高い目標を達成するために取り組むものです。

それに対してモデルベース改善とは、着眼点を網羅的に列挙したプロセスモデルやプロセス能力レベルのモデル内容と比較して、仕事のやり方の上で不足しているところや十分でないところを是正していこうとするアプローチです。

	課題ベース	モデルベース
陥りやすい問題	● 火がついた課題に着目するあまり、やるべき事に網羅性がない ● 将来の課題につながる手当てができにくいため、常に火消しの状態である	● 一度に多量のプラクティスを示されるため、現場に敬遠される ● レベル到達ありきになってしまう ● モデルを実装してしまいがちで、現場には負担になりやすい ● 推進側（SEPG）が進めてしまい、現場不在に陥りやすい
メリット	● 改善の初心者も入りやすい ● 改善の効果が即体感できる	● ソフトウェア開発としてやるべきこととして網羅的に仕組みの定義ができる

図 1-8-1　モデルベース改善と課題ベース改善の融合

8.2　様々な事実情報を扱う

SPINA^{3}CH 自律改善メソッドのスタートラインは課題ベース改善のスタイルをとっています。図 1-8-2 にあるような様々な問題や課題について、自分の周囲の具体的な問題状況を思い起こし、それを裏付ける事実情報を明らかにすることから始めます。

この部分はモデルベースのアプローチではありませんので、「モデル」の考え方に当てはめて「～ができていない」という課題発掘をするのではなく、実際の仕事の上で、チーム内、顧客との関係、エンドユーザとの関係、組織内部の関係や状況などに内在する問題や課題などを、ツールを使いながら具体的な事実情報により明らかにしていきます。

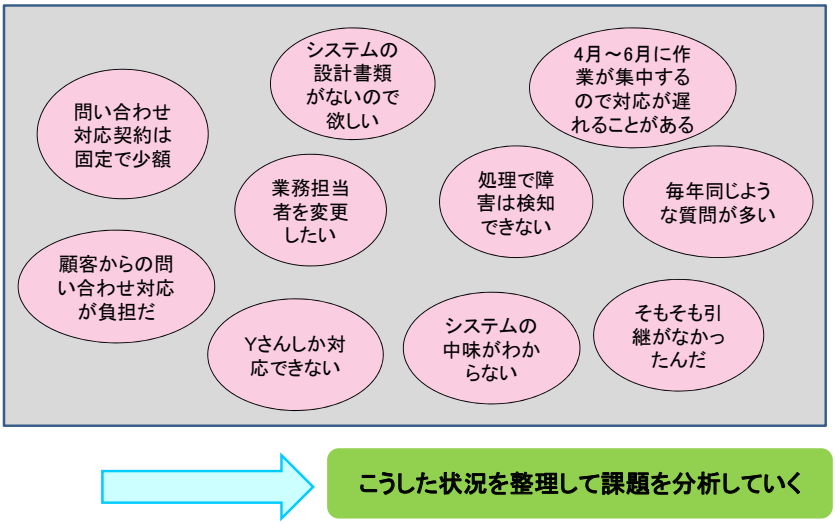


図 1-8-2　様々な状況・問題・要望（ある組織の例）

8.3 「問題気づきシート」の使用

SPINA³CH 自律改善メソッドは、開発当事者ばかりでなく、発注者やシニアマネージャなどいろいろな立場の人たちが行うべき仕事のスタイルや、期待する事項と現状との差異・ギャップを取り上げていきます。

メソッドの最初のステップとして、開発現場にフォーカスした「問題気づきシート」を活用します。「問題気づきシート」とは、図 1-8-3 のように、開発現場の仕事のワークフローを大づかみにしたものです。ここから、自分たちのリアルな問題点を引き出し、分析・整理していきます。

（「問題気づきシート」は第 4 章を参照してください）

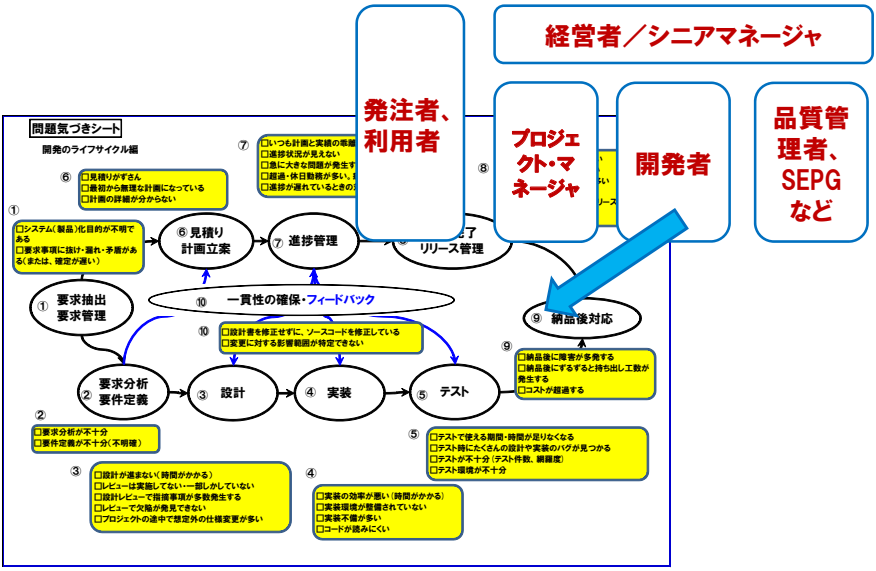


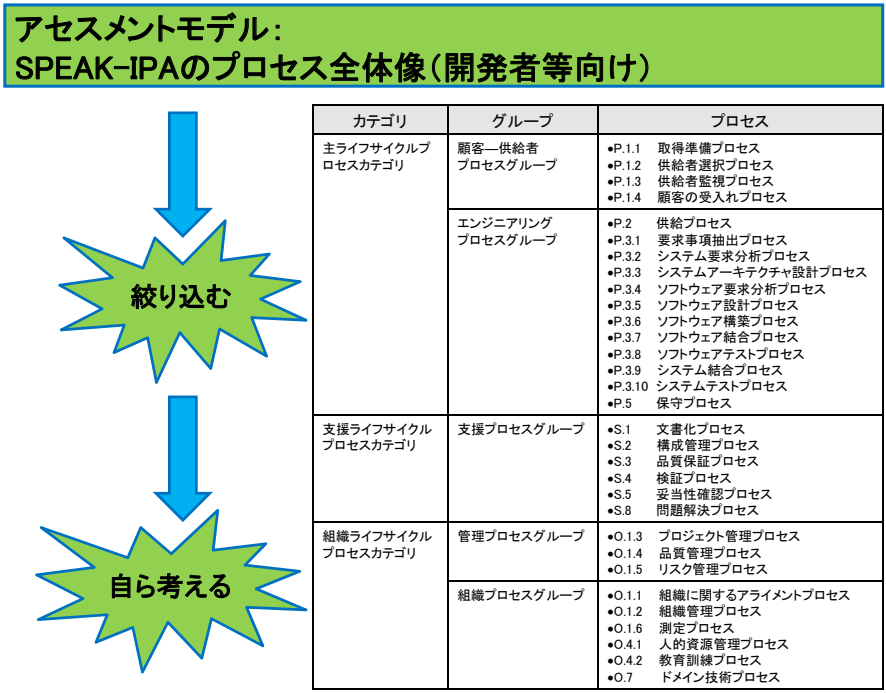
図 1-8-3　問題気づきシート

8.4　改善テーマを絞り込む

関係者はそれぞれの立場で、多くの問題点を個別に抱えているでしょう。それぞれの関係者が持つ個別の問題を幅広く収集した上で、改善すべき問題の絞り込みに入ります。

図 1-8-4 は、アセスメントモデル SPEAK-IPA のプロセスの全体像を示しています。このどこかのプロセスに個別の問題が含まれていると思われますが、これらを全方位で改善するのではなく、問題の因果関係などを考慮しながら、取り扱い可能で、より効果的に改善を行うことができるテーマに絞り込んでいきます。

ここでは、自分たちの現在状況や背景から、重点となる着目点・改善点を特定していくことが重要になります。



カテゴリ	グループ	プロセス
主ライフサイクルプロセスカテゴリ	顧客―供給者プロセスグループ	•P.1.1　取得準備プロセス •P.1.2　供給者選択プロセス •P.1.3　供給者監視プロセス •P.1.4　顧客の受入れプロセス
	エンジニアリングプロセスグループ	•P.2　供給プロセス •P.3.1　要求事項抽出プロセス •P.3.2　システム要求分析プロセス •P.3.3　システムアーキテクチャ設計プロセス •P.3.4　ソフトウェア要求分析プロセス •P.3.5　ソフトウェア設計プロセス •P.3.6　ソフトウェア構築プロセス •P.3.7　ソフトウェア結合プロセス •P.3.8　ソフトウェアテストプロセス •P.3.9　システム結合プロセス •P.3.10　システムテストプロセス •P.5　保守プロセス
支援ライフサイクルプロセスカテゴリ	支援プロセスグループ	•S.1　文書化プロセス •S.2　構成管理プロセス •S.3　品質保証プロセス •S.4　検証プロセス •S.5　妥当性確認プロセス •S.8　問題解決プロセス
組織ライフサイクルプロセスカテゴリ	管理プロセスグループ	•O.1.3　プロジェクト管理プロセス •O.1.4　品質管理プロセス •O.1.5　リスク管理プロセス
	組織プロセスグループ	•O.1.1　組織に関するアライメントプロセス •O.1.2　組織管理プロセス •O.1.6　測定プロセス •O.4.1　人的資源管理プロセス •O.4.2　教育訓練プロセス •O.7　ドメイン技術プロセス

図 1-8-4　SPEAK-IPA のプロセスの全体像

9. SPINA³CH 自律改善メソッドの道具の構成

SPINA³CH 自律改善メソッドによるプロセス改善は、2 つのフェーズに分けられます。第 1 フェーズは問題の気づき（引き出し）と、問題の絞り込みからなる部分、第 2 フェーズは改善検討シートで知識を整理し、アイデアを練り、改善プランに結びつけていく部分です。作業用のシート（フォーム）は、次のものを使います。

①問題気づきシート
②問題分析絞り込みシート
③改善検討ワークシート（記入シート）
④改善検討ワークシート（「ヒント集」）

④は、③を作成していくためのヒントを提供する技術情報を記載したシートです。（これらのシートは、第 4 章を参照してください）

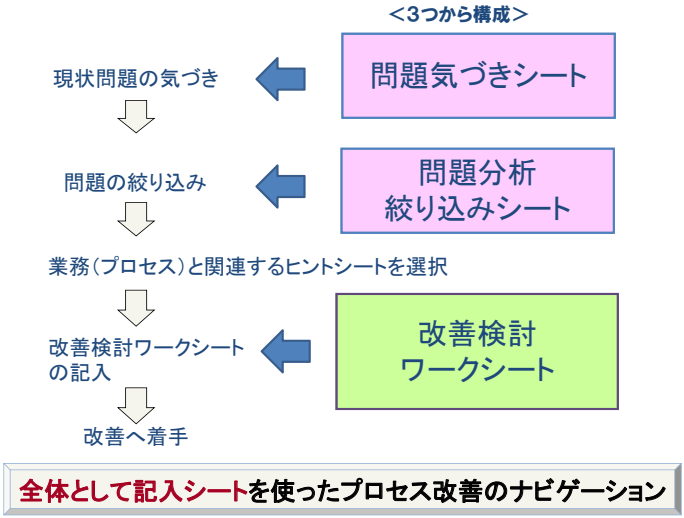


図 1-9　自律改善メソッドの 3 つのシート

第 2 章　実施手順編

ここから先は、SPINA^{3}CH 自律改善メソッドの使い方について解説していきます。

1.　どうやって使うか？

1.1　８ステップの構成

SPINA^{3}CH 自律改善メソッドは、８ステップで構成されています。

STEP1 ～ STEP5 で、問題の気づきから、分析、改善策の検討までを実施します。そして、次の STEP6 と STEP7 で、STEP1 から STEP5 までに気がついた問題の真因の深掘りや改善案に基づいて、仕事の仕方を変えていきます。最後の STEP8 で、STEP1 から STEP7 までの活動内容や結果をふり返り、次の新たな改善のサイクルへとつなげます。

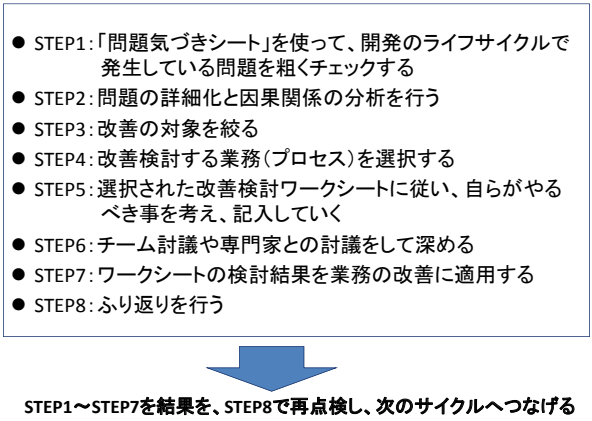


図 2-1-1　SPINA^{3}CH 自律改善メソッドの８ステップ

1.2　各ステップで使用する道具

図 2-1-2 は、8 つの各ステップで使う道具との関連を示しています。

STEP1 では、「問題気づきシート」「問題詳細化ヒント」「問題点カード」を使って、プロジェクトで起こっている問題を抽出します。“いったい、どないなってん！”ということです。

STEP2 で、「問題分析絞り込みシート」を使って、抽出された問題の因果を探っていきます。そして STEP3 では、STEP2 で分析した因果関係をもとに問題の真因を絞り込み、改善の領域を特定します。“どこがあかんねんやろ？”です。

STEP4 で、明確になった改善の領域を「改善検討業務選択シート」に照らして、業務（プロセス）を選択します。STEP5 では、選択した「ヒントシート」の内容に沿って、問題の改善方法を検討していきます。その際、「ヒント集」の事例を参考にしながら、「改善検討ワークシート（記入シート）」の項目を検討していきます。“どないしたらええんやろ～”です。

「改善検討ワークシート（記入シート）」を記入し終わったら、STEP6 で有識者を交えてブラシュアップしていきます。必要に応じて「改善計画」を作成します。もちろん、内容によっては「改善検討ワークシート（記入シート）」の中に計画を記述してもかまいません。“これでいこか”ということをプロジェクトのメンバや関係者と合意してください。そして、STEP7 で現在の作業の内容を改善し、実際に適用してみます。

最後の STEP8 では、改善の内容を“これでよかったんかなあ？”とふり返ってください。期待した成果が出ていない場合には「どこか悪かったのか？」「何がまずかったか？」、うまくいった場合には「次はどこをやろうか」と考えて、再び STEP1 ～ STEP7 を繰り返してください。改善活動は、自立的に、かつ自律的に、継続的に実施していくことが大切です。

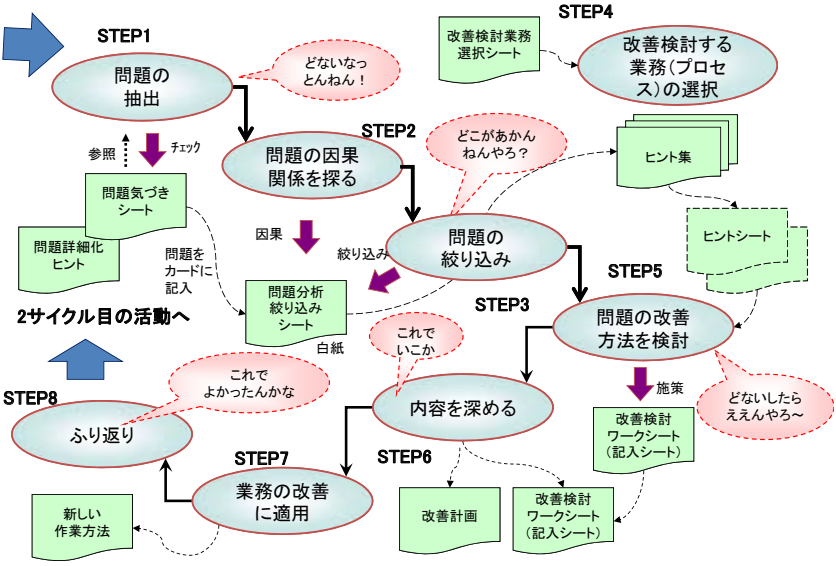


図 2-1-2　8 ステップのサイクルと道具

2.　準備する主な道具

SPINA^{3}CH 自律改善メソッドで使用する主な道具を、以下にリストアップしてみます。これらは、第 4 章にすべて収録しています。

（1）問題気づきシート

STEP1 で使用する道具です。現場で起こっている問題を粗く確認し、チェックします。

（2）問題点カード

問題気づきシートに記載された各開発のステップで起こるかもしれない問題を記載したカードです。

（3）問題詳細化ヒント

問題分析を行う場合の問題点カードの内容を詳細化する時に使用します。

（4）問題分析絞り込みシート

白紙のシートになっており、選んだ問題点カードを並べ、必要に応じて、問題を詳細化し（追記します）、因果関係の分析および改善対象の絞り込みに使用します。模造紙などの大きな紙を別途用意します。

（5）問題分析絞り込みシート（練習用）

あらかじめ問題点カードと典型的な因果関係を記述した練習用のシートです。初めて因果関係分析を行うときや、本メソッドの教育や演習時の時間短縮を行う場合に、印刷された薄い線を太くなぞって利用します。ただし、問題点や分析結果の共有や理解の観点からは、白紙の絞り込みシートを利用して分析を行う方が効果的です。

（6）改善検討業務選択シート

絞り込んだ改善対象から、該当する業務（プロセス）を選択するときの参考にします。

（7）改善検討ワークシート

絞り込んだ改善対象の問題の現況、改善点、対応方法などを検討するワークシートです。これにより、改善活動を推進していきます。

- 問題気づきシート
- 問題点カード
- 問題詳細化ヒント
- 問題分析絞り込みシート
- 改善検討業務選択シート
- 改善検討ワークシート

図 2-2　準備する主な道具

3.　STEP1：「問題気づきシート」を使って、発生している問題を粗くチェック

ここからは、ステップごとに行う作業を STEP1 から順番に解説します。

STEP1 では、まず問題気づきシートを使って、現場で発生している問題を粗くチェックします。

図 2-3 の問題気づきシートにあるように、中央の楕円と矢印で、開発現場の仕事や情報の流れをおおまかに示しています。ウォータフォールの図を意図したものではないので、注意してください。また仕事の区切りに厳密な意味もありません。

楕円と矢印の周囲に配置した問題の例は、日ごろ出会うソフトウェア開発作業の問題を想定して列挙したものです。抽象的な記述ですが、これらの項目から、自分のソフトウェア開発の問題と感じる点と類似した項目をチェックして、「■」の印をつけてください。類似していればよいので、厳密に対応している必要はありません。問題気づきシートに記載されている問題が何を意味しているのか理解しにくい場合は、問題詳細化ヒントを参照してください。そこには、より具体的な問題が記述されていますので、問題の洗い出しのヒントとなります。

また、シートに挙げた項目以外に気づいた問題点があれば、別途付箋紙等にメモしておいてください。

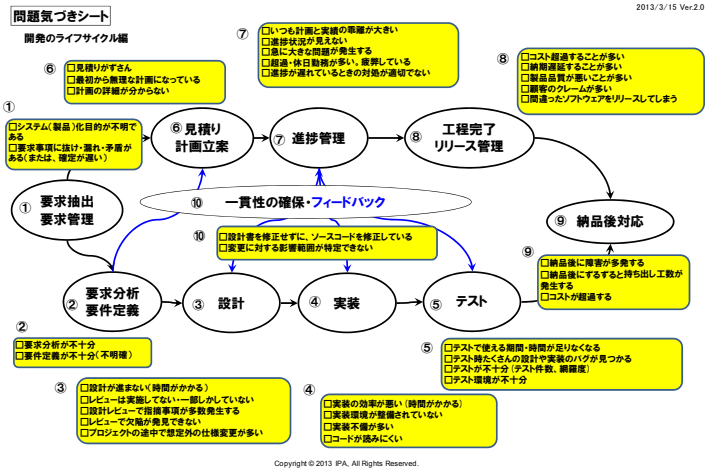


図 2-3　STEP1「問題気づきシート」で問題をチェックする

問題のチェックを行う際、次の点を留意してください。

- 一般論ではなく、事実を基にチェックすることが重要です。
- 自分のプロジェクトや、ある 1 つの現場を想定してください。組織全体や複数のプロジェクトを想定しないでください。
- 現場で起こっていることを粗くチェックしてください。まずはフランクに“そういうことがある”と思ったらチェックしてください。
- “いつもそうだ”あるいは“たまにある”など程度を問わず、少しでも該当する場合はチェックしてください。なるべく多くの項目にチェックが入るようにプロジェクトをふり返ります。
- 問題気づきシートにチェック項目がないが、“困っている”あるいは“いつも問題だ”と思っている事象があれば、別途付箋紙に記入したり、または空きスペースに追記します。
- ここでは、原因追究や改善検討をするのではなく、まず、問題を漏れなくチェックするように心がけてください。

4.　STEP2：問題の詳細化と因果関係の分析

4.1　問題点カードの記入

STEP1 で問題気づきシートに■をつけた項目に対して、詳細化を行います。

まず、図 2-4-1 に示す問題点カードの中から問題気づきシートに■をつけた項目を選び出し、切り離します。問題点カードにない項目があれば、メモした付箋紙を利用するか、空白の問題点カードを利用して同様のカードを作成し、それぞれのカードに対して、問題点の具体的な内容が分かるリアルな事実情報を付け加えます。この情報が、以降の詳細分析の助けになります。

図 2-4-1　問題点カード

4.2　問題の詳細化

問題の詳細化は、図 2-4-2 に示すように、次の観点で検討します。

- いつ（どのタイミングで）、どのようなことがあるのかを生々しく（度合いがあるならそれも）記載します。
- “品質が悪い”、“管理できていない”、“計画どおりに進まない”などの表現は、さらに踏み込んで具体的な程度を記載します。
  例：「管理できていない」→「作業進行度合いは担当者への聞き取りだけで把握している」
- “～不足”、“～がない”という表現は、対策型表現（対応手段を決定することと同等）なので、そのことを要因として起こっている“実際に困っていること”を記載します。
- 第三者でもそのことがありのまま把握できるような表現で記載してください。現状をありのまま共有することが、改善の成功の第一歩になります。関係者全員が理解できる表現になるよう心がけてください。

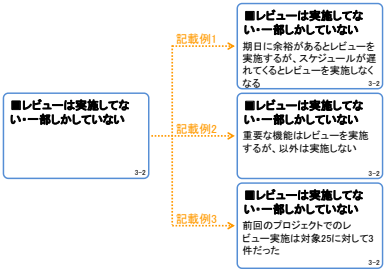


図 2-4-2　問題の詳細化

4.3　因果関係の分析

次に、事実情報を追加した問題点カードや付箋紙を白紙の問題分析絞り込みシートの上に因果関係に従って並べ、それぞれの因果関係を矢印で大まかに結んでいきます。

※問題分析絞り込みシートは、模造紙など問題点カードを自由に並べたり、メモを書き込める十分な大きさの白紙を利用します。

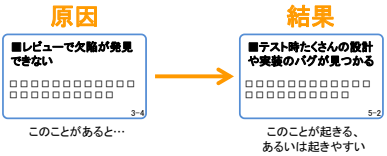


図 2-4-3　因果関係の分析

問題点カード間の因果関係を明確化するには、図 2-4-3 に示すように、以下の手順で実施するとよいでしょう。

- 任意の問題点カード 2 つを左右（または上下）に並べ、左（上）問題点が解決すると右（下）問題点が改善（緩和または削除）される関係が成立した場合に［左（上）→右（下）］と矢印を引きます。
- この対応を順次繰り返して、全体を構造化していきます。
  ※ここでの矢印は、問題点の原因・結果関係であり、情報のフロー（例：対応順や発生順）ではないことに注意してください。

以上の作業により、事実情報に基づいて問題発生の仕組みを構造的なつながりとして理解していってください。

4.4　トラブルモデル

STEP2 の開始時点では、大まかな問題点が並んだ状態でした。それ以降の分析で、問題点を並べ替え、事実の記述を加えることにより、ここまでに自分たちの姿をありのままに示したものになっていると思います。

その姿は図 2-4-4 に示すように、いわゆる“トラブルモデル”と呼んでもよいでしょう。ここでは、自分たちのリアルな姿をつかむことが重要です。完成した“トラブルモデル”は、右側が結果を示す要素になっており、左側はそれに至る過程を示すものになっているはずです。

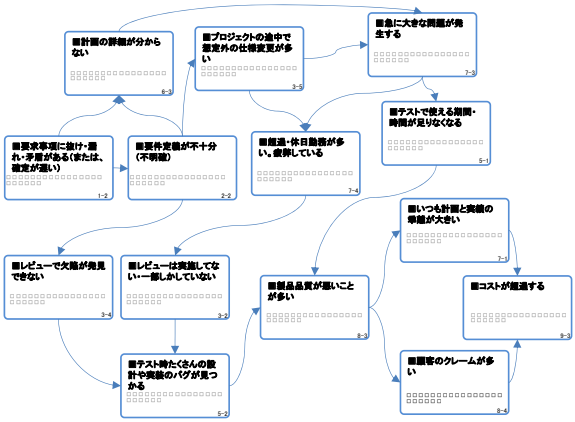


図 2-4-4　完成した“トラブルモデル”

- この分析中に新たな問題点が発見された場合は、新しく付箋紙を用いて問題点カードを追加します。
- 問題点の因果関係を分析するため、関連する問題点を隣り合うように配置してください。
- あくまで、問題点の因果関係を示したものであるので、作業の順序や情報の流れを示すものではないことに注意してください。

5.　STEP3：改善対象を絞る

ここでは、STEP2 で把握した“トラブルモデル”の中で、自分たちがどんな姿になりたいか（たとえば、顧客を満足させる、欠陥を減らす、など）を思い浮かべ、それを実現するために重点的な問題を抽出する作業を行っていきます。

この作業の観点は、次のようになります。

- 目標は、改善に取り掛かりたい問題を、現実的な取り組み可能な数や制御可能な領域に抑え込むことです。また、費用対効果を配慮することも重要です。
- あわせて、明確な原因と言うべきものがあれば、すでに述べた目標を考慮して、そこに改善努力を集中することを試みるようにしてください。
- 真因を特定するばかりではなく、改善すると期待した効果が得られる可能性が高い領域はどこかという視点で探ってみた方が、効果があるかもしれません。ただし、自ら改善せずに他社や他者が改善しなければ効果が獲得できないような他力本願的な領域は、対象外とします（実現可能性が高い場合は対象としてもよいです）。
- さらに絞り込んだ領域を改善する場合の費用（＋期間など）対効果も検討することで、より現実的な改善対応に近づくことができます。
- 以降のステップに進んでから、やはり問題の絞り込みを変えようという進め方も可能です。

改善の成功体験を積み重ねることも、継続性の点で重要です。毎回、真因を探すのではなく、改善の成功が見込める、効果が大きいといった活動につなげることを心がけてください。

図 2-5　改善対象を絞る

6.　STEP4：改善検討する業務（プロセス）を選択する

STEP4 では、STEP3 で絞った改善対象の問題に対応する業務（プロセス）を選択します。

図 2-6 は、改善対象の業務（プロセス）を選択する道具である「改善検討業務選択シート」です。絞り込んだ改善対象と同じ、あるいは類似の業務（プロセス）をここから選択してください。選択時には、ヒント集から該当する「ヒントシート」を抜き出し、「ヒントシート」の 1-(2) の「めざす成果目標」の内容を確認し、改善すべき業務（プロセス）として適切かどうかを十分に考慮してください。

なお、類似の業務（プロセス）がない場合は、関係者と議論して業務（プロセス）範囲を明確にするなど、検討を進めてください。

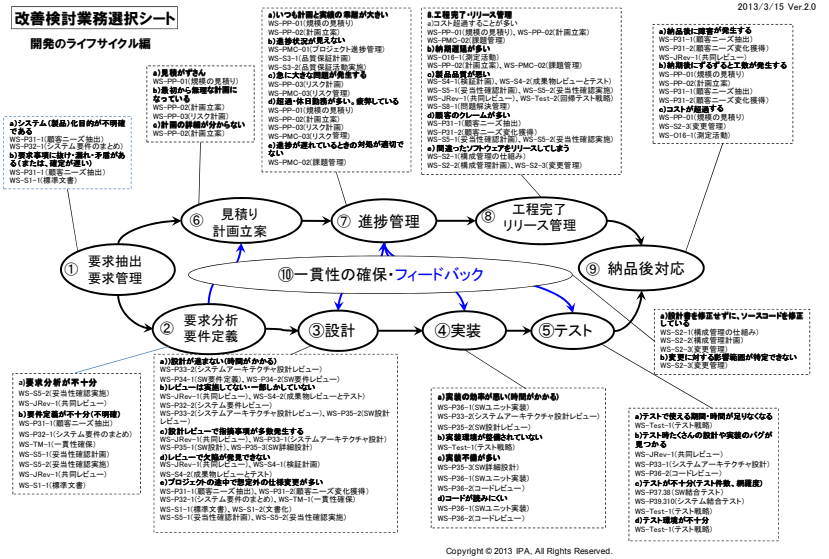


図 2-6　業務（プロセス）の選択

7.　STEP5：改善検討ワークシートを作成する

7.1　改善ワークシートの使用

ここでは、「改善検討ワークシート」を使用して、選択したテーマごとに自分の従事しているソフトウェア開発の具体的な改善方策を考察していきます。短期の改善、中長期の改善といった、それぞれ利用者に適した視点はあってもよいですが、ここに記述したことで、実際に改善に取り組む姿勢が重要です。学校での試験の解答のように「理屈としての正解」を求めているわけではないことに注意してください。

改善検討ワークシートには「記入シート」と「ヒント集」があります。記入シートが実際に作業する改善検討ワークシートです。「ヒント集」は、その作業を支援する情報集になっています。

記入シートは、必要に応じて何枚でも使ってかまいませんが、改善に取り組む業務（プロセス）課題またはそれをいくつかに細分した項目ごとに１枚使用してください。紙ベースで提供している記入シートは記入欄がやや狭いので、ワードプロセッサや表計算ソフト、Web アプリケーションなどを利用して、広めの記入欄を用いることも推奨します。

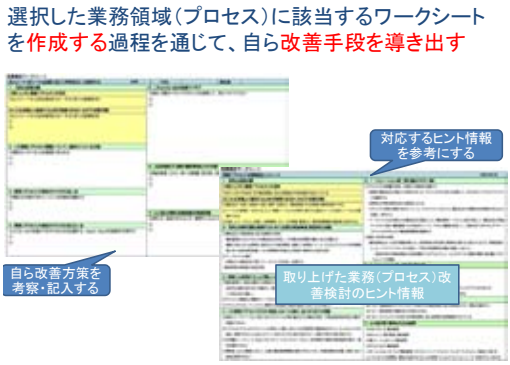


図 2-7-1　改善検討ワークシートの使用

7.2　改善検討ワークシートの記入シート

改善検討ワークシートの「記入シート」の使い方を説明します。

（1）目的と成果目標

選択した業務（プロセス）の目的と、めざす成果を書きます。

これらの項目は、対応するヒント集の中から該当するものを選び、そこに書かれているものを転記します。もし、STEP1 ～ STEP4 の結果として選んだ解決したい課題がヒント集の中に見当たらない場合は、自分自身で業務（プロセス）課題を設定してもいいでしょう。

「これを首尾よく達成すると何が実現できるか（めざす成果目標）」という項目には、「ヒントシート」を参照して、業務（プロセス）課題の達成したい内容を詳しく記入します。前のステップで絞り込んだ問題の解決イメージに近いかどうかを確認してください。

(2) この業務（プロセス）課題について、現在行っている方策

このテーマについて、現場ではどのように対処し、実施しているかを正直に記述してください。現在やっていることは、いずれにしても何らかの製品・サービスに結びついているはずです。そのことを肯定的にとらえる方向で、詳しく思い起して記述してください。作業として時間をかけているかどうかは関係ありません。その視点をもって作業しているなら、その項目を簡潔に書いていきます。

(3) 業務（プロセス）の現在のやり方の良い点

現状のやり方でうまくいっているところを、積極的に書きます。うまくいっていることは忘れず保持し、さらに伸ばしていくという視点が重要です。

(4) 業務（プロセス）の現在のやり方の良くない点

今のやり方でうまくいっていない点、改善したいと思っているところ、顧客やマネージャ等から改善希望があるところなどを記述してください。この項目は、STEP1 ～ STEP3 の結果として、すでに問題を整理したものを引き継いでいる項目ですから、その検討結果を素直に転記してもよいでしょう。改善の候補となるポイントの列挙でもかまいません。

(5) この良くない点の改善アイデア

前項（4）を踏まえ、では実際にどのような改善方策を立案・提案をするかを記述してください。記述に際して「ヒント集」を参照しましょう。ただし、「ヒント集」の記述は「プロセスのモデルが提示している観点」や、「世間でいろいろな条件下に提案されている事例」にすぎないので、自分が従事している実際の業務（プロセス）の改善方策として妥当か、あるいは十分かなどは、検討してみないとわかりません。改善検討ワークシートへの取り組み者が、自分の状況を踏まえ、創意工夫して提案・立案する必要があります。丸写しは禁物です。

(6) 上記改善をする際の懸念事項とその対策

前項（5）で提案・立案した案を実際に実現するには、コスト面、制度面、役割分担等で制約となる事柄が存在する可能性があります。そのことをできるだけ

明示しましょう。また、この提案・立案の実現のためには、周囲の関係者（チームの同僚､チームリーダや上司､顧客など）の協力が必要となるケースも多いので、それらについての要望等を記載しておきましょう。

（7）ふり返りの際の改善進捗の概要評価

この項目は、ふり返りのとき、この記入シートの内容を点検する際に用います（最初の改善提案時点では空欄のままです）。

（8）留意点

「4. 業務（プロセス）の現在のやり方の良くない点」では、CMMI® や SPEAK-IPA などのモデルに照らしてできていないことを取り上げるのではなく、問題分析絞り込みシートで出てきたリアルな問題に対して、改善すべき点を検討するようにしてください。

図 2-7-2　改善検討ワークシート（記入シート）

7.3　改善検討ワークシート（ヒント集）

次に、「改善検討ワークシート」の「ヒント集」の使い方を説明します。

「ヒント集」には、図 2-7-3 のように、それぞれの業務（プロセス）課題に沿った改善ヒント情報が記載してあります。STEP5 の作業の基本は、記入シートの各項目に沿って検討し、記入を進めることです。その際、「ヒント集」に有益な情報や点検すべき情報が書かれていますので、できるだけ役立てていただくのがいいのですが、あくまで検討の参考情報としてください。

（1）目的と成果目標

選択した業務（プロセス）の目的、めざす成果が書かれています。

「これを首尾よく達成すると何が実現できるか（めざす成果目標）」という項目は、業務（プロセス）課題の達成したい内容を詳しく説明したものです。具体的な「成果物」を得たいという項目ではなく、業務（プロセス）の実施によってどのような状態が実現されていればよいかを述べています。

（2）目的と成果目標を実現するために必要な実施事項（典型的な活動）

前項（1）の目的と成果目標を実現するための典型な活動を示します。この項目は、大きく言えば、プロセスのモデルが示唆しているプラクティスにあたります。しかし、モデルの示唆は現場の実情に応じて解釈する必要があり、必ずしもすべての項目が現場に対して有効であるかどうかはわかりません。解釈して、必要な項目を選択するようにしてください。

（3）現実には実現することが難しい点

典型的な活動を実施するときに、実現することが困難な内容が示されています。現場で起こっていることと関連づけて困難な点を分析し、それを克服したり、回避したりすることも考慮して、有効な活動を検討してください。

（4）この業務（プロセス）をうまく実施しなかった場合、起こるであろう問題

この業務（プロセス）をやらなかった場合に、往々にして起こり得る問題の例を列挙しています。現場で起こっている問題と関連がある可能性があります。

（5）ソリューション例

・取り組みやすい例

目的と成果目標を実現する典型的な活動を具体化、また改善を実施するための、実際の事例になります。事例であり、自分の現場にそのまま当てはまるとは限らないので、あくまで参考としてください。

・深く取り組む例

ソリューション例１の応用編と位置づけられる他の事例を提示しています。

（6）関連する他の改善検討ワークシート、参考プロセス項目

本テーマを実施するにあたり、参考にした方がよい他の改善検討ワークシート（ヒントシート）の項目です。

（7）ふり返り等で参考となる文献等

本業務（プロセス）について、さらに深く知りたい場合は、これらの資料を参照してください。SPEAK-IPA のプロセス、CMMI® のプロセス、その他の参考項目と資料文献を示しています。

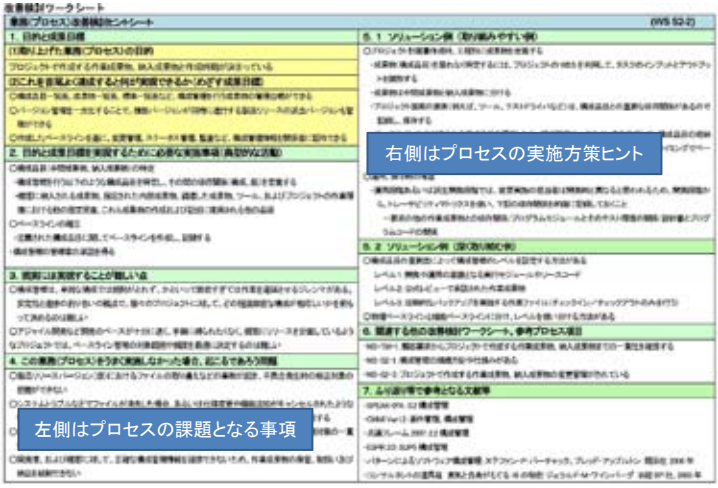


図 2-7-3　改善検討ワークシート（ヒントシート）

7.4　２回目以降のサイクル

本手法では、改善サイクルを回し、繰り返して分析と改善提案を行います。

その過程は、トライ＆エラーです。間違いややり過ぎを過度におそれてはいけません。「やってみなければ本当のところはわからない」という精神です。図 2-7-4 に示したのは、基本的には前に示した図 2-7-2 と同じものですが、2 回目以後のサイクルで使用するときのために、多少タイトル等の用語を変えてあります。

図 2-7-4　２サイクル目以降の改善検討ワークシート

7.5　作業の情報の流れ

STEP5 で行う作業の流れをわかりやすくするために、記入すべき情報の関連性を図 2-7-5 に解説しました。太い矢印に沿って、考察や情報参照が進みます。

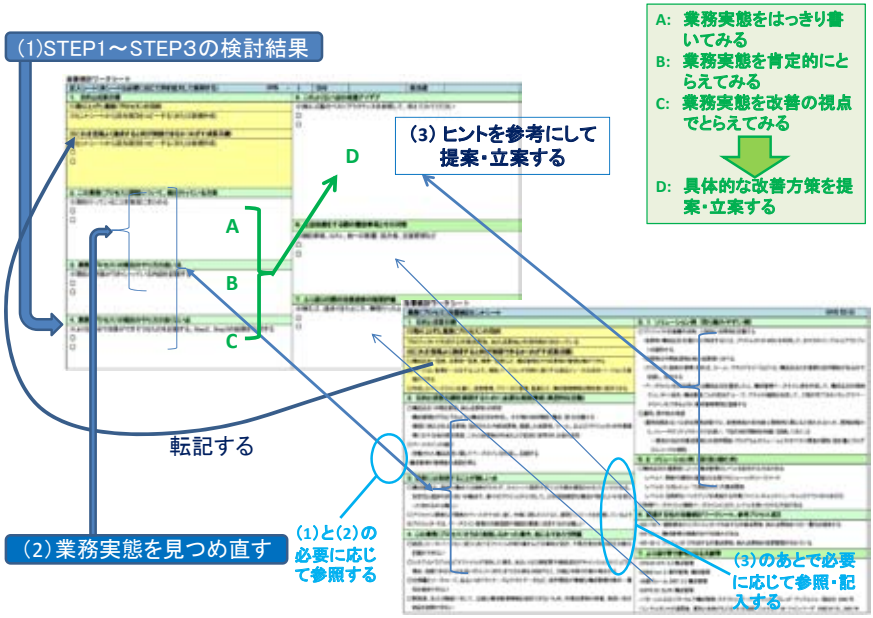


図 2-7-5　STEP5 の作業の情報の流れ

8.　STEP6：チーム討議や専門家との討議により具体的な改善計画に落とし込む

STEP5 までの内容で改善活動を開始してもかまいませんが、活動を他のチームに展開したり、組織の改善活動に持ち上げるためには、関連するチームや、専門家、組織の上位者を入れて討議すると効果的です。そのときには下記の内容に留意してください。

- 改善活動終了時点の改善の成果や達成目標を、より具体的にする。
- 改善内容に応じて取り組み方が異なるので注意する。たとえば、現場レベルで行う場合と組織レベルで行う場合は、体制や制約条件が異なる。
- 組織レベルで実施する場合は部門責任者と認識合わせを行い、組織的な活動として取り組む。

改善検討ワークシートの「記入シート」に改善実施プランを記述してもかまいませんが、別途、改善計画を作成した方が、より活動が確実になります。

その場合､「改善計画･実績シート」の活用や下記の内容を考慮するといいでしょう。

- 具体的な施策、取り組み内容
- 活動の成果、達成目標
- スケジュール（実施項目、担当者、日程）
- 制約条件、留意事項、など

改善計画を作成する上で留意する事項は、次のとおりです。

- 活動期間は３ヶ月～半年程度を目安とし、長期に及ぶ場合は活動をいくつかのステップに分ける。
- あまり重装備な計画を立てると計画倒れになり、改善が進まないこともあるので注意する。

このSTEP6で具体的な実施計画を立てる時に、組織の改善組織メンバが関わる場合は、組織の改善活動に今回の改善がつながるように導いてください。また、プロジェクトリーダの場合は、次回のプロジェクトで実施できる改善実施プランにしましょう。具体的な担当者を明示することも忘れないでください。

部門長の場合は、部分的な試行をベースに組織内の他プロジェクトに波及効果があるような改善施策にすることも考慮するとよいでしょう。

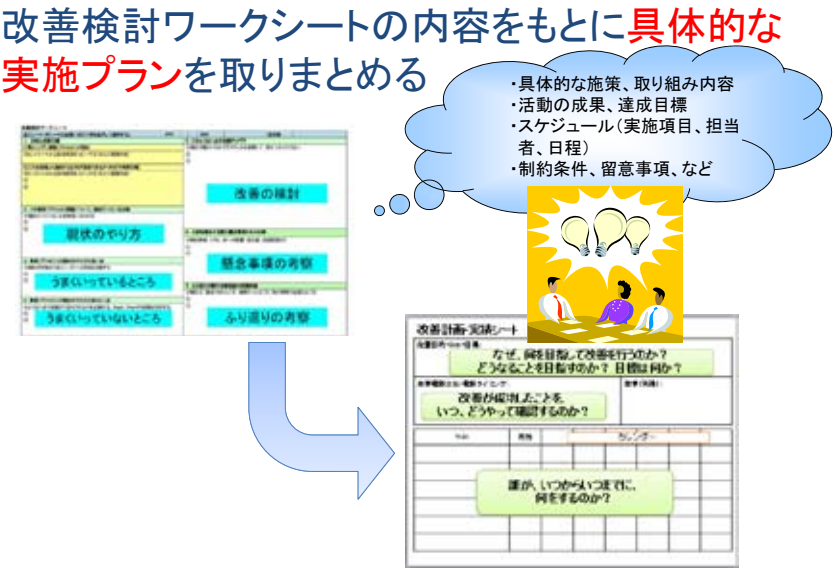


図 2-8　具体的な改善計画への落とし込み

9.　STEP7：これまでの検討結果を業務の改善に適用する

ここまでのステップで検討して明確化した改善計画に基づいて、改善を実践します。

実践前までに関係者全員に、改善の目的やめざす効果、対応方法、それぞれの役割と重要なマイルストーンを含めたスケジュール、成果物、管理方法などを説明し、疑問点や不明点を払拭してから始めてください。

改善手段は、下記のことに注意して実践しましょう。

- 目的や目標、改善の意義や意味を把握しない事務的、形式的な対応を避ける。
- 対応領域が大きい場合は、まず小さい領域で短期間試行し、得られた教訓などを整理してから広げることも考慮する。
- もし当初の検討結果よりも“より良い対応方法”が見つかれば、関係者間で共有した上で実践に移す。

- 実践中に発生した問題はタイムリーに報告し、関係者で共有する。
- 実践した結果、分かったこと、感じたこと、改善方法で考慮不足だった点とその打開方法などを書き留めておく。

管理者は、逐次改善の実践状況を把握して、問題があれば可能な限り早めに手を打つようにしてください。たとえば、改善手段では想定した効果が獲得できないことが分かった場合は放置せず、現状とめざす効果を再確認して、改善手段や改善計画を調整・見直しする必要があります。

また、実業務の状況変化により当初の計画や対応スケジュールが順守できない事態になることもあるでしょう。実業務対応が最優先であることは事実ですが、実業務を優先しすぎるあまり改善が消滅しないように、適宜計画内容を関係者と調整しながら進めてください。

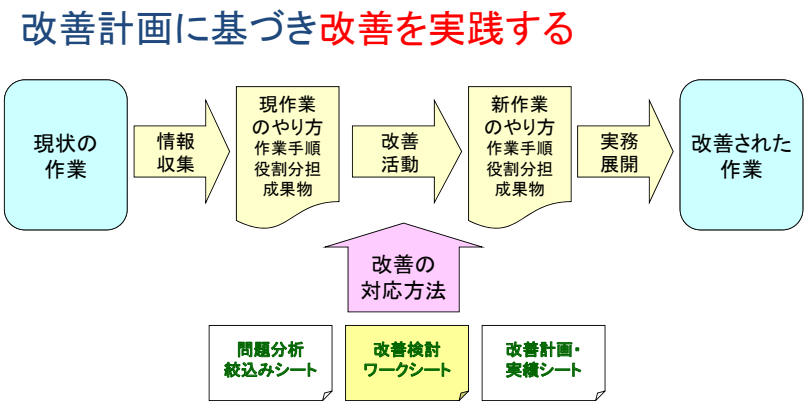


図 2-9　改善の実践

10.　STEP8：ふり返り

改善実践の結果や得られたノウハウ・教訓等をとりまとめ、関係者や組織内で情報・認識を共有し、以降のより良い改善実践や次の改善サイクルにつなげるためにふり返りを行います。

比較的短い期間で継続して改善を実施する場合には、ふり返りを簡単に実施して、次のサイクル（STEP1）を始めてもよいですが、ふり返りを組織的に行う場合、改善対応期間が長い場合や改善運営に慣れていない組織では、次のように行うと効果的です。

10.1　ふり返りの準備

（1）ふり返りの観点を設定する

ふり返りの観点を事前に設定します。観点には、計画した改善対応事項に対する実施実績と、改善目標に対する成果実績の両面を加えると、次のサイクルに有効な情報を引き継ぎやすくなります。

（2）実績情報を収集、整理する

前項（1）の観点でも触れたように、改善計画に対する対応実績は「①改善対応事項の実施実績」「②改善でめざした目標や成果に対する実績（達成度など）」を主な対象とします。これらとは別に、自らふり返りが必要と判断した情報があれば、追加するとよいでしょう。

これらの情報をふり返り直前に別途収集・整理しなくてもよいように、改善計画立案時点で、関連情報の収集･集約･保管方法等を明確化しておくと効率的です。

10.2　ふり返りミーティングの実施

ふり返りミーティングは下記のポイントに注意して実施してください。

（1）関係者全員の参画

ふり返りは可能な限り、関係者全員が参画するミーティングとして開催するのがよいでしょう。どうしても出席できないメンバについても、あらかじめふり返り結果を収集しておき、ミーティング時に共有するようにしてください。

（2）観点の共有と議論の流れの重視

全員でふり返りの観点を共有した上で、まずは「Keep：良かったこと・継続し

たいこと」から始め、次に「Problem：うまくいかなかったこと・問題点」に移り、最後に「Try：次にやりたいこと・次はどうすればよいか」を明確化します。

（3）進行役が適切に主導した運営

参加したメンバが自由にコメントを出せる場を作り出し、多少脱線してもかまいませんが、筋道を外さないようにしてください。有効な議論や情報共有、メンバ内の気づきを促進してください。

（4）全員がバランスよくコメントを出しながらワイワイ議論

慣れない場合でもメンバひとりひとりが最低でも 1 つはコメントを出すように運営します。特定のメンバだけがしゃべっているような場にしないことが重要です。

例：1 回に一人 1 コメントを出したら隣にシフトする・・・をぐるぐる回す、などゲーム感覚を取り入れるなどの工夫も有効です。

（5）前向きで建設的な議論

責任追及や特定の個人への非難、愚痴り合いなどは後ろ向きな議論になりやすくなります。良かったことを先にして、どうしたらもっと良くなるか、実現できるかを、前向きかつ建設的に議論してください。

（6）全員の合意獲得と記録

議論して出た結論は、参加者全員の合意を獲得して記録に残してください。

（7）「Try：次にやりたいこと・次はどうすればよいか」の具体化

Try 事項の検討時は、実際に対応する方法が具体的にイメージできるレベルに落とし込みます。“次回対応時に明確化する”などの抽象的な内容では、次の活動につながりにくくなります。

特に改善成果が期待どおりでなかった場合は、以下の点を再確認してください。

- 参加者の協力体制と取り組み状況
- 改善の達成目標と組織のスキルとの関係
- 改善の手法と進め方の内容

（8）最後にふり返り結果を整理して全員で情報と認識を共有

議論した内容をそのままにして終わるのではなく、見出しをつけて類似事項を集約し、編集するなど工夫した上で、関係者全員で再確認します。

改善で獲得した経験・ノウハウ情報を関係者で共有するだけでなく、改善の意味や意義、作業連携におけるチームワークの重要性などについて、認識を共有することが重要です。

これまでの改善活動を関係者でふり返り、実施してきた内容と成果の良かったところ、うまくいかなかったところ、次にどうするのがよいかを明確化し、共有する。この結果をもとに次の改善サイクル（STEP1～STEP7）につなげる

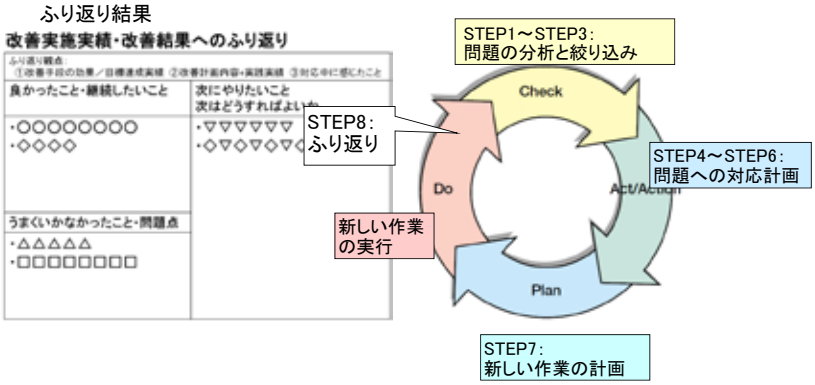


図 2-10　ふり返り

11．継続的な改善サイクルへ

11.1　改善に対するモチベーション

改善を継続して成果を上げていくには、改善を実践するメンバのモチベーションが重要です。その対策として下記のような工夫も有効です。

- メンバが改善で得られた効果を実感できる機会を増やすために、改善対象をできるだけ絞り込み、短期間で効果を確認できるように運営する。このことは、実業務で多忙な中で改善対応の負担を少なくする効果も期待できる。
- 組織的な改善活動の場合は、活動報告会や表彰制度などを設ける。
- 改善への取り組みや成果を社内外に発表する機会を作る。

●さらなる改善へ
●組織展開
●新たな課題に挑戦

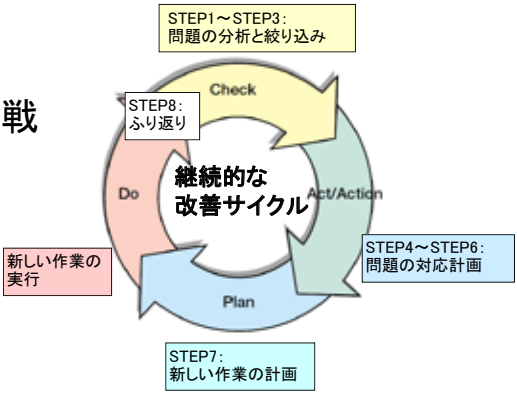


図 2-11-1　継続的な改善サイクルへ

11.2　新しい課題への挑戦

改善を継続することにより、一定のレベルで欲しい成果を獲得できたら、チームや組織として達成したい成果の実現に向けて新しい課題に挑戦してみましょう。

このような対応により、改善のマンネリ化を防ぎながら、組織としての競争力向上につながる対応に昇華させていくことが可能になります。

11.3　段階的にプロジェクト活動と一体化

はじめのうちは改善を手探りで運営するため、実プロジェクトとは別物として対応することが多くなると思います。しかし、継続して改善を回しているうちに、改善運営のコツがわかってくるはずです。

改善運営に慣れてきたら、下記の項目などを対象として、段階的にプロジェクト活動との一体化を進めてみてください。

- 改善計画立案とプロジェクト計画立案
- 改善状況把握とプロジェクト進捗管理
- 改善ふり返りとプロジェクトのふり返り

たとえば、下記のように段階的に改善活動を高度化するのも一案です。

- プロジェクト計画立案時に一緒に改善計画を立案し、相互に連携させる。
- 改善運営に活用していた各種様式や記載項目を、プロジェクトで使用している各種様式に組み込んで一体化してみる。
- 改善活動とその成果だけをふり返るのではなく、改善対象を含めた業務（プロセス）全体の実践状況と成果実績を収集・整理し、プロジェクトにおけるふり返りと連携・統合する。

このような継続対応によって改善活動の効率化を実現しながら、プロジェクトへの改善効果を高めていくとよいでしょう。

11.4 組織内への展開

改善によって得られたノウハウや成果は、対象プロジェクトだけで所有するのではなく、他のプロジェクトや組織などに有効活用して、全体で改善成果を獲得するのが効果的です。

その実現は簡単ではありませんし時間が必要になると思いますので、下記のような対応をきっかけとして徐々に広げていくことを考慮してみてください。

- 報告会やポータルサイトへの掲載などを通じて、関係者と情報共有を図る。
- 改善を推進する部署があれば、社内の類似プロジェクトへの適用提案を行う。
- 社内管理者やエンジニア教育の一環として、改善事例の活用ワークショップを開催する。
- 組織全体の事業計画項目の１つに、改善で獲得したノウハウや成果を活用した組織としての改善計画を盛り込んで実践し、組織としての成果を評価しながら運営する。

このように、「SPINA^{3}CH 自律改善メソッド」を活用して改善を継続的に実践する風土が組織内で醸成できれば、組織は継続的に成長していけるでしょう。

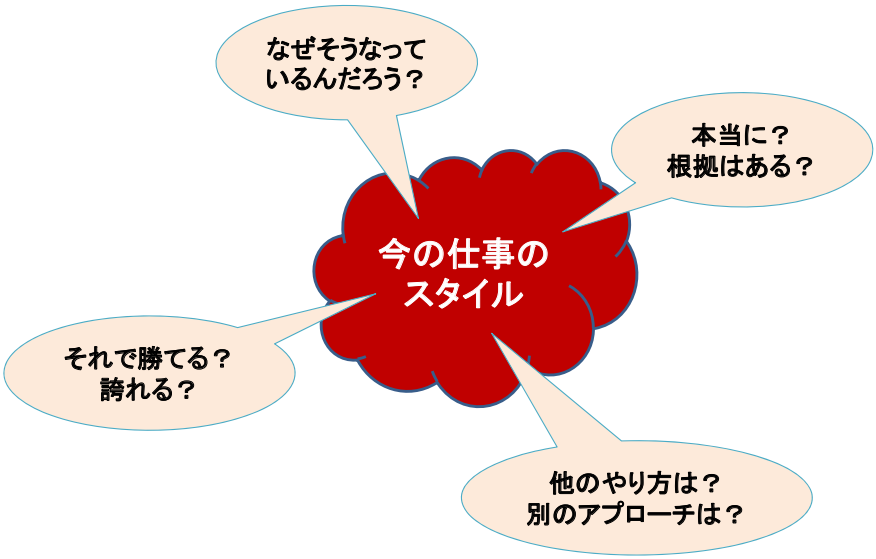


図 2-11-2　組織の継続的な成長のために

第３章　活用編

1．ワークショップの活用

SPINA^{3}CH 自律改善メソッドを組織内に展開する際、自ら資料を参照しながら、あるいは座学を中心としたアプローチに頼りながら進めると、実務に適用するときにどうしてよいか分からない、不適切なまま進めて思わぬ手戻りが発生する、欲しい結果が得られないなどの弊害が予想されます。そのようなリスクを回避する有効なアプローチの１つとして、実際の改善検討対応を「ワークショップ」形式で実施する方法があります。

SPINA^{3}CH 自律改善メソッドを体得したメンバ（必要であれば外部の有識者）が運営役＝チュータとなり、これから改善を検討するグループや個別のメンバを受講者として、実際の題材を対象にワークショップ形式で改善検討を実践することが有効です。

また、改善推進者の育成を目的として、当メソッドの活用方法や注意事項を習得するためのワークショップを実施することも可能です。

以下に、ワークショップ活用に必要な準備事項、実運営方法、および各種注意事項を記載します。

1.1　目的・目標の設定

ワークショップの目的と具体的な目標（ゴール）を設定します。ワークショップが終了したときにどのようになっていればよいのか、それを確認する方法は何かなどを具体的にイメージできるレベルで設定するのが理想です。

1.2　制約条件の確認

ワークショップ内容を設計するにあたり、受講者の特性と業務内容、使用できる施設・設備・情報、運営体制などのリソースを把握します。

1.3　ワークショップの準備

主に以下に示す内容を、ワークショップの目的および制約事項を考慮して具体化します。

（1）運営体制

- チュータ

  ワークショップの運営全体を取り仕切るチュータの運営力が、重要な成功要因となります。楽しく学べる雰囲気作りと、重要なポイントを外さずに本質に取り組むファシリテーションの実践が、チュータに求められます。受講者が多い複数グループによるワークショップの場合は、全体運営を取り仕切るリードチュータとは別に、各グループにそれぞれ専属で支援するチュータを付けるなどの工夫も必要です。

- グループ構築

  改善検討するメンバによってグループを構築します。理想のメンバ数は 1 グループ 3 名ですが、受講者数が多い場合は、最大 5 ～ 6 名程度を目安にするとよいでしょう。

（2）準備する資料と資材

- 各受講者向け

  ワークショップの段取りと当メソッドの重要なポイントを記載した資料を事前に配布します。

- 各グループ向け

  グループで 1 つの成果を構築する場合は、グループごとに 1 部用意します。

  課題を中央に置いて議論や検討ができる座席と机（2 台の横長机を合わせるなど）を用意します。

  - 模造紙またはホワイトボード
  - 付箋紙（可能な限り複数色、12.7cm × 7.6cm 程度）
  - 中太のペン（可能な限り複数色、人数分）

- 張り付け用テープ
- STEP ごとに使用する各種資料(問題点カード、各種ワークシートなど)

(3) 段取り設定と時間配分

以下の内容を含めて、ワークショップの段取りと時間配分を設定します。

- ワークショップの目的・目標の共有
- ワークショップの全体像と各種段取りの共有
- 各グループによる改善検討
- 結果の共有と講評および総評

次に、1 日のワークショップの時間配分例を示します。

STEP	開始時間	終了時間	時間	実施内容
1	9:30	9:35	0:05	ワークショップの狙いと進行方法共有
2	9:35	9:45	0:10	メンバ自己紹介
3	9:45	9:55	0:10	改善目的の共有
4	9:55	10:15	0:20	問題気づきシートによる問題抽出
5	10:15	11:00	0:45	問題詳細化
6	11:00	12:00	1:00	因果関係分析
7	13:00	13:30	0:30	改善対象特定
8	13:30	13:50	0:20	ワークシート選定
9	13:50	16:20	2:30	ワークシート作成(目標1枚仕上げる)
10	16:30	17:00	0:30	各グループから報告
11	17:00	17:15	0:15	総評
12	17:15	17:30	0:15	アフターミーティング

図 3-1-1　ワークショップの進め方（例）

1.4　ワークショップの実施

ワークショップ運営を取り仕切るメンバがチュータ役を含めて複数いる場合は、それぞれの役割分担や運営上の注意事項を共有するため、事前に意識合わせを行ってください。

以下に、ワークショップを実施する際に留意する事項をまとめます。

- 様々な視点での意見交換と検討を促す。
  例：特定の意見だけがまかり通っている場合は、異なる観点による検討を促進する。
- 時間を見ながら、“発散から集約へ”とシフトしてもらう。
- 安易に答えを教えない。可能な限り、受講者自身が自ら試行錯誤し、新しい視点や異なる価値などに気づくよう、運営を工夫する。
- 適切な考え方や有効な実践があった場合にはそれを認め、不適切な考え方やポイントを外した場合はフィードバックを実施する。
  例 1：ワークショップ中によい実践を見つけたらすぐに受講者と周囲のメンバにその旨と意味を伝える。
  例 2：不適切な考え方やポイントを外した実践を見つけたら、直接駄目出しせず、対応の意図の確認と大事なポイントの再確認を促す。
- 思考停止している受講者や集中できていないグループには、思考を促す。
  例 1：ありきたりの答えで安易に進もうとするグループには、問題点やリスク、異なる視点を伝える。
  例 2：疲れが目立つ場合は、チョコレートや飴などを補給する。
- Q&A の時間を随時設けるなど、受講者が疑問点をそのまま持ち帰らないように工夫する。

1.5　フォローアップ

ワークショップ終了後、実際に改善が進んでいるか、不明点などで止まっていないかなどを把握して、疑問点や前に進まない障害事項を取り除く＝可能な限り改善を促進するためにフォローアップを行うのが理想です。

2.　チーム討議の進め方

自律改善メソッドを現場に適用する際は、通常、現場のチームで検討する形をとります。

チームによる検討を効果的に進めるには、プロセス改善推進者という自覚と知見をもった人が、メソッドを適用するミーティングのリーダシップをとることが推奨されます。そして、現場チームが改善を実践するときは、全員が積極的に参画して進めることが重要です。

チームのメンバが集まって現状を把握し、改善方法を討議して進めることができるように、現場チームにおける討議の進め方をここで整理しておきます。チームによる討議の準備と実施に必要な事項の多くは、本章の「1. ワークショップの活用」で解説した内容を適宜読み替えることで把握できます。そのため、ここでは特に現場チームの討議に必要な配慮事項について記載します。

2.1 討議時の配慮事項

(1) 全員参画

特定メンバだけの意見やコメントで終わらないように、各自のオリジナルな意見やコメントを収集してください。「なぜそう感じたのか？」「具体的には？」「○○さんはどう思う？」などを問いかけて、メンバそれぞれが常に“考える”ようにアプローチします。

(2) 前向きで建設的な検討と議論

他者の責任追及、あるいは後ろ向きな意見やコメントばかりにならない場作りを心がけてください。異なる意見や視点を歓迎し、様々な切り口から前向きで建設的な検討と議論を行います。

(3) 現状把握時の評価禁止

意見やコメントへの即時評価は禁止です。推論や一般論を排除して事実準拠で確認し、具体的に把握してください。

(4) 改善検討時のバランス

前向きで建設的な議論を促進し、目的・目標をどうしたら実現できるか、および、その実現可能性をバランスさせます。他の手段がないかどうかも検討してください。

（5）本音が言える場にするための安全確保

本音や事実をありのままに言える場を作ることが重要です。本来、これは普段の業務運営時から考えるべきことですが、少なくとも討議時は、各自が意見やコメントを安全に言える場にしてください。

例：議事録には誰が言った意見なのかを載せない、他言無用で運営する、など

（6）納得がゆく合意形成

無理矢理「Yes」を言わせる、みんながそう言うからなんとなく合意する、というのではなく、様々な意見やコメントを、目的・目標や根拠・条件などから集約し、チームが持つべき共通の価値観に立脚した判断と合意形成をしてください。

2.2　討議を通じて認識共有すべきこと

現場チームの討議を通じて、以下の事項に対してチームのメンバ全員が共通の認識を持つことを促進してください。

- チームが持つ目標や大事にしたい価値観
- チームの現状と課題（改善必要事項）
- 各自の役割と周囲の期待
- メンバそれぞれの考え方や価値観と現状
- チームワークの重要性と各自に求められる具体的行動

3.　チームリーダの心構え

改善活動は多くの場合、複数人から構成されるチームで実施されるため、メンバは改善活動だけでなく、開発運用業務を抱えながら活動しています。そのため、開発の遅れやトラブル対応が、改善活動の遅れに繋がるというリスクが潜んでいます。その中でチームリーダに必要な心構えについて、次にまとめます。

(1) メンバの自主性を尊重する

有意義な改善活動とするには、リーダからの指示に対してメンバが動くのではなく、課題の抽出から解決策に至るまで、メンバによる自主性が重要です。結果について責任を持つことを示し、背中を押してあげることが大切です。

(2) ナビゲーションに徹する

リーダは知識と経験が豊富なことが多く、ついメンバに対して方針や解決策を提示してしまいがちです。それでは継続的かつ自律的な活動にはつながりません。リーダはメンバの発言や行動を注視しながら、ナビゲーションに徹することが大切です。

(3) スキルを磨き、知識を吸収する

「開発に適用される新技術」「プロセスモデルやプロセス技術」「活動に必要な道具の理解」から他社の事例に関する情報収集まで、幅広い知識が求められます。

(4) チームビルディングに努める

各メンバは改善活動以外の仕事を持っていることが多いため、思いどおりに進捗しないケースや、改善策が予想どおりの結果に結びつかないケースがある一方、思いがけなく逆に大きな成果に結びつくケースもあります。どのような状況下でもメンバのモチベーションを高く保つことを心がけ、努力することが求められます。

(5) 活動を「見える化」する

改善活動は、新しいものを生み出すというよりは、現状の課題へのアプローチとなることが多いものです。そのため、活動そのものや、成果や結果が見えにくくなることもあります。だからこそリーダは、常に活動を見える化し、その途中経過と結果について、定量的かつ定性的に示すことが求められます。

4.　改善活動の推進者（指導者）の心構え

改善活動そのものは、あくまでも現場が中心となって実施しなければ、実のある結果に結びつきません。ですから改善活動の指導者は、現場で実施されるプロセス改善活動をうまくフォローしていくことを念頭に置いて活動する必要があります。

この点に注意して、改善活動の推進者（指導者）としての心構えについて、以下にまとめます。

4.1　改善活動時の心構え

（1）関係者の意見をよく聞く

改善活動には、多くの関係者が存在しています。改善活動の主体は現場ですが、その他にプロセスオーナや改善活動組織の組織長、あるいは経営層まで関係する場合もあります。改善活動の指導者は、それぞれの立場における意見を収集し、組織としてめざすべき方向性を見据えた上で、改善活動のフォローを実施していく必要があります。

ときには、現場のやりたいことと上位層が求めるものが異なる場合も発生します。そのような場合でも、客観的な立場からお互いの理解が進むように調整していくことも、推進者としての重要な役割です。

（2）現場に足を運ぶ

活動の主体であり、実際に問題を抱えている現場の状況は、聞き伝えや連絡・報告だけで把握するのではなく、自らが出向き、重要性や緊迫感を肌で感じることが重要です。そうすることによって、自分も改善活動の一員であるという意識が生まれます。客観的な立場を保つことも重要ですが、さらに一歩踏み込んでフォローすることで、現場からの信頼を獲得でき、より良い改善に向けたアドバイスにもつながります。

（3）多くの事例を吸収し、組織に合致した改善の方向性をアドバイスする

改善活動の推進者は、様々な現場の改善活動に立ち会う機会がよくあります。多くの現場を見聞きして肌で感じることにより、自ずと経験値は上がります。経験を積むことによって改善に対する引き出しを増やし、それらを活用しながらフォローを行うことが重要です。

ただし、改善対象の組織が変われば、改善の目的や制約なども異なるので、過去の事例をそのまま活用できない場合が多いことに気をつける必要があります。改善活動の推進者は、過去の事例を参考にしながら、その時々で対象組織に合う方向を考えて、フォローしていかなければなりません。

知識	開発に適用される新技術 プロセスモデルやプロセス技術 改善推進に必要な道具の理解 他社の事例
経験	改善活動の経験 改善活動の推進経験
マネジメント	問題分析力 コミュニケーション力 計画力 定量的・定性的な評価実施力

図 3-4-1　改善を推進するリーダに必要な要素

4.2　各 STEP における留意点

SPINA^{3}CH 自律改善メソッドを用いて改善活動を行う際、特に初めは、推進者（指導者）によるフォローが必要です。本書の「第 2 章 実施手順編」の各 STEP に対して、特に留意すべき点をまとめます。

・STEP1「問題気づきシート」を使って、発生している問題を粗くチェック

- 様々な視点から改善の方向性を見つけ出すため、なるべく多くの項目にチェックが入るようにプロジェクトのふり返りで現場で起きている問題をお互いに出し合うのが良いでしょう。

- 同じプロジェクトのメンバでふり返りを行うと、問題に対する背景や経緯が共有でき、有効な問題がチェックできます。
- 複数のプロジェクトを想定してすべての問題を挙げるよりは、1 つのプロジェクトを対象とする方が、後の分析で要因追究が容易になります。
- 対象のプロジェクトで頻度が少ない問題や気づいた点もチェックします。
- 一般論ではなく、事実に基づいてチェックすることが重要です。
- 原因追究や改善検討をするのではなく、まず問題をチェックすることを心がけます。

・STEP2　問題の詳細化と因果関係の分析

- 分析中に新たな問題点が発見された場合は、検討の対象として新たに追加するよう伝えてください。
- 問題点の因果関係を分析するので、関連する問題点を隣り合うように配置します。
- 問題点の因果関係を示したものであり、作業の順序や情報の流れを示すものでないことに注意します。

・STEP3　改善対象を絞る

- 改善の成功体験を積み重ねることも、継続性という点で重要です。いきなり根本原因を探すのではなく、改善の成功が見込める、効果の大きい活動につなげられる対象に絞り込むことから始めるようにしてください。

・STEP4　改善検討する業務（プロセス）を選択する

- 選択にあたっては、ヒントシートの「1-(2) めざす成果目標」を確認し、さらに「4. この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題」も確認するよう促してください。自分たちが取り組む課題がイメージしやすくなります。
- 該当する業務（プロセス）が見つからない場合、絞り込んだ問題がどのような業務上の支障に繋がっているのかなどを再度話し合うと解決する場合があります。

・STEP5　改善検討ワークシートを作成する

- 改善検討ワークシートの「4. 業務（プロセス）の現在のやり方の、良くない点を書いてください」の部分では、CMMI® や SPEAK-IPA などのモデルに照らしてできないことを取り上げるのではなく、問題分析絞込みシートで出てきたリアルな問題に対して改善すべき点を検討してください。

・STEP6　チーム討議や専門家との討議により具体的な改善計画に落とし込む

- 実施プランは、第三者が見ても理解できる程度に具体化されているか確認してください。改善を実施する人やチームの改善経験も考慮して、場合によっては現実的な方法を実施するようにアドバイスすることも大切です。

・STEP7　ワークシートの検討結果を業務の改善に適用する

- 改善策実施にあたって、関係者の支援・協力が得られるように推進者が働きかけることが必要な場合もあります。

・STEP8　ふり返り

- ふり返りは、改善活動が継続していけるか否かの分かれ道になることがあります。改善に慣れていないチームには、推進者がリードしてふり返りを行うことも必要です。

5. 活用事例と活用のヒント

ここでは、以下の５つの事例ないしヒントを紹介します。

- ケース１：プロセス改善の経験が浅い
- ケース２：プロセス改善を実施している
- ケース３：プロセス改善に精通した人が一人でプロジェクトに適用する
- ケース４：CMMI® レベル３達成組織の自律改善に適用する

5.1　ケース 1：プロセス改善の経験が浅い

このケースは、組込みソフトウェア開発の統合支援（ビルド、リリース準備など）を行うチームで、3 ヶ月間の実証実験として試行したものです。

チーム構成は、マネージャ 1 名とメンバ 3 名です。プロセス改善の経験が浅いメンバで主体的に進め、マネージャは状況により適宜フォローする方針としました。ただし、メソッドの使い方については別途導入支援者が随時ガイドしました。

ステップごとの実施経過の概要は図 3-5-1 のとおりです。

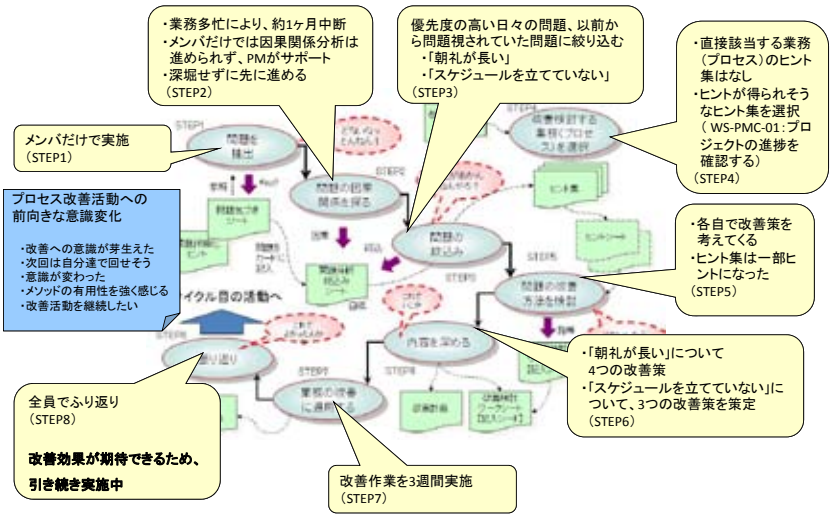


図 3-5-1　ケース 1 の実施経過（STEP1 ～ STEP8）

（1）問題点の洗い出しと絞り込み

まず、STEP1 は導入支援者がガイドして、メンバ自らが問題点を洗い出しました。ところが、メンバの業務多忙により作業が中断してしまいました。マネージャが関与して再開したものの、メンバだけで問題の因果関係分析は難しく、マネージャが問題を整理・分析して因果関係図をようやく完成させました（図 3-5-2）。

その後、メンバ全員でこの図を基に、問題点の具体的な内容と理由を挙げて問題の関係を確認しました。このとき、さらに掘り下げて因果関係を分析すべきだっ

たのですが、検討時間を確保することが難しく、取り組むべき問題を次の２つに絞り込みました（STEP3）。

- 「朝礼が長い」
- 「スケジュールを立てていない」

「朝礼が長い」は、業務に影響を及ぼす日々の問題であり、解決の優先度が高いものです。また、「スケジュールを立てていない」は、これが原因で作業期限や範囲が曖昧で状況がわからないことが、以前から問題視されていたものです。

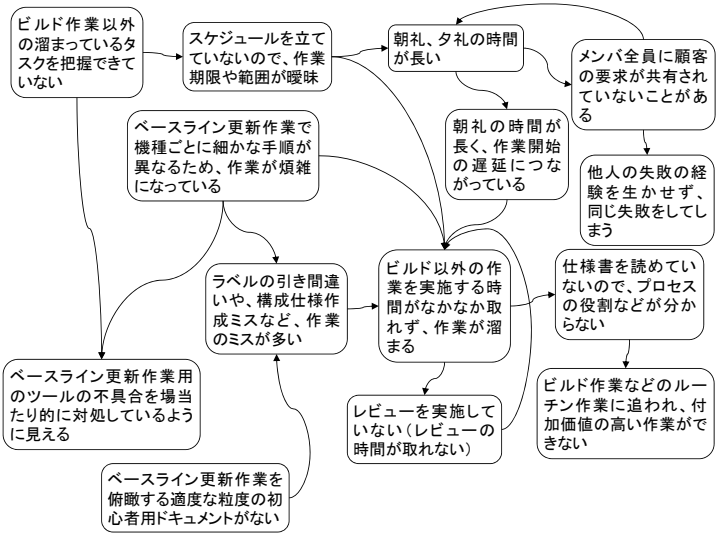


図 3-5-2　ケース１の因果関係図（トラブルモデル）

(2) ヒント集の利用と改善策

「朝礼が長い」および「スケジュールを立てていない」という問題点について、直接該当する業務（プロセス）のヒント集はありませんでした。そこで、ヒントが得られる可能性があると思われた「WS-PMC-01：プロジェクトの進捗を確認する」というヒントシートを選択し、それぞれの問題ごとに「改善検討ワークシート」を作成する担当者を決めました。そして、作成された「改善検討ワークシート」をメンバ全員でレビューしました。

その後、各自が考えてきた改善策について、「すぐに着手できるもの」、そして

「改善効果が比較的得られやすいと思われるもの」の観点から改善策を絞り込んだ結果、次のように、計 7 つの改善策が挙がりました（STEP6）。

- 問題点「朝礼が長い」の改善策
  - 「アジェンダを事前に作成し、朝礼の議題を共有しておく」
  - 「調整が必要だと思う議題は事前にプロジェクトマネージャに相談しておく」
  - 「終了時間を決めておく」
  - 「タイムキーパーを立てる」
- 問題点「スケジュールを立てていない」の改善策
  - 「ビルド以外の作業について WBS を考える」
  - 「スケジュールを作成し、マイルストーンを明確にする」
  - 「スケジュールを使って進捗を確認する」

（3）改善の実施と行動の変化

これらの改善策を改善計画としてとりまとめ、メンバ全員で確認した上で、3 週間にわたって現場で実施してみました（STEP7）。その結果、各改善策について以下のような行動の変化が観察されました。

- 改善策「アジェンダを事前に作成し、朝礼の議題を共有しておく」

  アジェンダのフォーマットを作成した。また、各自の議題を収集し、アジェンダにまとめる係を決めた。その結果、朝礼前までにアジェンダが作成され、朝礼の議題を共有できるようになった。さらに、各自がどのような順番で何について報告するかを事前に整理した上で報告するよう、行動の変化が見られた。
- 改善策「調整が必要だと思う議題は事前にプロジェクトマネージャに相談しておく」

  調整が必要と思われる議題について、事前に各自がプロジェクトマネージャに内容を説明し、相談するようにした。その結果、朝礼は議題の内容と対処などについて全員が共有する場となり、議題について状況確認

や議論を行うことが以前より少なくなった。

- 改善策「終了時間を決めておく」
  アジェンダ作成時に、終了時刻の目安を決めるようにした。
- 改善策「タイムキーパーを立てる」
  朝礼の進行状況を管理する進行役を分担するようにした。その結果、プロジェクトチーム全体で時間を意識し、必須連絡事項ならびに優先度の高い議題から先に確認しようとする傾向が見られた。
- 改善策「ビルド以外の作業について WBS を考える」
  成果物に対して必要な作業を整理するようにした。その結果、作業経験者に具体的な作業内容を確認する姿が見受けられた。
- 改善策「スケジュールを作成し、マイルストーンを明確にする」
  概略スケジュールを作成するようにした。その結果、チェックポイントとなるマイルストーンを共有することができた。
- 改善策「スケジュールを使って進捗を確認する」
  進捗を確認するよう、朝礼のアジェンダを追記した。

(4) ふり返りの実施

3 週間の改善実施後に、チーム全員がふり返りを実施し、良かった点、悪かった点、実施前後の変化などについて話し合いました (STEP8)。その結果、プロジェクトマネージャが「効果が期待できる」と判断し、現在も改善作業を継続して実施しています。

ふり返りで挙がった主な内容は、以下のとおりです。

問題点「朝礼が長い」について

- 良かった点
  - 議題が決まっているので、話題が外れにくくなった。
  - 議事録と、議論にかかった時間の記録が残るようになった。
  - アジェンダがあるので、議事録がとりやすくなった。
  - 時間を守るように意識するようになった。
- 悪かった点
  - アジェンダ作成に時間がとられる。

- 実施前後の変化
  - 朝礼での議論の内容と議論に費やした時間の記録が残るようになった。
  - 朝礼の時間は短くなっていない。

問題点「スケジュールを立てていない」について

- 良かった点
  - 作業詳細を把握していないことに気がついた。
  - スケジュールを作ることで、溢れる作業が把握でき、作業を他の人に振ることができた。
- 悪かった点
  - 大きな作業でない場合は、別途管理しているタスクリストの管理で十分で、時間をかけて WBS やスケジュールを作るまでもない。
- 実施前後の変化
  - 作業の負荷分散ができるようになった。
  - 締め切りを意識するようになった。

（5）考察（活用のヒント）

このケースは、プロセス改善経験の浅いプロジェクトでの適用事例でしたが、基本的にメンバが改善を実施して、本メソッドの研修を受講したプロジェクトマネージャはメンバの支援に回ったことが大きな特徴です。以下に考察をまとめます。

- プロセス改善活動導入のきっかけ作り

  プロセス改善経験が浅い実験参加者がほとんどだったが、改善活動の全体像を把握した上で本メソッドのステップをすべて実施した。本メソッドが改善活動の基本的な手順とシート、そして網羅された問題点リストや改善策検討のヒントを提供することで、改善経験がないメンバでもこれらを参考にしながら、改善活動を実施することができたのではないか。プロセス改善の経験が浅いプロジェクトが、改善活動の一歩を踏み出すきっかけとして本メソッドを活用できると考えられる。

- コミュニケーションの活性化

  このチームは、試行以前には個人で作業をすることが多く、さらに、業務全体に関わる問題やその要因について深く討論する機会が少なかったが、当メソッドを試行することで意見交換や問題領域の共有活動を行うことができ、チーム内のコミュニケーションが活性化される効果が得られた。

- ポジティブ思考

  チームメンバだけでは、因果関係分析を十分に進めることができなかった（経験・スキルの不足）。結果として、マネージャが問題の関連を整理した図を作成したが、問題の現象に近い部分を問題として特定し、実施することになり、問題を因果関係で結びつける材料が不足していた（STEP1 で抽出した問題を詳細化する作業の省略など）ようである。にもかかわらず、一定の効果をメンバが実感し、改善活動を続ける動機づけの効果があった。改善活動とは、トライ＆エラーを繰り返すことで経験・スキルが培われることを念頭に置き、前向きな姿勢で取り組むことが大切なのではないだろうか。

5.2 ケース２：プロセス改善を実施している

このケースは、組込みソフトウェア開発を行う 30 名以上のプロジェクトのうち、テストを担当するチームで３ヶ月間実証実験として試行したものです。

チーム構成は、マネージャ１名、プロジェクトリーダ１名、メンバ２名です。チーム全体の改善への意識は高く、日頃からすでに改善活動を実施しています。このチームは、開発工程に入ると残業や休日出勤が多くなり、まとまった時間が確保できなくなることが多いため、開発の切れ目で比較的チームメンバの時間に余裕がある期間に実施することにしました。

プロジェクトマネージャは、本メソッドの研修に参加して基本的な知識を得た後、本メソッドのガイドブックを使って簡単なガイダンスを開催しました。その結果、チームメンバは全員が本メソッドの作業の概要や流れを理解した上で活動を開始できました。

プロジェクトマネージャの判断により、負担なく実施できる範囲として、2 週間に 1 度（1 ～ 2 時間）程度の割合で進める作業量を見積もり、この範囲内で活動を行うことにしました。STEP ごとの実施経過の概要は図 3-5-3 のとおりです。

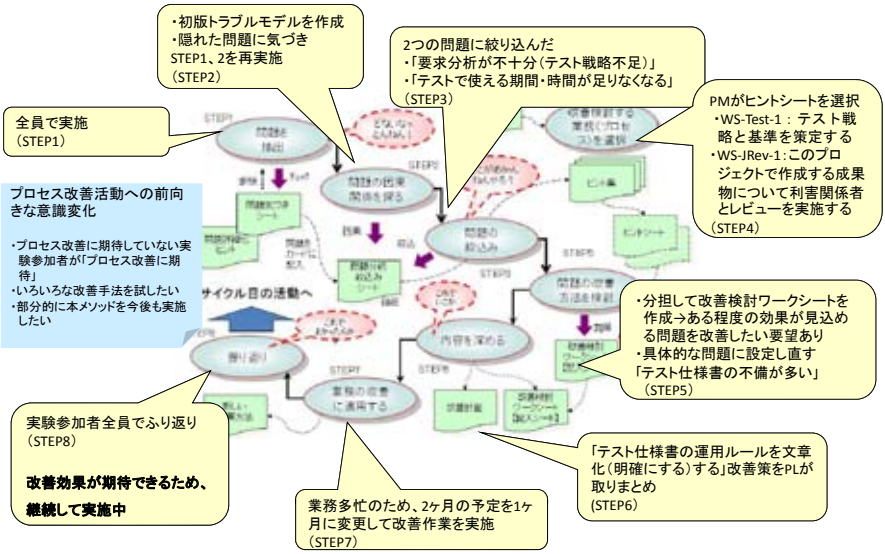


図 3-5-3　ケース 2 の実施経過（STEP1 ～ STEP8）

（1）問題の気づきとトラブルモデルの作成

最初にメンバ全員が「問題気づきシート」と「問題気づきシート（詳細）」を使用して、事前に考えてきた問題を出し合い、個々の内容を確認しました。

問題を整理しつつ、ホワイトボードに付箋紙などを用いて問題と問題とを線でつなぐ作業を進め、その過程で問題の詳細化も行いました。ある程度まとまったところで、プロジェクトマネージャがホワイトボードに描かれた因果関係図をまとめたものが、初版のトラブルモデルです（STEP2）。

図 3-5-4 に問題の因果関係を表したトラブルモデル（初版）を示します。

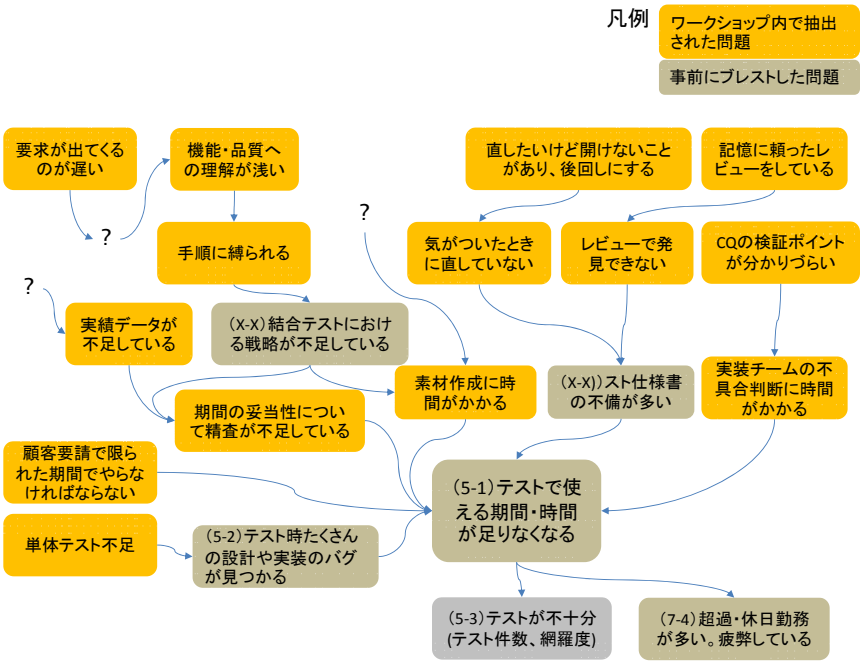


図 3-5-4　ケース 2 のトラブルモデル初版

このトラブルモデルをチームメンバ全員でレビューしたところ、問題と問題がうまくつながらない部分があり、隠れた問題があることが明らかになりました。また、プロジェクトマネージャには因果関係が明確でないという感覚がありました。そこで､改めて問題を抽出して分析する作業をプロジェクトマネージャがチームメンバに提案し、STEP1 と STEP2 を繰り返すことになりました。こうしてできたのが、トラブルモデルの改訂版（図 3-5-5）です。

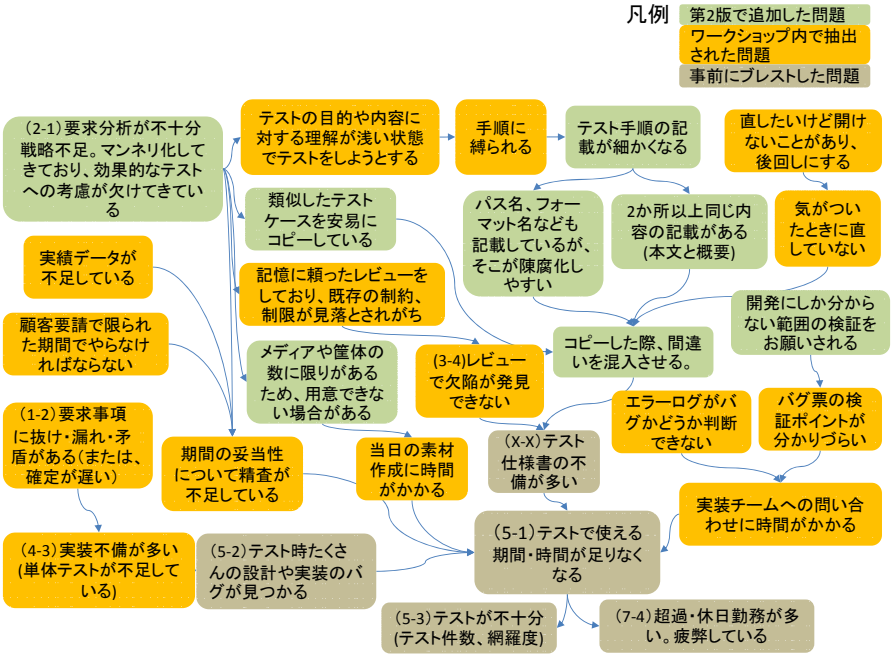


図 3-5-5　ケース 2 のトラブルモデル改訂版

(2) 問題点の絞り込み

このトラブルモデルを基にして、STEP3 ではチームメンバ全員で、以下 2 つの問題点に絞り込みました。

- 「要求分析が不十分（テスト戦略不足）」
- 「テストで使える期間・時間が足りなくなる」

問題点に対する改善対象業務（プロセス）は、プロジェクトマネージャが選択し、「要求分析が不十分（テスト戦略不足）」の問題については、「WS-Test-1：テスト戦略と基準を策定する」を選び、「テストで使える期間・時間が足りなくなる」の問題については「WS-JRev-1：このプロジェクトで作成する作業成果物について利害関係者とレビューを実施する」を選びました。

それぞれの改善対象業務（プロセス）についてメンバで分担し、「改善検討ワークシート」に沿って改善策を作成しました。そして、できあがった「改善検討ワー

クシート」をチームメンバ全員でレビューしながら、改善策の検討を進めました。

しかし、検討の結果、「要求分析が不十分（テスト戦略不足）」という問題点を解決するには顧客を巻き込む必要があり、着手することが困難であることが明らかになったのです。また、「テストで使える期間・時間が足りなくなる」という問題点を解決するには、実験実施計画内で効果が見込めるかどうかに懸念があり、加えて、ある程度の効果が見込める問題に取り組みたいという実験参加者の一致した意見もありました。そのため、「テストで使える期間・時間が足りなくなる」から、さらに具体的な問題を改善の対象とすることにしました。

この結果、STEP2 で作成したトラブルモデルから改善対象を再度絞り込む STEP3 以降を、再び実施することとなったのです。

（3）再度の絞り込み

2 回目の STEP3 では、「テストで使える期間・時間が足りなくなる」という問題のいくつかの原因のうち、最も改善効果が見込めると全員が賛同した、「テスト仕様書の不備が多い」という問題に絞り込みました。

続く 2 回目の STEP4 では、「テスト仕様書の不備が多い」問題に該当する改善対象業務（プロセス）が見当たりませんでした。

そして、2 回目の STEP5 では、テスト仕様書の改編に対するルールが明確でないことを改善すべき点と識別し、新たな改善策として、以下に絞り込むことができました。

- 「テスト仕様書の運用ルールを文章化（明確にする）する」

（4）改善の実施と結果

これを基に、プロジェクトリーダが改善計画をとりまとめ、それを実験参加者全員でレビューしました。専門家との討議は特に行いませんでした。

当初は改善策を 2 ヶ月間実施する予定でしたが、メンバの業務が多忙になったため、1 ヶ月間に短縮して実施しました。実施した内容は、以下のとおりです。

- 問題点「テスト仕様書の不備が多い」
- 改善策「テスト仕様書の運用ルールを文書化する（明確にする）」

- 具体的な施策
  - バージョン管理システムの利用についてルールをまとめた。
  - テスト仕様書初回発行時のルールについてまとめた。
  - 内部ならびに外部レビュー実施後の修正ルールについてまとめた。
  - テスト実施中の修正ルールについてまとめた。
  - 上記のルールをプロジェクト管理環境（Wiki）に公開し、共有した。

その結果、運用ルールが明確になり、チーム全体で共通の認識を持つことができました。ただし、テスト工程は本実験の後に予定されているため、「テスト仕様書の不備が多い」という問題がどこまで改善されるかは、今後確認することになります。

（5）ふり返りの実施

最後に、実験参加者全員でふり返りを実施しました。実験者がファシリテータを務め、良かった点と悪かった点、実施前後の変化などについてまとめた感想は、以下のとおりです。現在も、策定したルールに基づいて改善作業を継続して実施しています。

- 良かった点
  - 曖昧だったルールが明確になった。
  - 多忙になると後回しにしがちな作業がルール化されたため、問題に対して取り組もうという意識が高まった。
  - 所定の場所に閲覧できるようにしたのが良い。最新版が確認できる。
- 悪かった点
  - 特になし
- 実施前後の変化
  - テスト工程が先で効果が見えていないため、期待や意欲がまだ感じられない。

（6）考察（活用のヒント）

このケースは、改善経験が豊富なプロジェクトでの適用例でした。プロジェクトチームが改善経験の豊富なチームであったため、問題解決の手順をプロジェク

トチーム全員が身につけており、かつ、プロジェクトマネージャが改善活動に割り当てる工数をメンバの負担がない形で計画し、それをコントロールしたことで、短時間に効率よく実施することができました。

- 柔軟なステップの運用

  プロジェクトチーム全員が納得できるレベルまで、問題点の抽出と問題の因果分析を繰り返し実施したことで、最適な問題点とそれに対する改善策を導き出すことができたのではないだろうか。このような結果を生み出したのは、問題点の抽出と因果分析を繰り返し行うという決定を下したプロジェクトマネージャの判断力であり、それは豊富な経験に基づくものであると考える。活動の参加者が気づいたことを率直にチームで共有し、必要と判断したら立ち戻るというフレキシブルな行動も厭わないことが、結果として効率的な改善活動につながるのではないかと推察する。

- スコープを広げる

  これまでにプロジェクトチームで実施してきた改善活動は、個々の問題に対する局所的なアプローチがメインであった。業務全体に関わる問題を抽出し、その問題の関連性から根本原因を分析するといったプロジェクト全体を俯瞰したアプローチは実施していなかった。これは、業務が多忙のため、分析の時間が十分に確保できなかったことが理由と考えられる。今回の実験を通して、これまでとは異なる視点から問題に向き合う時間を確保して、プロジェクトチームで共通のトラブルモデルを作成したことが大きな成果の１つだと考える。この手法は､自分たちのスコープ内にとどまらず､視野を広げる結果となる。自分たちの仕事(プロセス)の本来の目的・ねらいを確認し、部分最適に陥らず、全体最適に向けた活動につなげやすいのではないだろうか。

5.3　ケース 3：プロセス改善に精通した人が一人でプロジェクトに適用する

このケースは、組込ソフトウェア開発の追加案件・障害対応を行うチームのプロジェクトマネージャ兼プロジェクトリーダが、一人で行った事例です。前述の 2 つの事例と同じく、3 ヶ月間の実証実験として試行しました。ちなみに、チームメンバはビジネスパートナー 2 名です。

プロジェクトマネージャは、本メソッドの研修に参加して基本的な知識は持っています。また、プロジェクトチームは比較的小規模で、プロジェクトマネージャ兼プロジェクトリーダがプロジェクト状況を把握しています。メンバには特に本メソッドを意識させることなく、改善策の実施に協力してもらっています。プロジェクトマネージャは社内の PMO 活動を推進する立場であり、プロセス改善経験や知識は豊富です。本メソッドを導入することにより、組織としての効率化が図れることを期待して実験に臨みました。実施経過は図 3-5-6 のとおりです。

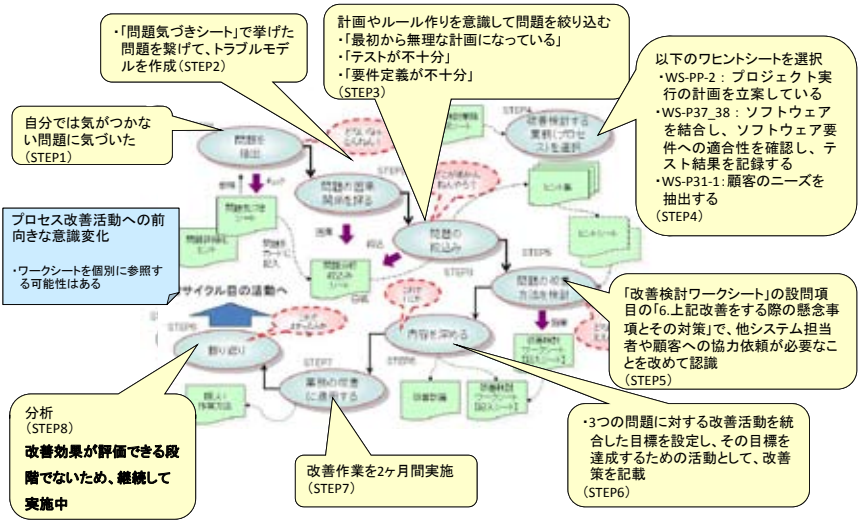


図 3-5-6　ケース 3 の実施経過（STEP1 ～ STEP8）

（1）問題点の抽出とトラブルモデルの作成

まず「問題気づきシート」を使用して、問題にチェックをつけました。次に「問題気づきシート（詳細）」を使い、プロジェクトチームで起きている問題を抽出しました。このほか、以前のプロジェクトメンバが状況をふり返り、洗い出した情報も参考にしました。

そして、抽出した問題点の因果関係を探り、トラブルモデル（図 3-5-7）を作成しました。トラブルモデルに挙げられた問題の数は全部で 14 個あり、そのうち、「問題気づきシート」に記載されている問題は 10 個でした。

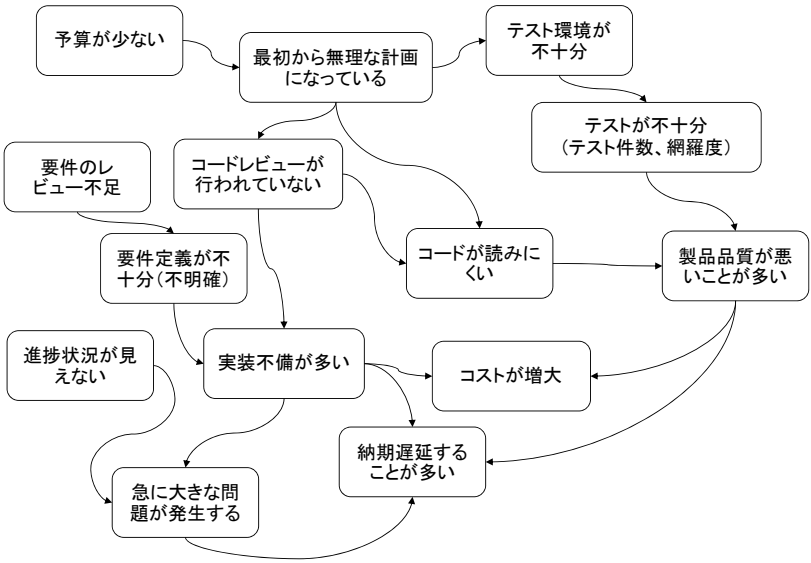


図 3-5-7　ケース 3 の因果関係図（トラブルモデル）

プロジェクト立ち上がり段階だったため、計画やルール作りを意識して、トラブルモデルの中から以下の 3 つを問題点として抽出しました。

- 「最初から無理な計画になっている」
- 「テストが不十分」
- 「要件定義が不十分」

(2) 改善策の決定

「最初から無理な計画になっている」の問題点については、「WS-PP-02：プロジェクト実行の計画を立案している」、「テストが不十分」の問題点については、「WS-P37.38：ソフトウェアを結合し、ソフトウェア要件への適合性を確認し、テスト結果を記録する」、「要件定義が不十分」の問題点については、「WS-P31-1：顧客のニーズを抽出する」といった改善対象業務（プロセス）をそれぞれ選択しました。

そして、問題点ごとに「改善検討ワークシート」を作成し、改善方法を検討しました。「改善検討ワークシート」の設問項目の「6. 上記改善をする際の障害など懸念事項とその対策」では、他システム担当者や顧客への協力依頼が必要なことを改めて認識しました。

ここで導き出した問題点と改善策は以下のとおりです。

- 問題点「最初から無理な計画になっている」の改善策
  - 「プロジェクト全体の計画書、WBS などを作成し、製品全体として品質を担保するにはどのくらい工数がかかるかを明確にする」
  - 「マイルストーンを記述したマスタースケジュールと、個々の作業を管理するスケジュールを区別して管理する」
- 問題点「テストが不十分」の改善策
  - 「テスト計画書を書いて、テストの程度、どのようにテストするか、誰がテストするかなど、ある程度明確にしたい」
- 問題点「要件定義が不十分」の改善策
  - 「口頭で依頼された要求も必ず顧客に要求仕様書を作成してもらう」
  - 「口頭の依頼を不可とし、顧客にメーリングリストに依頼を出してもらう」
  - 「見積りを顧客に提示し合意を得る」
  - 「全体の要求一覧を作成し、詳細ができているところとできていないところを認識する」

「改善検討ワークシート」を使って検討した改善方法を整理して、改善計画は 3 つの問題点に対する改善活動を統合した目標を設定しました。そして、その目標を達成するための活動として、改善策を記載する形にしました。その改善計画

をメンバに説明し、目標を達成するための改善活動への協力を依頼しました。

(3) 改善策の実施と結果

改善策は、約2ヶ月間実施しました。全部で7つの改善策のうち、プロジェクトチーム外の利害関係者が関係する2つの改善策は計画どおりに実施できませんでしたが、その他の5つの改善策は、予定どおりに実施しました。以下に7つの各改善策に対する実施結果を示します。

- 改善策「プロジェクト全体の計画書、WBSなどを作成し、製品全体として品質を担保するにはどのくらい工数がかかるかを明確にする」

  メジャー版プロジェクトからのQA対応や仕様変更に関して、プロジェクト全体の計画と作業工数の見積りを再実施し、作業工数と期間とを明確にした。今後は、開発状況や仕様変更に伴いプロジェクト計画や見積りを詳細化する予定である。
- 改善策「マイルストーンを記述したマスタースケジュールと、個々の作業を管理するスケジュールを区別して管理する」

  概算スケジュールを作成し、マイルストーンを明確にした。
- 改善策「テスト計画書を書いて、テストの程度、どのようにテストするか、誰がテストするかなど、ある程度明確にしたい」

  テストの実施タイミングやテストで確認する内容や程度などを検討し、テスト担当者と調整した。調整結果をテスト計画書にまとめた。
- 改善策「口頭で依頼された要求も必ず顧客に要求仕様書を作成してもらう」

  次回の要求仕様書提示以降、仕様変更について仕様書に作成してもらうよう顧客に要望する予定である。
- 改善策「口頭の依頼を不可とし、顧客にメーリングリストに依頼を出してもらう」

  仕様の変更に関しては、口頭ではなくメーリングリストへの変更依頼を基本的なルールとし、顧客側に要望した。また、変更依頼を仕様書に反映する担当を明確にした。
- 改善策「見積りを顧客に提示し合意を得る」

  次回の仕様変更提示後に再見積りを行い、顧客と合意する予定である。

- 改善策「全体の要求一覧を作成し、詳細ができているところとできていないところを認識する」

  顧客が管理する要求一覧リストに本プロジェクトの要求も管理するよう顧客に依頼し、要求一覧リストに追加された。

(4) ふり返りの実施

良かった点と悪かった点、および改善に関するふり返りは、次のように分析を行いました。特に、顧客が関係する改善策は未実施の部分があるため、絞り込まれた問題の改善効果を評価する段階ではないと判断し、現在も改善作業を継続して実施しています。

以下に示すのが、ふり返りで挙がった主な内容です。

- 良かった点

  選択肢や参考情報があるので、経験がなくてもある程度問題点をピックアップできた。ふり返りまでがセットになっているため、締め切り意識が働き、やり放しになりがちな改善のフォローができた。
- 悪かった点

  紙ベースなので、自分の該当する問題を探すのに非常に時間がかかった。STEP6 以降のフェーズには、あまりフォローがない。
- 改善に関するふり返り

  Keep
  - メンバに改善項目を明確に提示しているため、改善状況を把握しやすい。
  - ふり返りのタイミングを設定しているため、実施確認を怠らない。

  Problem
  - 改善の途中では意識が薄れがちになる。
  - 顧客依存の改善点についてはあまり無理にお願いすることは難しい。
  - プロジェクトの状況に流されてしまいがちである。

  Try
  - 今回はメンバの負担を考え、リーダのみがメソッドを意識して、メンバ以下には改善計画以降の実施だけをお願いした。今後機会があれば、

メソッドの STEP1 からメンバにもお願いしたい。

（5）考察（活用のヒント）

このケースの特徴は、部門 PMO の立場でもあるプロジェクトマネージャが一人で本メソッドの大半を適用したことです。この特徴を踏まえ、実験内容について考察します。

- 気づきの提供

  一人で活用した場合においても、開発全体を俯瞰し網羅的に問題を抽出することができた。また、検討した改善策について異なる観点からのさらなるヒントを得ることもできた。これは、本メソッドが問題点や改善策検討のための知見を網羅的に提供し、気づきを与えたからだと考えられる。

- 時間の効率化

  本実験では参加者が一人であったため、比較的短時間で実施することができた。複数人数による意見交換や、それに関わる議論が発生しなかったためである。時間の制約がある場合は、複数名で構成されるプロジェクトでも部分的に一人で STEP を実行する方法もあるかもしれない。ただし、本実験では短時間で実施しても実質的な内容や質を伴った改善策を得ることができたが、これは参加者が PMO の立場で改善活動に習熟していることが影響している可能性もある。今後は、一人で本メソッドを活用する場合の時間の効率性だけでなく、その実効性について様々なプロファイルを持つ実験参加者による検証を行うとよい。

- サポートの必要度合い

  本実験の参加者は PMO を兼務するプロジェクトマネージャであり、ソフトウェア開発やプロセス改善経験が豊富なため、サポートがなくても本メソッドを適用できた。これは、本メソッドのガイドブック等の参考情報があれば、どのような内容をどのように実施すればよいかを経験から容易に理解でき、実行に移せるからである。経験豊富な場合には、どこにどのような情報があるというポインタを確実に提供するサポートさえあればよい。

5.4　ケース 4：CMMI® レベル 3 到達組織の自律改善に適用する

（1）対象とする組織

このメソッド（SPINA^{3}CH）では、適用する組織あるいはプロジェクトとして、プロセスが未整備か、あるいは、プロセスがあっても形骸化している状況を想定しています。一方で、CMMI® レベル 3 に達成している多くの組織では、プロセスの構築、その維持管理、またはプロジェクトへのプロセスの実装を支援する EPG（Engineering Process Group）組織があることが一般的です。プロジェクトが実施すべきプロセスは整備され、その実施についても、SQA 部門によって確認されています。

しかし、こういった組織では、プロジェクトが組織で定義されたプロセスにそのまま従い、言われたことは確実に実装するものの、その効果的な実装についてはあまり考えることがないことも少なくありません。その理由は、改善活動は EPG 組織の役割であり、プロジェクトは関係ないと考えており、自律的な改善をプロジェクトは放棄しているのではないかと考えられます。

（2）導入の着眼点

当初はこの組織に自律的な改善を導入すべく、SPINA^{3}CH の適用を考えました。ですが、もともと SPINA^{3}CH とは、導入は課題ベース、しかしプロセスモデルの良さも活かしてプロジェクトのプロセスを徐々に構築していくことをめざすものです。すでに、組織としてプロセスを有している状況は想定していません。

しかし、SPINA^{3}CH はプロジェクトあるいはメンバのプロジェクトの課題への気づきを重視し、その気づきを適切なガイドで、自律改善へと向かわせていく手法です。そこで、すでにプロセスを構築している組織に導入できないかどうかを検討しました。CMMI® レベル 3 の組織では、組織のプロセスからテーラリングによってプロジェクトのプロセスを定義することに着目し、その部分にこのメソッド、すなわち自律改善の手法が適用できないかを考えました。

（3）改善の実施

CMMI® レベル 3 の組織では、プロジェクトの開始時にプロジェクトの特性に合わせて、組織のプロセスをテーラリングすることを基本としています。プロセスのテーラリングでは、適切なテーラリングガイドラインに従って、その範囲をある程度制限することが一般的です。しかし、ここでは自律改善のマインドを植え付けるために、プロジェクトのテーラリングの裁量を大きくして、自らのプロセス実施の考えを活かすようにしました。

プロジェクトの定義されたプロセスは、プロジェクトの開始時にそのプロセスが必要な品質を確保できることを確認するために、SQA の承認を得ることが必要です。開発の完了時には、プロジェクトのふり返りを実施しますが、その際に、STEP1 の問題気づきシートを使用して、プロジェクトで起こった問題を洗い出します。

次の STEP2 の問題分析絞り込みシートを使用して、問題の因果関係を探ります。そして、STEP3 の問題の絞り込みを行い、定義されたプロジェクトの定義されたプロセスのどこがまずかったのか、改善点を検討します。改善点およびその内容は、次プロジェクトの定義されたプロセスの検討時のインプットとします。

また、プロジェクトのふり返りで確認された良い点は、組織資産に反映します。これによって、プロジェクトの自律的な改善を進め、組織資産構築への参画を促すことができ、プロセス改善の自律化、そして自立化を実現が可能になります。

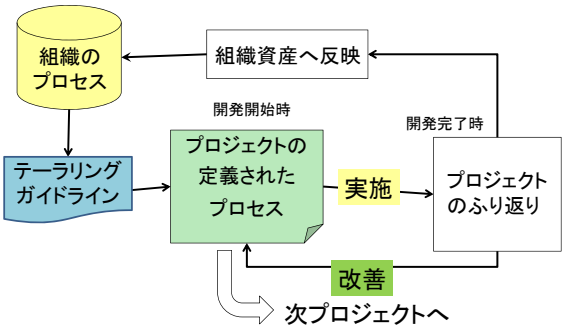


図 3-5-8　すでにプロセスを持つ組織への適用

6.　他の領域への活用

この自律改善メソッドは、ソフトウェアに関する業務の改善のフレームワークを提供するもので、他の関連する多様な領域にも適用できます。第 2 章で適用した開発者向けの業務の他に、たとえば、ソフトウェアの調達・運用を行う業務の改善、およびソフトウェアに関する品質管理活動、さらには組織体のソフトウェアプロセスの全体管理に関わる活動にも適用できます。

そうした他の領域への適用の際は、このガイドで提供しているメソッドの詳細な記述にとらわれず、元となる考えかたを理解して対象分野の特性に合わせ、適切な具体方策を考えるのがよいでしょう。以下に、メソッドの基本的な推進枠組みの考え方を示します。

6.1　改善の出発点

改善は、プロセスにおける何らかの問題意識に基づいて開始されます。問題意識は、現場の人々からも、プロジェクトマネージャからも、顧客サイドからも、また品質管理の人々からも提示されます。何らかのプロセスライフサイクルの意図的な導入や、社会における製品の利用者からも提起されるかもしれません。

改善は、現場の人々が主体的に推進することが望ましいでしょう。ただし、マネジメントの立場からの改善活動への支持は必要です。

6.2　改善の詳細活動

改善に携わるチームおいて必要となるアクティビティ（活動）を列挙すると、次のようになるでしょう。

（改善の組み立て）

・活動 1：プロセスの問題への気づき

ソフトウェアライフサイクルのワークフローにおいて意識される問題点を、チームが自ら収集・整理し、課題がどのような状況となっているのかをできるだけ広い視野で認識します。

・活動２：問題群とそれらの間の関係の分析

チームは、見出された複数の問題点を、抽象論でなく具体的な事象の間の総合的な関係として掘り起し、全体の関係の見取り図を把握します。必要な場合は、新たな問題把握の視点を追加します。総合的に見ると、複数の問題点はけっこう相互に関連していることが分かります。

・活動３：問題の焦点の見極め

活動２で見出された問題点の関連を効果的に解決していくには、どのような点について改善努力を集中していけばよいかを検討します。この際に集中すべき点は、チームから見てやり方を選択でき、改善可能なものでなければいけません。どのような方向から攻めれば効果的な改善が得られそうか考えてみます。

・活動４：　焦点問題に対応するプロセスの把握

活動３で見出された問題改善の焦点が、業務（世の中で言われている「プロセス」）のどの部分にあたるか探索し、特定します。

※第２章では、改善対象業務（プロセス）選択シートを準備して、その中から選択します。業務（プロセス）は 38 に分かれており、それぞれに対応する「ヒントシート」が用意されています。

・活動５：　プロセスの目的、成果、および必要な業務活動の明確化

特定した業務（プロセス）の目的と必要な成果は本来何であるかを考察します。そうした再確認の活動を通じて、チームは、業務（プロセス）の望ましい要素を検討することができます。また、チームは現在の自分たちがどのようにそうした目的と成果を実現しているのかを確認します。

この確認の活動を通じて、きちんと実現できている項目と、足りない項目、いつもつまずく項目などを整理します。

※第２章では、チームの検討作業をサポートするためにヒントシートを提供しています。ここには、業務（プロセス）の典型的な活動などが記載されています。

・活動６:　プロセスの関連ベストプラクティスの学習

世の中のベストプラクティス事例や、プロセスに関するアイデア事例、また提供されている支援ツール機能の事例を学習し、改善検討に活かします。

※第２章のヒントシートでは、ソリューション例を提示しているほか、関連する書籍等の参考情報も提供することで、チームの学習を支援しています。

・活動７：問題点の解消・緩和のための改善策の考察

活動５や活動６の結果を踏まえて､業務（プロセス）の改善の方策を考察します。改善方策は技術的なものも管理的なものもあり得ます。改善策が実現性の上で妥当なものかどうかも検討します。

・活動８：改善策とその現実性・影響の評価・考察

改善のアイデアを、専門的な知識を持つ人や他の関係者の意見も聞きながら具体的なものにしていきます。その際、実際に行うときにどのような影響が出て、どのような他の協力が必要かも検討します。

ここでは、具体的な改善実施のスケジュールと役割分担や改善が成功したかどうかを後で判定するための指標も決めます。

（改善サイクル）

・活動９：改善の進捗の点検

チームは実際の業務で改善されたプロセスを実施し、改善計画の内容が実施されていることを確認します。

・活動１０：改善サイクルの実施

チームは定期的に、または適切な節目を設定して、改善の進捗と効果状況をふり返り、次の改善サイクルとしてどのようにすればよいかをまとめます。

活動１からの改善の仕組みを、適切な調整を伴いながら２度、３度と繰り返し実施します。

・活動 11：プロセス内容とプロセス改善の知識・スキルの蓄積

改善サイクルを実施しながら得られた知識やスキルは、蓄積できるようにします。そのためには、知識、業務で使うテンプレート、業務実施のノウハウなどの蓄積、検索、適用が容易にできるような仕組みを構築します。

6.3 他のプロセスモデルやワークフローの適用

このガイドラインで提示しているワークフローは開発者向けのものです。また基礎としているプロセスのモデルは、SPEAK-IPA モデルや、いわゆる SLCP（Software Lifecycle Process）のモデル等をベースとしています。しかし、SPINA^{3}CH 自律改善メソッドの基本的な考え方は、他の担当者の現場、他のプロセスモデル、他のワークフローでも適用できます。

そのような応用のため、他の領域向けのメソッドの具体的な構築の手順を次に示します。

（1）誰の作業を対象にするのか見極める

開発担当者でなく、システム企画・発注担当者、品質管理担当者、プロジェクトマネジメント担当者、組織の全体のプロセスマネジメントといった業務の担当者の性格を見定め、その立場に向けてメソッドを構築することを決めます。

（2）どのようなプロセスモデルやワークフローを用いるかを決定する

上記の対象者イメージが確定したら、その人たちが実施している、または実施すべきと思われている業務（プロセス）のモデルを見極めます。プロセスモデルというほど確立したものがなければ、模範となりそうなワークフローのモデル化を利用します。

（3）そのプロセスモデルやワークフローで、どのような実施上の問題点が語られているかデータを収集する

見定めたプロセスモデルやワークフローモデルを主要区分項目に分解し、図示します。これらの主要区分項目において、日常業務実施上で現場的に語られる問題点、至らない点、嫌な点などをできるだけ集めます。

(4) どのような解決法が提案されているかデータを収集する

上記の（2）で見定めたプロセスモデルやワークフローモデルの主要区分項目ごとに、その本来めざすべき目的、求められる成果項目、プロセス（ワークフロー項目）が実施されなかった時に起こるであろうトラブル事例などを、プロセスモデル等に基づいて整理します。

また、プロセスモデルやワークフローモデルの主要区分項目ごとに、語られているベストプラクティス例やアイデア例をできるだけ広く収集します。

(5) メソッドツールとしての構築

本書で提示しているメソッドのツールを次のように書き換えていきます。

- 本書で提供しているメソッドのワークフローを（2）で考察したワークフローに置き換え、問題気づきシートの項目を、そのワークフローの項目と（3）で検討した「実施上の問題点」に置き換えます。
- 問題点カードの各項目を、（3）で検討した「実施上の問題点」に置き換えます。
- 改善検討業務選択シートを、(2) および (3) に対応するものに置き換えます。
- （4）で調べた内容を、プロセス改善検討ワークシートのヒント集としてまとめ、置き換えます。

第 4 章　道具編

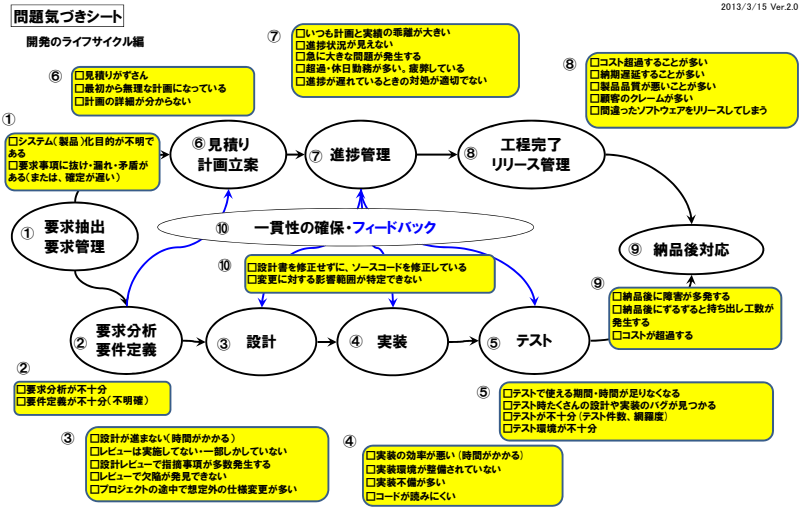
本章は、ここまでに説明してきた SPINA^{3}CH 自律改善メソッドの各 STEP で使用するシート類を収録しています。これらのファイルは下記からダウンロードできますので、有効にご活用ください。

http://sec.ipa.go.jp/std/ent02-b.html

1. 問題気づきシート
2. 詳細化ヒント
3. 問題点カード
4. 改善検討業務選択シート
5. 業務（プロセス）一覧
6. 改善検討ワークシート（記入シート）
7. 改善検討ワークシート（ヒント集）
8. 改善計画・実績シート、ふり返りシート

1. 問題気づきシート

このシートは巻末の折り込みを参照してください。



2. 詳細化ヒント

■開発のライフサイクル編

1.要求抽出・要求管理(顧客要求の明確化)

a)システム化目的が不明である

□	顧客要求の引き出し不足
□	顧客と開発チームの間で顧客要求に対するお互いの理解不足
□	使用されるシチュエーションが分からない

b)要求事項に抜け・漏れ・矛盾がある(または、確定が遅い)

□	顧客要求の引き出し不足
□	顧客要求が文書化できていない
□	顧客要求のプロジェクトへの説明不足
□	顧客とプロジェクトの間で顧客要求に対するお互いの理解不足
□	顧客要求をまとめる人がいない
□	顧客が要求を表現できない
□	顧客要求の矛盾や不明点がある
□	顧客要求が承認されていない(顧客側)
□	顧客要求が承認されていない(顧客と開発チーム間)

2.要求分析・要件定義

a)要求分析が不十分

□	要求分析のやり方が分からない
□	要求の妥当性のレビューが開発チーム内でされていない
□	要件定義の仕組みがあってもメンバーに教育がされていない

b)要件定義が不十分(不明確)

□	顧客要求との一貫性がない
□	要求分析のやり方が分からない
□	要件が文書化されていない
□	要件定義の方法が個人に依存している
□	機能要件のみで非機能要件が抽出されていない
□	妥当性確認ができていない
□	顧客と合意されていない
□	開発の終盤に要件の変更が頻発する

3.設計

a)設計が進まない(時間がかかる)

☐	設計手法が確立されていない
☐	設計手法の教育(説明)がされていない
☐	要件定義が不十分
☐	設計するための規格や実機環境の情報が不十分または入手できない

b)レビューは実施してない・一部しかしていない

☐	レビューの仕組みがない
☐	メンバーにレビューの教育がされていない
☐	レビューの仕組みがあっても守られていない
☐	レビューの効率が悪い
☐	時間がない、リーダが認めない、雰囲気がない、発想がない

c)設計レビューで指摘事項が多数発生する

☐	設計者のスキルが足りない
☐	設計手法が確立されていない
☐	レビューの方法がふさわしくない
☐	レビューの観点が明確でない

d)レビューで欠陥が発見できない

☐	レビューの方法がふさわしくない
☐	レビューアーのスキル不足
☐	レビューの準備不足
☐	レビュー対象の成果物の完成度が低い（指摘も出ない）

e)プロジェクトの途中で想定外の仕様変更が多い

☐	顧客要求の引き出し不足
☐	顧客要求との一貫性がない
☐	要求分析のやり方が分からない
☐	要件が文書化されていない
☐	要件定義の方法が個人に依存している
☐	機能要件のみで非機能要件が抽出されていない
☐	妥当性確認ができていない
☐	顧客と合意されていない
☐	仕様変更の制御がされていない
☐	再計画がされていない
☐	顧客の体制が変わる
☐	仕様変更が通知されない

4.実装

a)実装の効率が悪い(時間がかかる)

☐	実装手法が確立されていない
☐	実装手法の教育(説明)がされていない
☐	設計が不十分

b)実装環境が整備されていない

☐	デバッガの機能が低い
☐	ノウハウが共有されていない
☐	ツールの使い方が分からない
☐	コーディング規約がない
☐	静的検査ツールがない

c)実装不備が多い

☐	仕様書が不明瞭
☐	実装技術・知識が不足

d)コードが読みにくい

☐	個人ごとにコーディング・スタイルが違う
☐	コーディング規約が曖昧
☐	コーディング規約が守られていない

5.テスト

a)テストで使える期間・時間が足りなくなる

☐	テスト手法が確立されていない
☐	テスト手法の教育(説明)がされていない
☐	テスト工期の見積りが甘い
☐	再計画が実施されていない
☐	バグが収束しない
☐	テストの基準がない
☐	バグ修正の手順が確立されていない
☐	バグ修正の手順が教育(説明)がされていない

b)テスト時たくさんの設計や実装のバグが見つかる

☐	設計のレビューが不十分
☐	インターフェース仕様に誤りがある
☐	単体テストが不十分

c)テストが不十分(テスト件数、網羅度)

☐	テスト手法が確立されていない
☐	テスト手法の教育(説明)がされていない
☐	判定基準が甘い
☐	テストケースの洗い出しが不十分
☐	時間が足りなくなる

d)テスト環境が不十分

☐	試作機の不足、故障
☐	テスト環境(ツールや試験環境)の不足
☐	運用環境と試験環境とのバージョンの不一致
☐	テストデータの不足
☐	本番環境での不具合データ(ログなど)が取得できない

6.見積り計画立案

a)見積りがずさん

□	要件が十分に抽出されていない
□	見積り方法が不適切
□	見積りの妥当性確認ができていない
□	見積りのスキルが足りない
□	予測できる変更の見積りができていない
□	見積り手法の精度が悪い
□	納入物の定義が曖昧
□	作業の見積りがされていない(WBS)
□	顧客との分担が明確でない
□	テスト工程の見積りが甘い(顧客との意識のずれ、統合テストが長引く)

b)最初から無理な計画になっている

□	顧客と計画に対して合意がとれていない
□	計画が承認されていない
□	見積りが不十分
□	契約上の見積りが技術的な見積りと一致していない

c)計画の詳細が分からない

□	計画の粒度が粗い
□	納期からのみ導いた計画
□	スケジュールのみの計画になっている
□	マイルストーンが不明確
□	リリースするものが分からない
□	リソースの投入計画が不明確
□	体制と役割が不明確になっている
□	計画が文書化されていない
□	関係者間で計画の合意がとれていない（伝わっていない）
□	計画が更新されていない

7.進捗管理

a)いつも計画と実績の乖離が大きい

□	計画が不適切
□	進捗管理が再計画に結びつかない
□	計画に応じたリソースが投入されない
□	乖離の分析や対策が実施されていない

b)進捗状況が見えない

□	見ていない・報告していない
□	計画が不十分
□	進捗管理の尺度が適切ではない
□	実績を測定していない
□	進捗状況が関係者間で共有されていない

c)急に大きな問題が発生する

☐	進捗が見えていない
☐	課題の認識違い
☐	言いづらい雰囲気
☐	仕様と異なるものを作ってしまった
☐	リスクが認識されてない
☐	進捗管理の粒度が粗い
☐	大型プロジェクトの場合の情報伝達の不備
☐	要員の問題
☐	ツールや部品の供給元の事故

d)超過・休日勤務が多い。疲弊している

☐	計画が適切ではない
☐	仕様変更が多い
☐	作業分担が不適切
☐	リソース不足
☐	プロジェクト外作業が多い
☐	環境が整っていない
☐	作業の手戻りが多い
☐	要員の能力と難易度のミスマッチ

e)進捗が遅れているときの対処が適切でない

☐	計画変更が不適切
☐	プロジェクトリーダの情報・権限不足
☐	計画変更の手順がない

8.工程完了・リリース管理

a)コスト超過することが多い

☐	成果物の品質が悪い
☐	見積りが甘い
☐	適切な時期に再計画がされていない
☐	計画が不適切
☐	プロジェクトの制御ができていない
☐	問題を見過ごしている
☐	適切な是正措置が取られていない
☐	顧客要求の引き出し不足
☐	新技術投入に対する計画が甘い

b)納期遅延することが多い

☐	顧客の受入れ準備ができていない
☐	成果物の品質が悪い
☐	見積りが甘い
☐	適切な時期に再計画がされていない
☐	計画が不適切
☐	プロジェクトの制御ができていない
☐	問題を見過ごしている
☐	適切な是正措置が取られていない
☐	顧客要求の引き出し不足
☐	新技術投入に対する計画が甘い

c)製品品質が悪いことが多い

☐	設計品質が悪い
☐	レビュー時間の確保が不十分
☐	レビュー能力の不足
☐	テスト設計不足
☐	品質基準が不明確
☐	工程開始基準が不明確
☐	工程完了基準が不明確
☐	品質基準が不明確
☐	テスト環境が不十分
☐	テスト実施が不十分
☐	回帰テストを実施していない
☐	デグレーションが起こっている
☐	工数不足
☐	成果物の管理が不十分
☐	版管理ができていない
☐	ベースラインの管理ができていない
☐	製品が使いづらい
☐	要件の妥当性確認が不十分
☐	バグ修正の手順が確立されていない

d)顧客のクレームが多い

☐	品質が悪い
☐	納期が遅れる
☐	コスト超過
☐	顧客・エンドユーザのニーズに合わない
☐	クレーム対応が悪い
☐	利用者マニュアルが分かりにくい

e)間違ったソフトウェアをリリースしてしまう

☐	ライブラリー管理が不十分
☐	納期優先で、テスト不十分のままリリースしている
☐	リリース作業でのミスが多い

9.納品後対応

a)納品後に障害が多発する

☐	チームが解散してしまい、対応できない。
☐	成果物の所在が不明になる
☐	想定されていない利用形態やデータ量による問題発生

b)納品後にずるずると持ち出し工数が発生する

☐	品質が悪い
☐	顧客と要件が十分に合意できていない
☐	仕様変更が発生する

c)コストが超過する

☐	品質が悪い
☐	顧客と要件が十分に合意できていない

10. 一貫性の確保・フィードバック

a)設計書を修正せずに、ソースコードを修正している

☐	ソースコードしかない
☐	設計書を修正する仕組みがない
☐	変更時の対応が個人任せ
☐	変更が管理されていない

b)変更に対する影響範囲が特定できない

☐	変更時のレビューがされていない
☐	変更が個人任せ
☐	変更範囲を特定する資料がない(トレーサビリティマトリクスなど)
☐	設計が複雑で影響範囲の特定が網羅できていない
☐	保守性を考慮した設計になっていない

3.　問題点カード

□システム（製品）化目的が不明である
1-1

□レビューは実施してない・一部しかしていない
3-2

□要求事項に抜け・漏れ・矛盾がある（または、確定が遅い）
1-2

□設計レビューで指摘事項が多数発生する
3-3

□要求分析が不十分
2-1

□レビューで欠陥が発見できない
3-4

□要件定義が不十分（不明確）
2-2

□プロジェクトの途中で想定外の仕様変更が多い
3-5

□設計が進まない（時間がかかる）
3-1

□実装の効率が悪い（時間がかかる）
4-1

□実装環境が整備されていない

4-2

□テストが不十分(テスト件数、網羅度)

5-3

□実装不備が多い

4-3

□テスト環境が不十分

5-4

□コードが読みにくい

4-4

□見積りがずさん

6-1

□テストで使える期間・時間が足りなくなる

5-1

□最初から無理な計画になっている

6-2

□テスト時たくさんの設計や実装のバグが見つかる

5-2

□計画の詳細が分からない

6-3

□いつも計画と実績の乖離が大きい

7-1

□コスト超過することが多い

8-1

□進捗状況が見えない

7-2

□納期遅延することが多い

8-2

□急に大きな問題が発生する

7-3

□製品品質が悪いことが多い

8-3

□超過・休日勤務が多い。疲弊している

7-4

□顧客のクレームが多い

8-4

□進捗が遅れているときの対処が適切でない

7-5

□間違ったソフトウェアをリリースしてしまう

8-5

□納品後に障害が多発する

9-1

□納品後にずるずると持ち出し工数が発生する

9-2

□コストが超過する

9-3

□設計書を修正せずに、ソースコードを修正している

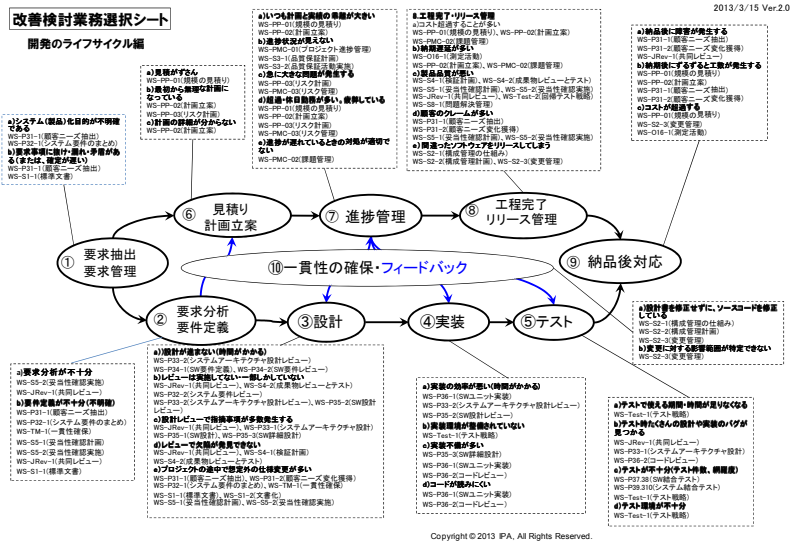
10-1

□変更に対する影響範囲が特定できない

10-2

4.　改善検討業務選択シート

このシートは巻末の折り込みを参照してください。



5.　業務（プロセス）一覧

(1)要求事項抽出
　WS-P31-1(顧客ニーズ抽出):顧客のニーズを抽出する
　WS-P31-2(顧客ニーズ変化獲得):顧客のニーズの変化を確実に取り入れる仕組みを確立する
(2)システム要求分析
　WS-P32-1(システム要件のまとめ):顧客要求からシステム要件にまとめ直す
　WS-P32-2(システム要件レビュー):システム要件の適切性を確認する
(3)システムアーキテクチャ設計
　WS-P33-1(システムアーキテクチャ設計):システム要件を満たすシステムアーキテクチャ設計を実施する
　WS-P33-2(システムアーキテクチャ設計レビュー):システムアーキテクチャ設計の適切性を確認する
(4)ソフトウェア要求分析
　WS-P34-1(SW 要件定義):システム要件からソフトウェア要件にまとめ直す
　WS-P34-2(SW 要件レビュー):ソフトウェア要件の適切性を確認する

(5)ソフトウェア設計
WS-P35-1(SW 設計):ソフトウェア要件を満たすソフトウェア設計を実施する
WS-P35-2(SW 設計レビュー):ソフトウェア設計の適切性を確認する
WS-P35-3(SW 詳細設計):ソフトウェア設計を基にソフトウェア詳細設計を実施する
(6)ソフトウェア構築
WS-P36-1(SW ユニット実装):ソフトウェアユニットを実装する
WS-P36-2(コードレビュー):ソフトウェアユニットを検証する
(7)ソフトウェア結合とテスト
WS-P37.38(SW 結合テスト):ソフトウェアを結合し、ソフトウェア要件への適合性を確認し、テスト結果を記録する
(8)システム結合とテスト
WS-P39.310(システム結合テスト):システムを結合し、システム設計への適合性を確認し、テスト結果を記録する
(9)一貫性の確保／レビュー／テスト
WS-TM-1(一貫性確保):顧客要求からプロジェクトで作成する作業成果物、納入成果物までの一貫性を確保する
WS-JRev-1(共同レビュー):プロジェクトで作成する作業成果物について利害関係者とレビューを実施する
WS-Test-1(テスト戦略):テスト戦略と基準を策定する
WS-Test-2(回帰テスト戦略):回帰テストの戦略を策定する
(10)文書化／構成管理
WS-S1-1(標準文書):作成すべき標準文書と記載すべき内容を明確にする
WS-S1-2(文書化):必要なドキュメントを作成し、適切な時期に関係者が利用できるようにする
WS-S2-1(構成管理の仕組み):構成管理の組織方針や仕組みがある
WS-S2-2(構成管理計画):プロジェクトで作成する作業成果物、納入成果物と作成時期が決まっている
WS-S2-3(変更管理):プロジェクトで作成する作業成果物、納入成果物の変更管理がされている
(11)品質保証
WS-S3-1(品質保証計画):品質保証の仕組みがあり、計画されている
WS-S3-2(品質保証活動実施):品質保証活動が実施されている
(12)検証／妥当性確認
WS-S4-1(検証計画):プロジェクトで作成する作業成果物の検証に関する組織方針や基準を策定している
WS-S4-2(成果物レビューとテスト):プロジェクトで作成する作業成果物のレビューやテストを実施する
WS-S5-1(妥当性確認計画):妥当性確認の組織方針や基準を策定する
WS-S5-2(妥当性確認実施):妥当性確認(顧客が本来やりたいことが実装できているかを確認すること)を実施する
(13)問題解決
WS-S8-1(問題解決管理):プロジェクトで起きた問題について系統的に対処する
(14)プロジェクト計画プロセス
WS-PP-01(規模の見積り):プロジェクトの作業範囲に基づき規模を見積もっている
WS-PP-02(計画立案):プロジェクト実行の計画を立案している
WS-PP-03(リスク計画):プロジェクトのリスクを特定し、分析し、対応策を決定する
(15)プロジェクトアセスメントおよび制御プロセス
WS-PMC-01(プロジェクト進捗管理):プロジェクトの進捗を確認する
WS-PMC-02(課題管理):プロジェクトで起こった問題の是正措置を行う
WS-PMC-03(リスク管理):リスクの状況を監視し、軽減策を実行する
(16)測定
WS-O16-1(測定活動):測定活動が実施されている

6.　改善検討ワークシート（記入シート）

改善検討ワークシート（記入シート）

(WS　　－　　)
1.　目的と成果目標
(1)取り上げた業務(プロセス)の目的
※ヒントシートから該当項目をコピーする(または新規作成)
(2)これを首尾よく達成すると何が実現できるか(めざす成果目標)
※ヒントシートから該当項目をコピーする(または新規作成)
2. この業務(プロセス)課題について、現在行っているのは、どのような方策か
※現在行っていることを簡潔にまとめる
3. 業務(プロセス)の現在のやり方の良い点を書いてください
※現在の方策がうまくいっている内容を記載する
4. 業務(プロセス)の現在のやり方の良くない点を書いてください
※良くない点で改善ができそうなものを記載する。STEP2、STEP3の結果を引用する

※本シートは必要に応じて枠を拡大して使用する

日付		担当者	
5. この良くない点をどのように改善したいか			
※たとえばヒントシートを参照して、考えてみてください			
6. 上記改善をする際の障害など懸念事項とその対策			
※制約事項、コスト、他への影響、協力者、支援要望など			
7. ふり返りの際の改善進捗の概要評価			
※たとえば、達成できたところ、無理だったところ、別の考察が必要なところ			

改善検討ワークシート（記入シート）　　＊改善実施の継続記入シート

(WS　　－　　)
1．目的と成果目標
(1)取り上げた業務(プロセス)の目的
※ヒントシートから該当項目をコピーする(または新規作成)
(2)これを首尾よく達成すると何が実現できるか(めざす成果目標)
※ヒントシートから該当項目をコピーする(または新規作成)
2. この業務(プロセス)課題について、現在行っているのは、どのような方策か
※現状の現実を簡潔にまとめる
3. 業務(プロセス)の現在のやり方の良い点、良くなった点を書いてください
※改善実施後どのような良い点があったかを記載
4. 業務(プロセス)の現在のやり方の、まだ良くない点を書いてください
※継続改善の視点でさらに改善ができる点を記載してください

※本シートは必要に応じて枠を拡大して使用する

日付		担当者	
5. この良くない点をどのように改善したいか（継続改善アイデア）			
※たとえばヒントシートを参照して、具体方策を考えてみてください			
6. 上記改善をする際の障害など懸念事項とその対策			
※改善実施の際の制約事項、コスト、他への影響、協力者、支援要望など			
7. ふり返りの際の改善進捗の概要評価			
※たとえば、達成できたところ、無理だったところ、別の考察が必要なところ			

7.　改善検討ワークシート（ヒント集）

(WS-P31-1 顧客ニーズ抽出)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

顧客のニーズを抽出する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○顧客要求が明確になる。

○想定しない仕様変更の発生が少なくなる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○顧客の期待やニーズ、要求の入手と記録・顧客へのインタビュー、定例会議、情報共有等により、顧客の期待やニーズ、要求を把握し、記録する。

・入手した要求ごとに番号等の識別子を付与し、要求項目と要求概要をまとめた要求一覧表を作成する。

○顧客要求の明確化（顧客の期待やニーズの分析）

・顧客の断片的な期待、ニーズ、要求情報を分析し、顧客要求全体を網羅的に明確化する。

・入手した期待、ニーズ、要求情報の過不足、曖昧さ等は問い合わせ票にて顧客に確認し、顧客要求を確定させる。

○顧客要求のとりまとめと確認

・明確化した顧客要求を要求一覧表にとりまとめ、顧客と共に顧客要求として過不足、不整合がないかを確認する。

・顧客や関係部門とレビューイベントを開催し、結果をとりまとめた要求一覧表を確定させ、顧客の合意をもらう。

3．現実には実現することが難しい点

○顧客から要求を抽出するための決定的な手法は確立されておらず、人対人のコミュニケーションを重視した対応を行う。また、顧客がすべての要求を漏れなく提示するのは難しいため、顧客から要求を得るのではなく、積極的に“引き出す”対応が必要である。さらに、はじめからすべてのことを明確化するのは困難であるため、段階的に引き出し、詳細化していくことが求められる。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○顧客の期待やニーズ、要求が曖昧、過不足あり、不整合な状態のまま開発を進めることで、開発中に仕様変更（追加要求、変更要求）が多発する。

5.1 ソリューション例（取り組みやすい例）

○顧客ニーズ、要求を明確化する上でのキーマンを明確化する。
○基本的な確認事項を明確にした質問票等を活用して顧客の期待やニーズ、要求を把握する。
・システム化の目的（達成したい事項）、解決したい問題点
・システムの利害関係者を含めた利用状況
・制約条件（期間・コスト・品質レベル、など）
・関係する法規制
○顧客の期待、ニーズ、要求事項の確定、変更管理方法を明確化し、顧客と合意する。
・顧客を含めた関係者によるミーティングを定例化する。
・把握した顧客の期待、ニーズ、要求は文書化し、顧客を含めた関係者でレビューを行う。
レビューを通じて、抽象的な内容は具体化し、矛盾する内容は解消する。
・変更要求はその重要度や影響度等を評価後に双方協議して実施判断する。
○確定した顧客要求にはベースラインを設定し、以降は変更管理を行う。

5.2 ソリューション例（深く取り組む例）

○営業担当等が構築したペルソナ（典型的な利用者像）からタスクシナリオ・運用シナリオを構築してニーズ、要求の内容とその満足レベルを把握する。
○プロトタイプ（ペーパープロトタイプを含む）等により明示されたニーズや要求だけではなく、暗黙のニーズ、要求を引き出す。
○ QFD（品質機能展開）により、顧客の期待やニーズを該当するソフトウェア品質特性に位置づけ、顧客と共にレビューを行う。
○他の対応方法：ブレーンストーミング / 市場調査 / 顧客満足度調査 / ベータテスト / 事業事例分析

6．関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-2(顧客ニーズ変化獲得)
・WS-TM-1(一貫性確保)
・WS-JRev-1(共同レビュー)
・WS-PP-01(規模の見積り)
・SPEAK-IPA：P.3.1 要求事項抽出
・CMMI® Ver1.2:：要件開発 (RD)
・共通フレーム 2007：1.4 企画プロセス、1.5 要件定義

7．ふり返り等で参考となる文献等

・ソフトウェア要求：日経 BP 社　Karl.E.Wiegers (著)、渡部 洋子 (翻訳)
・ライト、ついてますか：ドナルド・C・ゴース、G.M. ワインバーグ著 木村 泉訳 共立出版

(WS-P31-2 顧客ニーズ変化獲得)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

顧客のニーズの変化を確実に取り入れる仕組みを確立する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○顧客の期待やニーズ、要求の変化に的確に対応できる。

○顧客要求の状況や変化について関係者の認識を共有できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○顧客のニーズの継続監視

・顧客との共同レビュー、定例会議、情報共有（要求一覧表を含める）等により、ニーズの変化を把握する。

○変更要求の記録と管理台帳による管理

・顧客やハードウェア部門など外部から仕様や設計などの変更要求が発生した場合、内容を確認し、変更要求票・管理台帳に記入する。

○変更による影響分析・評価

・変更要求に対する関連成果物（仕様・設計内容）や作業への影響を詳細に分析評価する。

○変更実施判定

・プロジェクトの状況や変更の影響度、重要度を考慮して変更の実施（あるいは保留･却下等）を判断する。

○要求一覧表の更新

→このあと“変更計画立案と実施”は「プロジェクト管理」にて行う。

3．現実には実現することが難しい点

○顧客のニーズを獲得する仕組みや要件を管理する活動はプロジェクト開始時に実施することが多いが、開発ライフサイクル全般にわたり変化を捉えながら実施・保守し続けることがおろそかになる場合が多い。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○プロジェクトの進行過程で発生する変更要求に適切に対応できない。

その結果、要求～設計～実装の一貫性が維持できない、品質悪化、手戻りの発生、納期遅延、コスト超過、顧客不満足などの不具合事象が発生する。

5.1 ソリューション例（取り組みやすい例）

○当初設定し、顧客を含めた関係者で合意した顧客の期待、ニーズ、要求の明確化方法と変更管理方法に従い、開発期間全般にわたりニーズの全体状況と変化を確実に捉える。

・顧客との定期的な打ち合わせの実施

・重要な成果物のレビューに顧客側担当者を加える。

・要求を変更する場合は、あらかじめ定めた手続きに沿って対応する。

・要件管理や課題管理の情報 (Excel など) を顧客側と共有する。

5.2 ソリューション例（深く取り組む例）

○変更要求に基づき、当初構築したタスクシナリオ・運用シナリオを変更し、影響度を把握する。

○変更要求に基づき、当初構築した QFD（品質機能展開）を変更し、影響度を把握する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-1(顧客ニーズ抽出)

・WS-TM-1(一貫性確保)

・SPEAK-IPA：P.3.1 要求事項抽出

・CMMI® Ver1.2：要件管理 (REQM)

・共通フレーム 2007：1.3 契約の変更管理

・ESPR[1] 2.0：SUP7 変更管理

7. ふり返り等で参考となる文献等

1 ESPR：組込みソフトウェア向け開発プロセスガイド ：ソフトウェア・エンジニアリング・センター

(WS-P32-1 システム要件のまとめ)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

顧客要求からシステム要件にまとめ直す。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○顧客要求が明確になっている。
○顧客要求が実現可能になっている。
○システムの要求及び要件が整理されている。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○入手した顧客要求の内容を確認する。
○顧客要求以外の要求（たとえば内部関係部門からの要求や法規､業界基準など）を引き出す。
○機能要求以外の非機能要求（性能や保守性、セキュリティなど）を引き出す。
○入手した顧客要求と新たに引き出された要求を合わせる (一覧表)
○要求を具体的な仕様に変換し、システムで実現できる要件としてまとめる。
○要件を実現する開発手法やツールを明確にする。

3．現実には実現することが難しい点

○顧客と開発側でお互いに要求を共有できない。
○顧客から一方的に要求が出される。
○非機能要件（性能や運用効率、使用容易性、信頼性、セキュリティなど）が十分に定義されていない。
○分析の決まった方法論が少なく、経験に頼る場合が多い。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○非機能要件が明確に定義されていないため、保守性を考慮した設計になっていない。
○顧客要求がシステム要件に変換されていないため、実装のもれが発生している。
○システム要件に矛盾があるため、顧客のニーズを実現できない。
○システムの分析時に顧客要求の曖昧性が露呈する。

5.1 ソリューション例（取り組みやすい例）

○顧客以外からの要求を引き出す。

・顧客からの要求だけでなく、他部門（たとえばハードウェア部門や営業など）からの要求を引き出す。

・システムの性能やシステムからの制約、再利用性、保守性の観点からも要求を引き出す。

○顧客要求は一般的に顧客の用語で記載されていることが多く、それらをシステムで実現できるように技術的な用語に変換する。

・要求ごとに具体的に仕様化した要件定義書を作成する。

5.2 ソリューション例（深く取り組む例）

○要求管理ツールを導入する。

・要求を一元的に管理するため、商用の要求管理ツールを導入する。

・入力された要求と関連する他の作業成果物の情報も入力すると、それらの一貫性を保つことができる。

○非機能要求を確実に引き出す。

・品質特性（JIS X 0129）を活用して、非機能要求を明確にする。

○顧客のニーズを実現し、加えて、新たな（代替案も含めて）提案を行う。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-1(顧客ニーズ抽出)

・WS-P32-2(システム要件レビュー)

・WS-Test-2(回帰テスト戦略)

・WS-PP-01(規模の見積り)

・SPEAK-IPA：P.3.2 要求分析

・CMMI® Ver1.2：要件管理、要件開発、技術解 (TS)

・共通フレーム 2007：1.6.2 システム要件定義

・ESPR 2.0：SYP1 システム要求定義

7. ふり返り等で参考となる文献等

(WS-P32-2 システム要件レビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

システム要件の適切性を確認する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ビジネスの視点 (新規性、事業性）から見てシステム要件が適切であることを確認できる。
○システムの前提（動作環境や動作条件、コンテキスト、対象ユーザ）とシステム要件に矛盾がないことを確認できる。
○顧客とシステム化のベースラインを合意して、利害関係者の契約の基本文書を構築できる。
○抽出された顧客要求と定義されたシステム要件の一貫性を確認でき、かつシステム要件がテストで検証可能であることが確認できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○システム要件に矛盾がない（正確性）ことを確認する。
○システム要件がテストで検証可能であること（テスト計画性）を分析する。
○運用シナリオを作成し、システム要件が運用環境に与える影響を検討する。
○システム要件の適切性を確認するために実施した活動および作業成果物について、関係者とレビューを行う。
○適切性が確認されたシステム要件を承認する。
○顧客や関係者から入手した顧客要求、確立したシステム要件、システムテストシナリオの一貫性を確認しベースライン化する。

3．現実には実現することが難しい点

○システム要件の適切性を評価する基準を作成するのは難しい。
○システムの非機能要件（性能、セキュリティ、品質など）はまだ明確になっておらず、確認を忘れがちである。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○予測できない要求や技術的課題などのリスクが開発中に起きる。
○システムで実現すべき機能 / 非機能面などの要求事項を関係者に周知することができない。
○システム要件をレビューやテストで、活用することができない。
○未決定事項を明確に確認できない。

5.1 ソリューション例（取り組みやすい例）

○要求の適切性や妥当性を確認するために、運用環境や利用のシナリオなどを明確にして、プロトタイプ、シミュレーションモデル、シナリオ、絵コンテなどを利用し、顧客などの利害関係者と合意を得る。

○システム要件事項の適切性の確認に、顧客をはじめとする利害関係者を参加させて、その結果に承認をもらう。

○要件定義時に定義した要件がテスト可能かどうかを検証するため、システムテスト設計を行う。テストができない要件は利害関係者と調整し、要求を見直す。

5.2 ソリューション例（深く取り組む例）

○一貫性を確認するためには、トレーサビリティマトリクスの機能を提供する商用のツールを利用する。

○未決定事項を明確にするため、利害関係者のニーズおよび制約として、費用、スケジュール、性能、機能性 / 非機能性、再利用可能な構成要素、保守性、またはリスクを話題として取り上げる。

6．関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-1(顧客ニーズ抽出)
・WS-P32-1(システム要件のまとめ)
・WS-TM-1(一貫性確保)
・WS-S4-2(成果物レビューとテスト)
・WS-S5-2(妥当性確認実施)
・SPEAK-IPA：P.3.2 システム要求分析
・CMMI® Ver1.2：要件管理、要件開発、技術解
・共通フレーム 2007：1.6.2 システム要件定義
・ESPR 2.0：SYP1 システム要求定義

7．ふり返り等で参考となる文献等

・プロセス改善ナビゲーションガイド　～ベストプラクティス編～：ソフトウェア・エンジニアリング・センター

(WS-P33-1 システムアーキテクチャ設計)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

システム要件を満たすシステムアーキテクチャ設計を実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○システムの最上位レベルでのシステムアーキテクチャが確立され、ハードウェア、ソフトウェア、および内部・外部インターフェースなどのシステムの要素が識別される。

○システム要求は、いずれかのシステムの要素にすべて配分される。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○システムアーキテクチャ設計は、システムの主要要素を特定し、それらがシステム要求分析で定められた要求事項を満たすように定義する。

○その際に、システムの要求事項の機能要件および非機能要求を考慮する。

○システム要求事項をシステムの主要要素に割り付ける (ソフト / ハードの切り分けを含む)。

○主要なシステム構成要素のインターフェースを定義する。

○ユーザインターフェース、入出力仕様を定義する。

○システム要求、システムアーキテクチャ設計、およびそれらの関係がベースライン化され、すべての影響を受ける関係者に対して連絡される。

3．現実には実現することが難しい点

○システムアーキテクチャ設計は、アーキテクチャの選択と適用のために必要な要件（アーキテクチャ設計時に考慮する機能要件 / 非機能要件）と制約（品質、納期、コスト、安全性、ユーザビリティ）など多面に渡り、すべてを文書化あるいは見える化できない。

○顧客要求を完全に引き出すことは困難であるため、システム要求を満たすシステムアーキテクチャのすべての構成要素を詳細に配分することは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○システム要求事項が下記の点などで実現可能であるか保証できない。

・機能要件と非機能要件
・動作環境などシステム全体の振る舞い
・品質・コスト・納期
・信頼性・安全性、再利用性・拡張性

○ユーザインターフェースが不完全となる。

○システムアーキテクチャ設計書、資料などを整理・体系化し、文書化できないため、構成管理・レビューなどの管理的側面の作業ができない。

5.1 ソリューション例（取り組みやすい例）

○システムアーキテクチャ設計の結果、下記のような資料を作成し、顧客の機能要求が反映されていることを確認する。
・機能ブロック図、データフロー図、機能階層図
・システム間インターフェース仕様書、ユーザインターフェース仕様書、入出力仕様書
・システム構成図（ネットワーク、ハードウェア / ソフトウェア等）

○適切なタイミングでアーキテクチャ設計を凍結し、構成管理表にバージョン名を明記して記載し、ベースライン化する。

○システムの非機能要件に関する、性能見積り表、信頼性見積り表、セキュリティ設計書などに具体的な数値を明記し、関係者に確認する。

5.2 ソリューション例（深く取り組む例）

○設計を評価するため、プロトタイプを作成し評価・改善する。

○顧客要求に対応する、システム要求の割当て表を作成し、すべてのシステム要求がシステムの要素に配分されていることを確認する。

○ QFD(品質機能展開) 法などを用いて、要求事項から設計項目への展開を系統的に実施する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P32-2(システム要件レビュー)
・WS-P33-2(システムアーキテクチャ設計レビュー)
・SPEAK-IPA：P.3.3 システムアーキテクチャ設計
・CMMI® Ver1.2：技術解
・共通フレーム 2007：1.6.3 システム方式設計
・ESPR 2.0：SYP2 システム・アーキテクチャ設計

7. ふり返り等で参考となる文献等

(WS-P33-2 システムアーキテクチャ設計レビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

システムアーキテクチャ設計の適切性を確認する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○システムアーキテクチャ設計が、システム要求分析で定義した項目を満足していることを追跡できる（トレーサビリティの確認）。

○システムアーキテクチャ設計の実施した活動と成果物の実現可能性について、関係者とレビューし、合意できるようになる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○システム要求分析で定義したベースラインとシステムアーキテクチャ設計の間の一貫性を確認する。

○それについて、追跡可能な状態にする。

○システムアーキテクチャ設計の生産物について関係者とレビューを行う。

3．現実には実現することが難しい点

○規模の大きなプロジェクトでは、トレーサビリティマトリクスの作成は手動では困難である。また､市販されているプロジェクト管理ツール（特に要件管理や構成管理ツール）では、現在においても自動化の範囲が限られている。

○要求管理ツールには、要求タイプ間のトレーサビリティ関係や要求に変更が入った際に記号が表示されるものがあるが、疑いのあるケースが表示されるだけで、要求が自動的に更新されるわけではない。したがって、システムアーキテクチャ設計がシステム要求分析で定義した項目を満足していることは、手作業での追跡が必要となる。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○次のような内容が関係者間で十分に合意できず、開発後半で大きな問題が発生する。

・顧客要求の確認とシステム要求の適切性および実現可能性
・システムアーキテクチャ設計を評価する基準
・開発側が利用する設計技法および設計実現に関わるスキル・環境の存在
・システム要求とシステムアーキテクチャ設計のベースラインの内容およびベースライン確定の時期

5.1 ソリューション例（取り組みやすい例）

○関係者とのレビューで、以下の点を明示して、システムアーキテクチャ設計の適切性を合意する。

・システム要求事項との合致性、実現性

・システム要求事項の解釈の曖昧性の排除

○システムアーキテクチャ設計が適切であることを確認するため、システム要求の割当て表やトレーサビリティマトリクスを利用する。

5.2 ソリューション例（深く取り組む例）

○システムアーキテクチャ設計の実現可能性、妥当性、システム要件との適合性などをレビューする。

○アーキテクチャ設計で QFD 等を利用した場合、QFD 等のチャートについても関係者間レビューを行う。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P32-2(システム要件レビュー)

・WS-P33-1(システムアーキテクチャ設計)

・WS-TM-1(一貫性確保)

・WS-S4-2(成果物レビューとテスト)

・WS-S5-2(妥当性確認実施)

・SPEAK-IPA：P.3.3 システムアーキテクチャ設計

・CMMI® Ver1.2：技術解

・共通フレーム 2007：1.6.3 システム方式設計

・ESPR 2.0：SYP2 システム・アーキテクチャ設計

7. ふり返り等で参考となる文献等

・非機能要求仕様定義ガイドライン：日本情報システム・ユーザー協会（JUAS）2008 年

(WS-P34-1 SW 要件定義)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

システム要件からソフトウェア要件にまとめ直す。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェア要求として開発者側の視点で明確に定義し直すことができ、ソフトウェアに割り当てられた要求の曖昧さや矛盾が無くなる。

○ソフトウェア要求の実装の優先順位が明確になり、プロジェクトの見積りが具体的でより厳密になる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ソフトウェアに割り当てられた要求を確認する。

・ソフトウェアに対する要求に不足分や曖昧な点を洗い出す。

○ソフトウェアの特性に合った適切な開発方法論を採用する。

○ソフトウェア要求の複雑度や難易度、機能間の依存性を考慮し、実装の優先順位を付ける。

3．現実には実現することが難しい点

○ソフトウェア要求の分析や定義の重要性を認識していても、十分な期間と工数が投入されない。

○ソフトウェア要求は、ソフトウェアによって実現する機能に加え、ハードウェアからの要求、制約事項、およびユーザインターフェースなども含めるため、エンドユーザにとって理解しにくい手法で表現され、顧客要求の妥当性の検証が難しい。

○ソフトウェア要求の表現手法としては、自然言語による曖昧な表現から統一モデリング言語（UML）も記述言語として有効であるが、まだ普及していない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○曖昧なまま開発を進めることによって、開発終盤に仕様変更や手戻りが起こる。

※ソフトウェアとして割り当てられた機能要件の分析だけに着目し、非機能要件に対し適切な見直しをしないと、開発終盤に仕様変更や手戻りが起こる。

（例）信頼性要求、使用性要求、効率性要求、保守性要求、移植性要求

○ソフトウェア要求をまとめないで実装にとりかかると、上流の成果物との一貫性が失われる可能性が高まる。

5.1 ソリューション例（取り組みやすい例）

○割り当てられたソフトウェア要求が顧客要求やシステムアーキテクチャに合致していることを確認する。

○ UML、DFD、E-R 図、デシジョンテーブルなど適切なデータモデリング技法を決定する。

○非機能要求がソフトウェア要求に反映していることを確認する。

5.2 ソリューション例（深く取り組む例）

○ JIS X 0129 などを参照して非機能要件に関する目標基準を定義する。

・性能
・安全性
・セキュリティ
・人間工学的仕様（人間と設備の相互作用など）
・アクセシビリティ
・運用や保守についての要求

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P33-2(システムアーキテクチャ設計レビュー)
・WS-P34-2(SW 要件レビュー)
・WS-TM-1(一貫性確保)
・SPEAK-IPA：P.3.4 ソフトウェア要求分析
・CMMI® Ver1.2：技術解
・共通フレーム 2007：1.6.4 ソフトウェア要件定義
・ESPR 2.0：SWP1 ソフトウェア要求定義

7. ふり返り等で参考となる文献等

・要求仕様の探検学：D.C. ゴーズ、G.M. ワインバーグ 共立出版 1993 年
・ソフトウェアの仕様化と設計　日科技連ソフトウェア品質管理シリーズ 第 2 巻：花田 収悦他 日科技連 1986 年

(WS-P34-2 SW 要件レビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェア要件の適切性を確認する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェア要求が適切であることを確認できる。

○システムアーキテクチャ設計に対する妥当性、一貫性、完全性およびソフトウェアテストに対する実現可能性を確認できる。

○ソフトウェア要求の変更が、コスト、スケジュール、および技術的影響について評価されて、採否が合議され、ベースラインに記録される。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ソフトウェア要求に、矛盾や抜け漏れがないことを確認する。

・レビューで対象となるソフトウェア要求を特定する。

・エンドユーザ、運用･保守に関わる責任者を含め、レビューに参加する関係者を特定する。

・ソフトウェア要求分析の活動と作業成果物について関係者とレビューを行う。

○ソフトウェア要求が運用環境に与える影響を検討する。

○ソフトウェア要求が、テストで検証可能であることを確認する。

○一貫性の確認を行い、ベースライン化する。

3．現実には実現することが難しい点

○ソフトウェア要求のレビューには画面、操作方法、帳票、環境への影響など、ユーザインターフェースの適切性が含まれるが、エンドユーザがレビューに参加しないことがある。

○トレーサビリティマトリクスや機能の割当て表を完璧に作成するのは困難な場合が多く、追跡可能性の検証は難しい。特にソフトウェア要求の変更において、変更の影響分析、採否判断、ベースライン管理の役割と責任が不明確で個人に依存してしまう。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○システム要求に合致せず、また品質要求を考慮していないソフトウェア要求を関係者に配布してしまい、開発後半において大きな問題を起こしてしまう。

○保有していない技術を使ったソフトウェア要求を作成し、ソフトウェア設計やコード作成ができなくなり、開発スケジュールが守れなくなる。

○システムアーキテクチャ設計とソフトウェア要求との一貫性が確保できず、変更要求の際、影響範囲の特定ができなくなる。

○ソフトウェア要求に矛盾を残したまま次フェーズに進み、結果的に手戻り作業が増えてしまう。

○過剰な要求が含まれているとコスト超過や生産性の低下を招く。

5.1 ソリューション例（取り組みやすい例）

○システムアーキテクチャ設計との一貫性や追跡可能性などの項目を含めたチェックリストを準備し、ソフトウェア要求の適切性をレビューする。
○ソフトウェア要求の適切性レビューには、システムアーキテクチャの設計者、ソフトウェアの設計者、エンドユーザ、その他必要な関係者を加える。
○ソフトウェア要求に優先順位 (複雑度、重要度、依存関係など) を付けてレビューする。
○開発予算やスケジュールに合わないソフトウェア要求が分かった時は、プロジェクトの範囲やリリース時期の変更を交渉する。

5.2 ソリューション例（深く取り組む例）

○要求変更は、変更内容の大小にかかわらず、変更管理の仕組みに従って実施する。
○過去の要求変更の履歴を分析して、非機能要件など変更されやすい要求を特定し、変更の影響をあらかじめ想定する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P33-2(システムアーキテクチャ設計レビュー)
・WS-P34-1(SW 要件定義)
・WS-S4-2(成果物レビューとテスト)
・WS-S5-2(妥当性確認実施)
・SPEAK-IPA：P.3.4 ソフトウェア要求分析
・CMMI® ver1.2：技術解
・共通フレーム 2007：1.6.4 ソフトウェア要件定義
・ESPR 2.0：SWP1 ソフトウェア要求定義

7. ふり返り等で参考となる文献等

・ソフトウェア要求仕様に対する推奨プラクティス：IEEE Std 830-1998

(WS-P35-1 SW 設計)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェア要件を満たすソフトウェア設計を実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェア要素とその振る舞いおよび機能ユニット間のインターフェースが定義され、設計結果をソフトウェア設計のノウハウとして、再利用できる。

○ソフトウェア要求を満たすソフトウェアアーキテクチャが設計され、ベースライン化される。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ソフトウェア要求に記述された設計条件の確認

○ソフトウェア構成の設計

・コンポーネント（機能ブロック）を検討し、整理する。

・コンポーネントを構成する機能ユニット (単体) を検討し、整理する。

○データベースの設計

・データベースの論理設計を行う。

○コンポーネント間のインターフェースの設計

・ソフトウェアを構成するコンポーネント間のインターフェースを設計する。

○ソフトウェア設計結果を文書化する。

3．現実には実現することが難しい点

○ソフトウェア設計の重要性を認識していても、適切な期間と工数が投入されず、十分な検討・設計ができない。

○ソフトウェア設計の文書化が行われていないため、改修や再利用が困難となる。

○機能拡張や再利用を繰り返す中で、ソフトウェア設計への反映が不十分となり、ソフトウェア構造が崩れ、機能拡張や改修が予定どおりできない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソフトウェア設計の目的・達成基準が明確化されずアドホックに行われた結果、必要な分析が不十分で要求の実装漏れが生じる場合がある。

○納期に追われると、実装を優先させるため、ソフトウェア要求とソフトウェア設計の整合性やベースラインが軽視され、一貫性のとれていない成果物を作成してしまう場合がある。

○既存ソフトウェアを再利用する派生開発で、ソースコードとソフトウェア設計が一貫していないため、変更点の把握が困難になる。

5.1 ソリューション例（取り組みやすい例）

○次のような設計手法を用いる

・OOD（Object Oriented Design：オブジェクト指向設計）

・設計モデリング手法・UML（Unified Modeling Language：統一モデリング言語）など

・構造化分析設計

○あらかじめ次のようなソフトウェア設計書の項目をテンプレートとして準備しておき、設計を進める。

（1）概要

（2）システム構成

（3）コンポーネント概要

（4）処理方式（処理シーケンス / ユースケースとコンポーネントの対応）

（5）コンポーネント詳細

（6）システムで扱うデータやデータベースの定義

5.2 ソリューション例（深く取り組む例）

○シーケンス図、状態遷移図、アクティビティ図、ユースケース図、コンポーネント図を活用する。

○設計技法を標準化し、設計書を共通化して、再利用する。

○適切なタイミングでソフトウェア設計を凍結し、構成管理表にバージョン名を明記して記載し、ベースライン化する。

○顧客要求の変更や要求仕様の変更がある場合には、ソフトウェア設計の更新もチェックして、一貫性の確保に努める。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P34-2(SW 要件レビュー)

・WS-P35-2(SW 設計レビュー)

・WS-P35-3(SW 詳細設計)

・SPEAK-IPA：P.3.5 ソフトウェア設計

・CMMI® ver1.2：技術解

・共通フレーム 2007：1.6.5 ソフトウェア方式設計

・ESPR 2.0：SWP2 ソフトウェア・アーキテクチャ設計

7. ふり返り等で参考となる文献等

・ソフトウェア開発・検証技法：松本 正雄、小山田 正史 電子情報通信学会 1997 年

・初めてのアジャイル開発　スクラム、XP、UP、Evo で学ぶ反復型開発の進め方：クレーグ・ラーマン 日経 BP 社 2004 年

(WS-P35-2 SW 設計レビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェア設計の適切性を確認する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェア設計が、ソフトウェア要求事項で定義した項目を満足しているかを早い段階で確認することにより、問題を早期に発見し解決することができる。

○ソフトウェア設計の実現可能性、主要な要件、設計課題、および制約を確立できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ソフトウェア要求とソフトウェア設計とのトレーサビリティが取れているか確認する。

○ソフトウェア設計書の内容が適切であるかどうかを確認する。

○ソフトウェアアーキテクチャ設計書を利害関係者とともにレビューする。

3．現実には実現することが難しい点

○規模の大きなプロジェクトでは、トレーサビリティマトリクスの作成は手動では困難である。

○レビューの重要性を認識していても、期間と工数の制約や利害関係者が集められないなどにより、十分な確認ができない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソフトウェア設計の内容が明確でない、あるいは妥当でない成果物を関係者に配布してしまい、機能、振る舞い、インターフェースなどの面で問題が発生する。

○ソフトウェアの動作環境が不明確な状態でソフトウェアユニットを実装してしまい、テスト、運用、保守がスムーズに実施できなくなる。

○ソフトウェア設計の実施方法の妥当性を関係者で確認できず、ベースライン化できないため、品質、費用、納期が守れない。

5.1 ソリューション例（取り組みやすい例）

○ソフトウェア設計の適切性の確認は、以下の設計資料をレビューする。

・ソフトウェア設計書、ソフトウェアアーキテクチャ設計書、コンポーネント設計書、UML 記述などの高級言語で描かれた状態図、アクティビティ図

○ソフトウェア設計の適切性の確認には、責任の明確化とレビュー時間短縮の観点から、ピアレビューを実施する。ピアレビューは、内容がよく分かった者同志が行うため、問題発見の効率や効果がいっそう上がる。

5.2 ソリューション例（深く取り組む例）

○システム要求、ソフトウェア要求を満たしていることを、利害関係者と共に次の観点でレビューする。

・ソフトウェア要求の実現可能性と要求の変点 / 制限の明確性
・保守 / セキュリティを含めた実運用への対応性、後工程へのスムーズな移行性
・ハードウェアを含めたプロジェクト計画 / 標準 / 規約の順守
・アーキテクチャ設計中に発生した、他の開発グループにまたがる問題
・設計遅れ、手戻りの発生を避けるべく発生した問題の早期解決方法

○ソフトウェア設計が適切かどうかを決めるため、次のような基準を設定してレビューする。操作インターフェース標準 / テストシナリオ / 安全性標準 / 設計制約 / 生産の制約 / 設計の許容誤差 / ソフトウェアユニット標準

6．関連する他の改善検討ワークシート、参考プロセス項目

・WS-P34-2(SW 要件レビュー)
・WS-P35-1(SW 設計)
・WS-P35-3(SW 詳細設計)
・WS-TM-1(一貫性確保)
・WS-S4-2(成果物レビューとテスト)
・SPEAK-IPA：P.3.5 ソフトウェア設計
・CMMI® ver1.2：技術解
・共通フレーム 2007：1.6.5 ソフトウェア方式設計
・ESPR 2.0：SWP2 ソフトウェア・アーキテクチャ設計

7．ふり返り等で参考となる文献等

・ソフトウェア開発・検証技法：松本 正雄、小山田 正史 電子情報通信学会 1997 年
・ソフトウェアの仕様化と設計　日科技連ソフトウェア品質管理シリーズ 第 2 巻：花田 収悦 他 日科技連 1986 年

(WS-P35-3 SW 詳細設計)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェア設計をもとにソフトウェア詳細設計を実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○システムアーキテクチャ設計書とソフトウェア設計書から、処理内容やデータベースを実装できるレベルまでにソフトウェアユニットを詳細化できる。

○ソフトウェア詳細設計で出力された資料を、整理・体系化して、関係者に周知することで、ソフトウェア詳細設計書の整合性を確認できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○組み立て、テストすることができるソフトウェアユニット（モジュール）の記述を含む詳細設計 (detailed design) をする。

○ソフトウェアを構成する個々の機能ユニット、プログラムユニットそれぞれの間のインターフェースを詳細設計する。

○ソフトウェア詳細設計書を作成する。

3．現実には実現することが難しい点

○ソフトウェア詳細設計は、設計者に左右されることが多い。一定の品質を保つには、実装可能性を検証することが必要となる。

○設計者が入出力定義まで関与し、アルゴリズムやプログラミングはプログラマが担う場合、プログラマの差による問題発生を少なくする仕組みが不十分である。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソフトウェアユニットのデータやインターフェースの内容などの下記の詳細情報が不足し、実装段階に進めない、または実装段階で問題が発生する。

(たとえば)

・プログラム構造上の不具合
・データサイズの違いによる性能の不具合
・記号の意味が異なる人的ミス
・データベースの初期値ミス

○ソフトウェア詳細設計書の内容、詳細化が不十分で、整合性や実装可能性を確認できない。

5.1 ソリューション例（取り組みやすい例）

○あらかじめ標準的なソフトウェア詳細設計書の項目をテンプレートとして準備しておき、設計を進める。

○ソフトウェアユニットの処理内容の意図を明確記述する。

○ソフトウェア詳細設計を実施する際、ピアレビューを定期的に実施して内容の整合性を確認し、考えられる問題事項をインシデントとして記録し、上司や利害関係者が参照できるようにする。

○適切なタイミングでソフトウェア設計を凍結し、構成管理表にバージョン名を明記して記載し、ベースライン化する。

5.2 ソリューション例（深く取り組む例）

○ソフトウェアの詳細設計には以下の要素を盛り込む。

・プログラム構造とその各役割、相互関係、共用する資源（テーブル等）

・データと処理方式（実現方法）

・性能設計等

○ソフトウェア詳細設計書をソースコードへ変換する機能を有する UML の統合開発環境を使う。

○顧客要求の変更や要求仕様の変更がある場合には、ソフトウェア詳細設計の更新もチェックして、一貫性の確保に努める。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P34-2(SW 要件レビュー)

・WS-P35-1(SW 設計)

・WS-P35-2(SW 設計レビュー)

・SPEAK-IPA：P.3.5 ソフトウェア設計

・CMMI® ver1.2：技術解

・共通フレーム 2007：1.6.6 ソフトウェア詳細設計

・ESPR 2.0：SWP3 ソフトウェア詳細設計

7. ふり返り等で参考となる文献等

(WS-P36-1 SW ユニット実装)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェアユニットを実装する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェアの設計により定義されたソフトウェアユニットが産出され、ベースライン化できる。

○ソフトウェア要求および設計とソフトウェアユニット間にトレーサビリティが成立する。

○すべてのソフトウェアユニットに対して、ソフトウェア設計を反映しているかどうかを実行可能なコードにより検証することができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ソフトウェアユニットをソフトウェア詳細設計どおりに作成する。

○コーディング規約に準拠する。

○コンパイルし、エラーがあれば修正する。

・コンパイラのバージョンを記録し、構成管理する。

3．現実には実現することが難しい点

○プログラミング言語にはプログラミング時の制約が存在するため、実装しにくいソフトウェアユニットがある。

○実装段階になって設計が実装できないことが判明するリスクがある。

○コーディングをアウトソーシングする場合、ソフトウェア詳細設計書の詳細化が不十分なため納期、コスト、品質を確保するのは難しいことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソフトウェアユニットを合理的に実行する状態にない。

・ソースコードに欠陥がある。

・開発の効率が上がらない。

・テスト可能な状態で実行プログラムが準備できていない。

・再利用するソフトウェア部品が用意できない。

○ソフトウェアユニットを検証することができない。

○他のソフトウェアユニットとの結合と結合テストを行えない。

5.1 ソリューション例（取り組みやすい例）

○新規作成時のコーディング規約を作成しておく。
・コーディング規約は独自に作成することもあるが、標準的な規約として SEC BOOKS「C 言語コーディング作法ガイド」や「MISRA C」などがある。
○新規作成するか再利用するかを決め、再利用する場合は、バージョン、既知の不具合、制約事項を確認しておく。
○ソフトウェアユニットを作成する。
・この際、詳細計書とユニットのトレーサビリティを確保する。
○ソフトウェアユニットテストの手順を作成し、テストデータを作成する。

5.2 ソリューション例（深く取り組む例）

○ソフトウェアユニットのコーディング手法は、以下のものを用いる。
・構造化プログラミング
・自動コード生成
・ソフトウェアコードの再利用
・テスト駆動開発手法（TDD：Test-Driven Development）
○ユニットの行数や複雑度とか静的パス数などの目標値を設定する。
○ペアプログラミングにより実装しながらレビューを並行して行う。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P35-3(SW 詳細設計)
・WS-P36-2(コードレビュー)
・SPEAK-IPA：P.3.6 ソフトウェア構築
・CMMI® Ver1.2：技術解
・共通フレーム 2007：1.6.7 ソフトウェアコード作成及びテスト
・ESPR 2.0：実装および単体テスト

7. ふり返り等で参考となる文献等

(WS-P36-2 コードレビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェアユニットを検証する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○作成したソフトウェアユニットが、ソフトウェア設計に適合していることを確認できる。
○コーディング規約に準拠したコードになっていることを確認できるので保守性が良くなる。
○単体テスト計画書、単体テスト仕様書、コードインスペクション基準など定義した検証基準を利用して、ユニットテストの結果を検証し不適合部分を的確に修正することができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○すべてのソフトウェアユニットのテスト項目および検証基準を定義している。
○静的解析を実施する。
○ソフトウェアユニットテストを実施し、結果を記録している。
○検証の結果、検出された不適合を修正した記録がある。
○コードレビューを実施する。

3．現実には実現することが難しい点

○ユニットテストは、プログラムをコーディングした本人が実施するため、適正なプロセスとして計画・管理されていないことが多い。
○ユニットテストは、チーム作業でないため、計画や記録が疎かになりやすい。
○ユニットテストは進捗状況、品質状況を見える化することが難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ユニットテスト結果を事前に想定し、バグ数などを正確に見積もるためのソフトウェアユニットの下記の検証基準文書が管理できない。
(たとえば)
・単体テスト計画書
・単体テスト仕様書
・コードインスペクション基準
○レビューやインスペクション、品質保証管理者の承認、ベースライン化などを行うためのソフトウェアユニットを検証したという下記の記録が管理できない。
(たとえば)
・単体テスト結果報告書
・コードインスペクション結果報告書
・障害一覧表

5.1 ソリューション例（取り組みやすい例）

以下のようなユニットテストの技法を実施する。

○ホワイトボックステスト

・正常値、異常値を設定してユニットの動作を確認する。

・網羅率（C0: 命令網羅、C1: 分岐網羅、C2: 条件網羅）を変動させ、正確性と生産性のバランスを取る。

○ブラックボックステスト

・同値分割や境界値分析を利用する。

・プログラムへの入力が複数で相互に影響する場合、デシジョンテーブルや原因結果グラフを用いて入力値を変動させ、テストの正確性と生産性のバランスを取る。

○静的解析ツールを利用してコード規約への順守性を確認する。

○検証には、以下の観点を入れる。

・設計とのトレーサビリティ確保 / 設計との外部一貫性 / ユニット間の内部一貫性

5.2 ソリューション例（深く取り組む例）

○テストケースを自動生成し、ホワイトボックステスト、ブラックボックステストを自動実行するツールを利用する。ただし、自動生成ではソースコードの現状どおりにテストケースを作成するため、誤ったテストケースになってしまうことがあるので、ソフトウェア詳細設計内容をもとに生成されたテストケースをレビューで確認する。

○テストカバレッジ情報など詳細なリポートを収集する。

○テストツールは、実装からの情報が必要な場合もあるので、プロジェクト全体で導入を判断する。

○ペアプログラミングにより実装しながらレビューを並行して行う。

○テスト駆動開発手法（TDD：Test-Driven Development）を用い、製造時にテスト項目を設定する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P35-3(SW 詳細設計)

・WS-P36-1(SW ユニット実装)

・WS-TM-1(一貫性確保)

・WS-Test-1(テスト戦略)

・SPEAK-IPA：P.3.6 ソフトウェア構築プロセス

・CMMI® Ver1.2：技術解、検証 (VER)

・共通フレーム 2007：1.6.7 ソフトウェアコード作成及びテスト

・ESPR 2.0：SWP4 実装および単体テスト

7. ふり返り等で参考となる文献等

・ソフトウエア品質管理ガイドブック：森口 繁一 編集 日科技連 1990 年

・単体テストにおける CMM 的アプローチ：保田 勝通他 第 7 分科会（B グループ）日科技連

(WS-P37.38 SW 結合テスト)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

ソフトウェアを結合し、ソフトウェア要件への適合性を確認し、テスト結果を記録する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフトウェア要求どおりにソフトウェアが実現されていることが評価できる。

○ソフトウェアユニット間のインターフェース（入出力など）をテストし共同レビューすることで、ソフトウェア設計どおり実現されたことが確認できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○計画に従って、ソフトウェアユニットを結合する。

○結合テスト項目に従ってテストデータを作成する。必要に応じてスタブ / テストドライバを作成する。

○テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。

○ソフトウェア結合テストを実施し、結果から合否を判定する。

・未解決の問題がないか？ある場合は、問題の内容を確認して早急に対応すべきか、次テストフェーズに持ち越すべきかを判断する。

・未実施となっているテスト項目がないか？ある場合は、理由を確認して対応を検討する。

○確認結果を整理し、問題およびその対応元を明記した上で関係者に通知する。

○不具合検出時、不具合管理票に不具合内容を記録し、その後の対応を判断する。

○不具合を修正した場合は、回帰テストを実施する。

3．現実には実現することが難しい点

○ソフトウェア結合テストでは、プログラムにバグがないことを証明できないため、テストの終了基準を定義しにくい。

○終了基準は、信頼度成長曲線による判定、バグ密度やテストケース密度などの基準値が一般的に用いられるが、理論的裏付けなしに決めることが多く、客観性に欠ける。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソフトウェア結合テスト結果を事前に想定し、バグ数などを正確に見積もるためのソフトウェア結合の下記の検証基準文書が管理できない。

（たとえば）

・テスト計画書、テスト仕様書、テストケース、テストデータ、テスト結果評価基準 / 完了基準

○レビューやインスペクション、品質保証管理者の承認、ベースライン化などを行うためのソフトウェアユニットを検証した記録が管理できない。

5.1 ソリューション例（取り組みやすい例）

○プログラムユニットを結合する時には、バージョン、品質（既知の不具合など）、コンパイル、リンクの条件設定は適切かといった点を確認する。

○作成されたテスト仕様を以下の観点で確認する。

・規模に見合ったテスト項目数（ケース密度）/ 機能の組み合わせ / テスト項目の網羅性 / 外部インターフェース / 割込み処理 / 非機能要件の評価基準の妥当性（パフォーマンスなど）/ エラー処理

○テスト項目に与える入力値については、正常データのほか、以下のようなシステム動作にとっての特異点なども考慮する。

・データ未設定（NULL 値）/ 境界値 / 仕様値を大きく超える値 / 条件 / 最大値・最小値

○テスト結果を確認するときの注意事項

・テスト結果に関しては、テスト仕様の誤りの可能性についても考慮する。

・不具合検出数は、品質基準を満たしているかを確認する。

・回帰テストの場合は、テスト範囲と環境は適切であったかを確認する。

※回帰テストを行うときは、ヒントシート WS-Test-2 を参照する

5.2 ソリューション例（深く取り組む例）

○すべてのユニットを組み合わせてから一気に動作検証する、ビッグバン方式と呼ばれるテスト方法もある（ビッグバン方式は、個々のユニットの確認を十分にした後に行わないと効率が悪くなるので、その採用は慎重な判断が求められる）。

○ソフトウェア結合テストの手法には、トップダウンテスト、ボトムアップテスト、折衷テストがある。説明も入れて構成し直す。

○ソフトウェア結合テストには、トップダウン方式の利点とボトムアップ方式の利点を組み合わせる折衷方式のテストを採用するとよい。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P34-2(SW 要件レビュー)

・WS-P39.310(システム結合テスト)

・WS-Test-1(テスト戦略)

・WS-Test-2(回帰テスト戦略)

・SPEAK-IPA：P.3.7 ソフトウェア結合、P.3.8 ソフトウェアテスト

・CMMI® Ver1.2：成果物統合 (PI)

・共通フレーム 2007：1.6.8 ソフトウェア結合、1.6.9 ソフトウェア適合性確認テスト

・ESPR 2.0：SWP5 ソフトウェア結合テスト、SWP6 ソフトウェア総合テスト

7. ふり返り等で参考となる文献等

・客観的ソフトウェア結合テスト終了基準の提案：高橋 寿一他 日科技連

(WS-P39.310 システム結合テスト)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

システムを結合し、システム設計への適合性を確認し、テスト結果を記録する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○システム要件の不備を発見できる。
○システム設計の意図どおりに稼働することが確認できる。
○システムテストで得られた知見を運用時のチューニング資料として利用することができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

＜システム結合＞
○システム全体を結合するための調整と設定作業を行い、結合を実施する。
○定められた基準を用いてシステムの主要要素が正しく結合できるか検証する。
○システム結合テストの結果を記録する。検出された不適合を修正し、記録する。
○システムアーキテクチャ設計に適合した結合済みシステムを完成する。
＜システムテスト＞
○定められた基準を用いて、結合済みシステムを検証する。
○システムテストの結果を記録する。検出された不適合を修正し、記録する。
○システム要件に適合したシステムであることを確認する。
○システムが使用可能、かつ納品可能であることを確認する。

3．現実には実現することが難しい点

○システム要求事項やシステム設計のドキュメントは、組織や担当者により名称、対象範囲、記載粒度等がまちまちであることも多く、テスト時の確認内容が不適切になることも多い。
○検出される不具合や残存バク数はあらかじめ予測するのは困難なため、リリース後に修正に要する工数予測と適切な体制整備は難しい。
○実務上の問題点を解決する要求事項や設計になっていない場合も多く、使いにくい、もっとこうしてほしい、別機能や修正等の追加要求頻発など大きな問題として表面化することも多い。
○システム結合テストは、業務で利用するすべてのデータ（種別・容量）をテストすることはできない。さらに開発側の最終テストであるにもかかわらず、本番環境で実施できない場合も多い。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○システム要件やシステムアーキテクチャ設計の問題点が先送りされる。
○他システムとつなぐと動かない、許容できないレベルの応答時間になる、特定機種しか動かないなど実運営に耐えられないことがあとで判明する。
○納期遅延、顧客が受け入れられない、実稼働時に問題となる等様々な悪影響が発生する。

5.1 ソリューション例（取り組みやすい例）

＜システム結合＞
○システム全体を結合するための調整と設定対応を行い、結合を実施する。
○結合した機能やシステムが正しく結合できているかを検証する。特に他システム間やハードウェア、ソフトウェアなどの間の相互処理やシステム間インターフェースを確認する。
○欠陥の存在箇所を早期に特定し、検出・修正するために、システム結合は少しずつ行う。
○システム結合には特定の非機能要求（例：性能）のテストを含む場合がある。
＜システムテスト＞
○システムテストでは、以下のようなテストによりシステム全体の機能要求・非機能要求およびデータの品質特性を検証する。
・ユーザビリティテスト　・負荷テスト　・パフォーマンステスト（性能テスト）
・障害テスト　・セキュリティテスト　・構成（両立性）テスト
○テスト環境と本番環境が異なる場合があるため、状況に応じてロングランテスト、高頻度テスト、ボリュームテスト、ストレージテストなどを実施する。

5.2 ソリューション例（深く取り組む例）

○十分なテストケースを確認できない場合、機能要件については典型的な業務プロセスのシナリオを抽出して検証し、非機能要件については過去のプロジェクトのテストケースを再利用する方法もある。
○顧客の要望の優先順位に応じ、テスト項目の網羅性と効果性のバランスに配慮してテストする。
○スキルが伴えば、探索的テストやエラー推測を含めることでより効果的、効率的なテストが可能になる。
○テストのリソースを節約し本番環境に近づけるため、負荷テストツールなどを利用する場合もある。
○あらかじめ定めた（検証･テスト戦略･計画）テスト戦略に基づきテストの定量管理を行う。定量管理を効果的、かつ効率的に行うためにテスト管理ツールを採用する。
（注記）不具合発生については、ヒントシート WS-Test-1 および WS-Test-2 を参照のこと

6．関連する他の改善検討ワークシート、参考プロセス項目

・WS-P33-2(システムアーキテクチャ設計レビュー)
・WS-P37.38（SW 結合テスト）
・WS-Test-1(テスト戦略)
・WS-Test-2(回帰テスト戦略)
・SPEAK-IPA：P3.9 システム結合、P3.10 システムテスト
・CMMI® Ver1.2：検証、妥当性確認 (VAL)
・共通フレーム 2007：1.6.10 システム結合、1.6.11 システム適合性確認テスト
・ESPR 2.0：SYP3 システム結合テスト、SYP4 システムテスト

7．ふり返り等で参考となる文献等

・テスト技術者資格制度 Foundation Level シラバス 日本語版 Version2011.J01

(WS-TM-1 一貫性確保)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

顧客要求からプロジェクトで作成する作業成果物、納入成果物までの一貫性を確保する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○プロジェクトの最初のインプットである顧客からの要求が、最終の納入物も含めたすべてと作業成果物に確実に反映されている。

○変更があったときにも、その変更が確実にすべての作業成果物に反映されている。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクトで作成する作業成果物を明確にして一覧にする。

○それぞれの作業成果物は、何をもとに作成するのかその入力成果物を明確にする。各プロセスの定義にて、何の成果物をもとに何を作成するのかを明記すればよい。

○ソフトウェアの構成を示すベースラインを作成する。（※ベースラインとは、ある時点での入出力関係にある各成果物のどれが含まれているのか、それらのバージョンは何かを示したもの）

○ベースラインを作成したことを関係者に報告する。

○ベースラインを変更する場合は変更の影響分析を実施する

○関係する成果物の一貫性を確認する。（※一貫性とは、入力成果物の内容をもとに成果物を作成しているということ）

○関係する成果物の追跡可能性を確認する。（※追跡可能性とは、作成したある成果物のある項目が、入力成果物のどの項目と関係があるかを示すことができること）

※関係する成果物の一例を示す。

顧客の要求事項とシステム要求事項、システム要求事項とシステムアーキテクチャ設計、システムアーキテクチャ設計とソフトウェア要求事項、ソフトウェア要求事項とソフトウェア設計、ソフトウェア要求事項および設計とソフトウェアユニット、ソフトウェア要求事項および設計とソフトウェアユニット、システムアーキテクチャ設計と結合されたシステム

3．現実には実現することが難しい点

○ 2. で記述した構成管理の概念がまだ理解できていない場合が多く、実施できていない。教育が必要である。各種ツールが流通しており、ツールを使用すると部分的な構成管理の活動ができる場合もあるが、構成管理の概念を確実に理解することが重要である。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ソースコードが変更されても、上流の設計書などが変更されない。

○要求仕様に書いてあるものが実装されていない。

○要求仕様に書かれていないものが実装されている。

5.1 ソリューション例（取り組みやすい例）

○プロジェクトの初期時に構成管理計画を作成し、プロジェクトで作成する成果物を洗い出す。あわせてベースラインを作成する時期を計画する。
○小規模、少人数開発では、フォルダ管理や成果物の命名規約による版管理も可能である（構成管理計画に記載する）。
○ある時点でのソフトウェアの構成を示すために、フォルダにあるバージョンの成果物を格納し、フォルダにベースライン番号を付与する。
○成果物の変更管理の手順を作成している。

5.2 ソリューション例（深く取り組む例）

○バージョン管理ツールを導入し、成果物の版管理を実施する。
○構成管理ツールのタグ機能を使って、ベースライン対象のファイルおよびバージョンの関連づけをしてベースラインとする。
○一貫性の確認を行うために、成果物の相互の関係を管理するツールを導入する。
○顧客要求から、設計書、ソースコード、テストケースなどの関連を記述したトレーサビリティマトリクスを作成し、変更時の影響度合いの分析に活用している。
○仕様などの変更要求があったときに、変更の受け入れの影響を判断したり、承認を実施する役割を置いている。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S2-1(構成管理の仕組み)
・WS-S2-2(構成管理計画)
・WS-S2-3(変更管理)
・SPEAK-IPA：S.2 構成管理
・CMMI® Ver1.2：要件管理、構成管理 (CM)
・共通フレーム 2007：2.2 構成管理
・ESPR 2.0：SUP5 構成管理

7. ふり返り等で参考となる文献等

・プロセス改善ナビゲーションガイド　～ベストプラクティス編～：ソフトウェア・エンジニアリング・センター

(WS-JRev-1 共同レビュー)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで作成する作業成果物について利害関係者とレビューを実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○顧客を含む関係者（利害関係者）とプロジェクト中間成果物を段階的に確認することで、問題を早期発見し、対処することで、後工程での手戻り作業を少なくすることが期待できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○利害関係者とプロセスおよび成果物について共同レビューを行う。

○レビュー結果を記録し、指摘事項をフォローする。

3．現実には実現することが難しい点

○利害関係者に共同レビューに参加してもらうためには、その目的や必要性を十二分に説明し理解してもらうことが重要である。実際には、顧客を含む利害関係者の理解が得られなかったり、リソース手配が難航し、その結果、不十分な実施に終わることも多く、効果が得られないことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○作業成果物およびその提供プロセスを客観的に評価していないため、プロジェクトの目標が達成できるのか確信がもてない。

○様々な観点から確認していないことから、本来は早い段階で見つかる問題の発見が遅れる。

○作業成果物の品質など出来具合について、関係者全員での共通認識ができない。

5.1 ソリューション例（取り組みやすい例）

○プロジェクトの状況などを考慮し、レビュー対象、レビュー計画、レビュー参加者を決め、レビューの準備を行う。

○レビューは問題検出を中心に進め、レビュー記録を作成する。

○レビューでの指摘への対応を確実にするために指摘内容を一覧化して問題解決を確実にする。

5.2 ソリューション例（深く取り組む例）

○プロジェクト計画時に利害関係者とのレビューについて、目的・レビュー対象・実施時期・実施方法・参加者等を明確にし、協力を依頼しておく。

○フェーズの開始・終了のタイミングでプロジェクト関係者を集めた共同レビューを実施し、次フェーズへ進んで問題がないか確認する。明らかになった問題については、課題一覧として解決するまでフォローする。

○プロジェクトであらかじめ決めておいた設計書やテスト結果をチーム外の技術者が重点的にレビューする。

○設計レビューやテスト結果からの定量分析結果を品質保証部門がレビューする。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-1(顧客ニーズ抽出)

・WS-S4-1(検証計画)

・SPEAK-IPA：S.4 検証

・CMMI® Ver1.2：検証

・共通フレーム 2007：2.4 検証、2.6 共同レビュー

・ESPR 2.0：SUP8 共同レビュー

7. ふり返り等で参考となる文献等

(WS-Test-1 テスト戦略)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

テスト戦略と基準を策定する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ソフト結合戦略、要素間のインターフェース戦略、リリース戦略をシステム特性や、顧客要求事項の優先順位に沿って策定できる。

○トップダウンテスト方式やボトムアップ方式の単純なテスト設計ではなく、プロジェクトに合わせた最適なテストが設計できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクト、システム、ソフトウェアの特性に最も適したテスト戦略を選定する。
成果物の例:テスト対象、要求品質、システムの特性、テストの観点（セキュリティ、ユーザビリティなど)、テストレベル、テスト技法（どの技法を誰がいつ使うか)、制約条件、完了基準、など

○テスト対象となる作業成果物が要求事項を満たしていることを検証するための、テスト合格基準を作成する。

○テスト計画書を作成する。
例：テスト戦略目標、スケジュール、役割・責任・連携方法、テスト終了条件、など

3．現実には実現することが難しい点

○個々のプロジェクトは、オペレーション、動作環境、シナリオテストなどに固有の難しさを含んでおり、それに応じたテストの観点を事前に整理していなければ、必要なテスト項目に漏れが生じる場合がある。

○テスト基準が、どのテスト技法をいつ使うかという観点から明確に整理ができていないため、境界値テストや制御パステスト、状態遷移テスト等の技法を選択するタイミングが合わないことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○システムの特性にあった最適なテストが十分に実施できず、プロジェクト全体から見るとテストがボトルネックになってしまう。

○網羅性から見て、テスト項目が漏れるなど、システム全体の欠陥が取り除けない。

○ソフトウエアが備えるべき性質を網羅したテストができない。

○テストケース、テストデータ作成やテスト実施の効率が悪くなる。

5.1 ソリューション例（取り組みやすい例）

○テスト方針（戦略）・計画を文書化する。
○テストカテゴリを大項目、中項目ぐらいまで決める。合格基準も含めて表にまとめる。
○必要なテストデータの系統的な生成、準備の方法を検討する。
○プロジェクトの目的と性質から品質保証のタイプ分けを行って基準を決定し、テスト戦略を選定する（セキュリティテスト、安全性テスト、ユーザビリティテスト等の具体化検討を含む）。

5.2 ソリューション例（深く取り組む例）

○テスト戦略は、ブラックボックステストかホワイトボックステストか（開発者主体か、第三者が主体かを含む）、また不具合検出を主な目的とする「検出型」、想定された範囲内での動作を保証することを主な目的とする「網羅型」、検出型と網羅型の両方の特性を持つ「最善型」などの中から選択する。
○テスト戦略は、一般的にテスト設計を開始するタイミングによって、予防的アプローチと対処的アプローチに分類される。基本設計の段階で予測される項目は予防的アプローチで進め、下流工程になって表出化した項目は、対処的アプローチで進めるよう実施方法をチューニングする。
○テスト戦略は、テストの水準がさらに高まれば、テスト主導アプローチ、分析的アプローチ、モデルベースアプローチ、プロセス準拠アプローチ、経験則的アプローチ、などが適用できる。プロジェクトマネージャは、テストケースにふさわしいアプローチを選定して適用する
○外部のテスト専門業者を活用する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P36-2(コードレビュー)
・WS-P37.38（SW 結合テスト）
・WS-P39.310(システム結合テスト)
・WS-Test-2(回帰テスト戦略)
・WS-S4-1(検証計画)
・SPEAK-IPA:.3.7 ソフトウェア結合、P.3.8 ソフトウェアテスト、P.3.9 システム結合、P.3.10 システムテスト
・CMMI® Ver1.2：検証
・共通フレーム 2007：1.6.8 ソフトウェア結合、1.6.9 ソフトウェア適格性確認テスト、1.6.10 システム結合、1.6.11 システム適格性確認テスト
・ESPR 2.0：SWP5 ソフトウェア結合テスト、SWP6 ソフトウェア総合テスト、SYP3 システム結合テスト、SYP4 システムテスト

7. ふり返り等で参考となる文献等

・演習で学ぶソフトウェアテスト特訓 150 問：正木 威寛 技術評論社
・ソフトウェアテスト入門：ソフトウェア・テスト PRESS 編集部
・第 9 回 組込みシステム技術に関するサマーワークショップ（SWEST9）：西 康晴 2007.8.30-31

(WS-Test-2 回帰テスト戦略)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

回帰テストの戦略を策定する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○デグレード回避のための最適なテストを実施できる。

○属人性を排除し、テストの水準を向上させることができる。

○テスト自動化や、スクリプトなどテスト仕様の再利用プロセスを構築することで、アプリケーション変更の信頼性を高められる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクト、システム、ソフトウェアの特性、発見された不具合の特性を検討して、回帰テストの戦略を立てる。

○戦略に基づき、回帰テスト計画を立てる。

○回帰テスト自動化の検討を行う。

○回帰テストを必要な都度、実施する。

3．現実には実現することが難しい点

○網羅的なテストは、戦略的に行うべきであるが、ビジネス系アプリケーションでは人海戦術で行われることがあり、それにより非効率でかつ品質の確保も不十分になりがちである。

○航空機や自動車の人命に関わる制御系アプリケーションでは、回帰テストごとに全数テストを実施することもあるが、どの程度徹底的に行うべきか判断基準の明確化が難しい。

○客観的にテストするためにテストだけを外部発注している企業もあるが、発注先との責任分担の決定、テスト目標についての発注先とのコミュニケーションの水準確保が難しい場合がある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○不具合改修後に以前は動作していた機能が動かなくなる。

○不具合修正後のテスト（回帰テスト）が、不十分にしか実施されず（場合によっては、ほとんど実施されず）、潜在的な不具合が増加する。

○合理性に欠くテストを行い、回帰テストの回数・工数が過大となる。

○コストパフォーマンスが不明なので、回帰テスト自動化のための投資効果が行われない。

5.1 ソリューション例（取り組みやすい例）

○プロジェクト、システム、ソフトウェアの特性を考慮して回帰テストの方針（戦略）を文書化する。その内容について、場合により、顧客とのレビューで合意する。

○発見された不具合の特性と実施した修正の影響範囲を検討して、個別の回帰テスト実施時点での方針を文書化する。

○網羅的な回帰テストは時間がかかるため、重要度により優先順位および実施範囲を決める。

○テスト作業の省力化のため、修正範囲や確認範囲に応じて回帰テストケースのセットとテストデータのセットを用意し、利用する。個々の回帰テスト実施場面で、それらを単に繰り返し利用するのではなく、どのテストケースが適用できるか見極め、修整して実施する。

5.2 ソリューション例（深く取り組む例）

○早期に再リリースするメリットと不具合再発リスクのトレードオフ分析を行う。

○テスト自動化機能を持つ Eclipse などの統合開発環境ツールを利用すると次の利点があるので、積極利用を検討する。

・ユーザの操作を記録しスクリプトを自動生成できる。

・データプールからフィールドへの入力値や期待値を取得できる。

・テストスクリプトを再実行し、何度でもテストを行える。テスト水準・テスト品質の均質化を図れる。

・変更管理ツールと連携し、変更・機能追加要求とテストスクリプトを一元管理できる。

○回帰テストは、自動テストに向かないという議論もあることを考慮し、パフォーマンステスト、機能テスト、ユーザビリティテストを明確に区分して、自動化するのが適切かどうか検討して行う。

○プロダクトの品質の均等化を図り、テストの精度向上・品質向上・生産性向上を図るため、システムのいろいろな構成部分に対するテストの徹底性の水準を明確に統一する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P32-1(システム要件のまとめ)

・WS-Test-1(テスト戦略)

・SPEAK-IPA：P.3.7 ソフトウェア結合、P.3.8 ソフトウェアテスト、P.3.9 システム結合、P.3.10 システムテスト

・CMMI® Ver1.2：検証

・共通フレーム 2007：1.6.8 ソフトウェア結合、1.6.9 ソフトウェア適格性確認テスト、1.6.10 システム結合、1.6.11 システム適格性確認テスト

・ESPR 2.0：SWP5 ソフトウェア結合テスト、SWP6 ソフトウェア総合テスト、SYP3 システム結合テスト、SYP4 システムテスト

7. ふり返り等で参考となる文献等

・ソフトウェアテスト 293 の鉄則：Cem Kaner 他 日経 BP 社 2003 年

(WS-S1-1 標準文書)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

作成すべき標準文書と記載すべき内容を明確にする。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○ライフサイクルにおいて作成すべき文書が明確になる。
○文書の様式や記述方法がルール化され、分かりやすくなる。
○必要な情報が漏れなく記録され、承認済みかどうかが分かる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○組織で用いる文書について規定する。
○提案から運用までの、ソフトウェア製品およびサービスの全ライフサイクル期間を通して、作成する文書について戦略を立てる。
○プロジェクトでの作成文書を決める。
○文書に関わるルールを決める。

3．現実には実現することが難しい点

○現状は文書化に対する意識が低く、プロジェクトを推進しながら、作成すべき文書を個人レベルで決めており、古い様式や記入要領をそのまま流用し文書を作成している。また、各文書の責任者や承認者が曖昧であり、文書のレビュー記録も残っていない状況である。数人規模のプロジェクトであれば何とかなるものの、プロジェクトの大規模化や、業務の引継ぎが発生した場合に、混乱に陥る可能性が高い。また、プロとしてのマナーに欠けることになる。
○今後は、文書は個人の所有物ではなく、組織の財産であり管理対象であると考え、文書化の基本方針を策定する必要がある。
○一方で、文書は作成されるが、不要なものを山ほど作成し、利用や参照をされないものになっている。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○どのような文書があるのか分からない。
○作成された文書様式が不統一で分かりにくい。
○最新の文書がどれでどこにあるのか分からない。
○正式な文書であるかが確認できない。

5.1 ソリューション例（取り組みやすい例）

○文書のライフサイクルを意識して、番号体系、管理者、承認者、保管方法などを決める。
○文書の開示範囲、配布先について方針を決める。
○プロジェクトの工程ごとに作成する文書名と記載する項目を決める。
（要件定義書、基本設計書、詳細設計書、テスト計画書、テスト仕様書、テスト結果報告書）
○プロジェクトを通じて使用する管理文書・記録と記載項目を決める。
（進捗管理表、レビュー記録、バグ票、課題一覧）
○作成する文書について一覧表を作成して、確実に作成されたか確認できるようにする。

5.2 ソリューション例（深く取り組む例）

○構成管理ツールを活用して、プロジェクトで作成する全文書を管理できるようにしておく。
○文書管理ルールを作成し、プロジェクト内での運営が円滑に行われるように周知徹底する。
・事前にプロジェクトで必要とする文書の種類を洗い出し、様式と記述方法を決めておく。
・テンプレート・作成マクロを準備し、チェックリストで記載内容をチェックする。
・各文書に必須 / 選択の区分を設け、最低限必要な情報の抜け漏れを防ぐ。
・作成した文書のレビュー要領 (参加者、タイミング、チェックシートなど) を決める。
・顧客を含め、完成した文書の承認ルートを決める。
○利用者向け文書については、専門家（テクニカルライター）に依頼する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S1-2(文書化)
・SPEAK-IPA：S.1 文書化
・共通フレーム 2007：2.1 文書化
・ESPR 2.0：SUP4 文書化と文書管理

7. ふり返り等で参考となる文献等

(WS-S1-2 文書化)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

必要なドキュメントを作成し、適切な時期に関係者が利用できるようにする。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○正しい情報に基づいて開発作業ができるので、品質や生産水準が向上する。

○必要な情報が的確に交換され円滑なチームワークが図られる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○定められた文書は、ルールに従いタイムリーに作成する。

○あらかじめ明確化された利用者と配布先にタイムリーに提供する。

3．現実には実現することが難しい点

○プロジェクト途中で要件や仕様が変更する際、影響範囲が多岐に渡ってドキュメントとプログラムにタイムリーに反映することが難しくなることがある。

○プログラム機能の変更や追加時にドキュメントの修正を後回しにして、整合性を崩すことがある。

○正当な利用者以外に機密情報が漏れるなど、文書のセキュリティが保持されないことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ドキュメントがばらばらになり、組織や時間により違う内容を参照してしまう。

○改訂内容や版数が異なるドキュメントを間違って参照してしまう。

○機密性・完全性・可用性の観点から問題が発生する。

5.1 ソリューション例（取り組みやすい例）

○プロジェクトで作成すると決めた文書の一覧を作成し、誰が最新状態を維持し、関係者へ伝達するかをあらかじめ決めておく。ルールどおりに運営されているかはプロジェクトリーダが確認することで徹底させる。

○ドキュメントを電子メールにて関係者へ配布するときは、暗号化する。

5.2 ソリューション例（深く取り組む例）

○作成したドキュメントを関係者がタイムリーに利用できるようにするために、オンラインドキュメンテーションの仕組みを構築する。開発プロジェクトの進行中は、構成管理ツールを利用し、完成後はイントラネットやグループウェアなどの環境を利用する。

○ドキュメント格納フォルダにアクセス制御を設定し、適切な関係者のみに参照可能にする。

○不測の事態に備え、バックアップを取っておく。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S1-1(標準文書)
・SPEAK-IPA：S.1 文書化
・共通フレーム 2007：2.1 文書化
・ESPR 2.0：SUP4 文書化と文書管理

7. ふり返り等で参考となる文献等

・ソフトウェア品質管理ガイドブック、第 7 章 ドキュメンテーション：日本規格協会 1990 年

(WS-S2-1 構成管理の仕組み)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

構成管理の組織方針や仕組みがある。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○構成管理の組織方針や仕組みを確立することで、構成管理の３要素である「バージョン管理」「ベースライン管理」「変更管理」についてのルール作りと運用を徹底できる。

○プロジェクトのライフサイクルで生成される構成要素をベースライン管理することで、デグレードやファイルの取り違えを起こすことなくリリースにつなげることができる。

○リリースバージョンにどのような機能が盛り込まれているかが適切に管理できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○構成管理方針の確立

・構成管理方針、作業の一覧、責任者、構成監査方針などを明確にする。

・仕様書やソースコードなど、構成管理の対象となる成果物を明確にする。

○構成管理手順（仕組み）の確立

・構成管理の手順を明確にする（ツールの使用、利用方法等）

・内部使用および顧客への納入のためのベースライン設定のタイミングや、成果物の変更を追跡し制御手順を明確にする。

○構成管理活動の監査手順を明確にし、監査を実施する。

3．現実には実現することが難しい点

○手順はあるが、守られていない。

・仕様変更などで、ソースコードのみを修正してしまい、設計書やテスト仕様などの一貫性を保つ必要がある関連の成果物を修正しない場合がある（大規模プロジェクトでは機能追加などで仕様変更が多くなり、成果物の構成要素も増加し、変更が管理されておらず、成果物のバージョンや変更の履歴がとれない)。

・実際には関連するドキュメントを最新化して同時に構成要素をベースライン管理する煩雑な作業の手間を惜しみ、プログラムモジュールのバージョンを管理することに精一杯である。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○成果物の構成要素の一貫性と追跡可能性を維持できなくなる。

・仕様変更や保守を管理する際に、ドキュメントの内容を分析し、必要な変更事項を決定し、該当するプログラムを変更する等の手順が踏めなくなり、作業が混乱する。

○ソフトウェア開発の作業にデグレード、ファイルの取り違えなどの事故や不具合が発生し、非効率な作業、不正確な情報、不必要な手戻りが発生する。

5.1 ソリューション例（取り組みやすい例）

○構成管理ツールを導入しない場合、チェックアウト・チェックインなどの機能をフォルダ管理により代替する方法を採用

・たとえば、プロジェクトの共用フォルダにパスワードを設け、構成管理担当者とプロジェクトマネージャ以外は「書込み禁止」にすることで、プログラム担当者が勝手にアクセスできないようにする。作成したプログラムは構成管理者を通し、チェックアウト・チェックインをする。

5.2 ソリューション例（深く取り組む例）

○構成管理方針を作成

・構成管理の適用範囲（対象成果物）、作業の一覧、責任者、構成監査などに関して、その方針をまとめた方針書を作成する。

・構成管理ツールを導入する。

○構成管理手順を作成

・構成管理ツールを使った構成管理対象物の管理手順（ベースラインの作成方法、成果物のバージョン付与方法、トレーサビリティ確保の手順、ベースラインを特定するための命名規約作成等）を作成する。

・構成管理方法は、組織やプロジェクトによって異なるため、組織やプロジェクトでの構成管理の手順を、研修・OJT・オリエンテーション等で周知する。

○構成監査を実施する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-TM-1(一貫性確保)
・WS-S2-2(構成管理計画)
・WS-S2-3(変更管理)
・SPEAK-IPA：S.2 構成管理
・CMMI® Ver1.2：要件管理、構成管理
・共通フレーム 2007：2.2 構成管理
・ESPR 2.0：SUP5 構成管理

7. ふり返り等で参考となる文献等

・ソフトウェア管理の落とし穴：エドワード・ヨードン著、松原 友夫訳 トッパン 1993 年
・ソフトウェア開発の神話：F.P. ブルックス著、山内 正彌訳 企画センター 1977 年

(WS-S2-2 構成管理計画)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで作成する作業成果物、納入成果物と作成時期が決まっている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○構成品目一覧表、成果物一覧表、標準一覧表など、構成管理を行う成果物の管理台帳ができる。
○バージョン管理を一元化することで、複数バージョンが同時に進行する製品リリースの派生バージョンも管理ができる。
○作成したベースラインを基にして、変更管理、ステータス管理、監査など、構成管理情報を関係者に配布できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○構成品目（中間成果物、納入成果物）の特定
・構成管理を行う以下のような構成品目を特定し、その間の依存関係（構成、版）を定義する。
顧客に納入される成果物、指定された内部成果物、調達した成果物、ツール、およびプロジェクトの作業環境における他の固定資産、これら成果物の作成および記述に使用される他の品目
○ベースラインの確立
・定義された構成品目に関してベースラインを作成し、記録する。
・構成管理の管理者の承認を得る。

3．現実には実現することが難しい点

○構成管理は、単純な構成では統制がとれず、かといって緻密すぎては作業を遅延させるジレンマがある。安定性と進捗の釣り合いの観点で、個々のプロジェクトに対して、どの程度緻密な構成がふさわしいかを前もって決めるのは難しい。
○アジャイル開発など開発のペースが十分に速く、手順に縛られたくなく、頻繁にリリースを計画しているようなプロジェクトでは、ベースライン管理の対象範囲や頻度を最適に決定するのは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○製品リリースバージョン（版）におけるファイルの取り違えなどの事故が起き、不具合発生時の修正対象の把握ができない。
○システムトラブルなどでファイルが消失した場合、あるいは仕様変更や機能追加がキャンセルされたような場合、信頼できるもっとも近いポイント（世代）まで立ち戻る手段がなく、大幅な手戻り作業が発生する。
○仕様書とソースコード、あるいはテストケースとテストデータなど、依存関係が複雑な構成管理対象の一貫性を維持できない。
○開発者、および顧客に対して、正確な構成管理情報を提供できないため、作業成果物の保管、取扱いおよび納品を統制できない。

5.1 ソリューション例（取り組みやすい例）

○プロジェクト計画書作成時、工程別に成果物を定義する。

・成果物（構成品目）を漏れなく特定するには、プロジェクトの WBS を利用して、タスクのインプットとアウトプットを識別する。

・成果物は中間成果物と納入成果物に分ける。

・プロジェクト固有の要素（たとえば、ツール、テストドライバなど）は、構成品目との重要な依存関係があるので記録し、保存する。

・ベースライン化の対象となる構成品目を選定したら、構成管理ベースライン表を作成して、構成品目の格納ディレクトリ体系、構成要素ごとの担当グループ、アクセス権限を決定して、工程の完了のタイミングでベースライン化できるように構成管理環境を整備する。

○運用、保守時の考慮

・運用段階あるいは派生開発段階では、変更実施の担当者は開発時と異なると思われるため、開発段階からトレーサビリティマトリクスを使い、下記の依存関係を的確に記録しておくこと

要求の他の作業成果物との依存関係 / プログラムモジュールとそのテスト環境の関係 / 設計書とプログラムコードの関係

5.2 ソリューション例（深く取り組む例）

○構成品目の重要度によって構成管理のレベルを設定する方法がある。

レベル１：開発や運用の基盤となる実行モジュールやソースコード

レベル２：公式レビューで承認された作業成果物

レベル３：定期的なバックアップを実施する作業ファイル

（ベースライン化は公式に実施せずチェックイン / チェックアウトのみを行う）

○物理ベースラインと機能ベースラインに分け、レベルを使い分ける方法がある。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-TM-1(一貫性確保)

・WS-S2-1(構成管理の仕組み)

・WS-S2-3(変更管理)

・SPEAK-IPA：S.2 構成管理

・CMMI® Ver1.2：要件管理、構成管理

・共通フレーム 2007：2.2 構成管理

・ESPR 2.0：SUP5 構成管理

7. ふり返り等で参考となる文献等

・パターンによるソフトウェア構成管理：ステファン・P・バーチャック、ブレッド・アップルトン 翔泳社 2006 年

・コンサルタントの道具箱　勇気と自身がもてる 16 の秘密：ジェラルド・M・ワインバーグ 日経 BP 社、2003 年

(WS-S2-3 変更管理)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで作成する作業成果物、納入成果物の変更管理がされている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○変更要求が QCD の面から正しいものであることが確認できる。
○変更要求に対して、実際に成果物に手を加える前に、成果物への影響範囲を事前に正しく把握し、変更を計画することができる。
○変更作業とその結果を記録し確認することにより、成果物間の整合性を保証できる。
○障害の内容と対策を記録し、成果物の状況および変更履歴を報告することにより、関係者に構成品目の最新の状態を報告することができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○構成品目の変更要求およびそのリリースに対し、レビューを実施し、承認する。
○変更要求およびリリースした構成品目を、関係者が利用できるよう通知する。
○構成品目の最新の状態を記録し、かつ構成管理者に報告する。
○構成品目の保管、取り扱いおよびリリースを管理する。

3．現実には実現することが難しい点

○変更要求は、要求の変更、設計の変更、実装の変更などによって影響範囲が大きく異なる。たとえば、プロジェクト終盤での上流工程での変更要求は、影響範囲は甚大で大幅な手戻りが発生するため、その影響範囲を正確に見積もるのは納期などの点から現実には難しい。
○変更が発生した場合には、リスクヘッジの意味を含めて、すべてのテストを可能な限り再実行することが望ましいが、要求間の依存関係を根拠にしたテストで品質を保証することは難しい。
○変更管理手順が煩雑なため、変更依頼の承認がないまま、作業者が勝手に構成変更をする場合がある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○変更要求を鵜呑みにして、変更による影響（対応によって得られるメリットと、対応に必要なコスト）を事前に評価しなかったために、納入成果物のＱＣＤに影響を及ぼしてしまう。
○構成品目間の一貫性、完全性が崩れてしまう。
○変更が繰り返される場合、最新の変更状況や変更理由が把握できず、対応者が変更箇所の影響範囲を狭く見積もり、変更要求への対応が他の成果物へ悪影響を及ぼす。

5.1 ソリューション例（取り組みやすい例）

○変更要求に対する受付、承認、記録などの手順の確立

・変更要求に対する、変更の受付と見積り、変更の扱いと優先度の決定、変更の実施、変更の完了処理、およびトレーサビリティ確保の仕組みなどの手順を作成する。その際、変更の承認者、各作業の責任者を明確にしておく。

5.2 ソリューション例（深く取り組む例）

○リリースした構成品目の変更は、変更の関係者だけでなく、利害関係者すべてに影響するため、開発の規模が大きい場合などは、変更の承認や構成の管理を実施するような組織（たとえば、構成管理委員会）が、変更の影響を分析・評価し、変更の採否を的確に意思決定する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-TM-1(一貫性確保)
・WS-S2-1(構成管理の仕組み)
・WS-S2-2(構成管理計画)
・SPEAK-IPA：S2 構成管理
・CMMI® Ver1.2：要件管理、構成管理
・共通フレーム 2007：2.2 構成管理
・ESPR 2.0：SUP5 構成管理

7. ふり返り等で参考となる文献等

(WS-S3-1 品質保証計画)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

品質保証の仕組みがあり、計画されている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○利害関係者に対して、プロセスおよび関連する作業成果物の品質を保証する客観的な方針や仕組みができている。

○プロジェクトの上位管理者および品質保証管理者の承認を得ることで、品質達成に関して、プロジェクト従業者が独断的に判断しない仕組みを担保できる。

○品質保証活動の PDCA サイクルで知り得た様々なノウハウを自然に蓄積して、品質達成および品質保証の仕組みをより良くするためにフィードバックできる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○ 品質保証活動の戦略を立てる。
たとえば、品質保証方針（顧客の品質要求に応える等）

○ 品質保証の実行計画を立てる（体制・役割分担、スケジュール、対象物、実施方法など）。

○ 品質保証の基準を設定する（たとえば、納品基準、プロセスの順守率）。

3．現実には実現することが難しい点

○品質保証計画の考えが浸透せず、プロジェクトが品質保証に対応する活動計画を立てない。

○品質測定ツールが、的を射た、また適切な量的・精度的水準のデータを提供するとは限らない。

○プロジェクト計画策定時に、文書や記録の作成の工数を見積りに含めないことがよくあり、勤務工数としてカウントされないことまで散見されるので、品質保証活動に必要な文書や記録を作成する時間が不十分になる。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○ビジネス目標に適合した、また中長期的な視野から見た、品質保証の意義が的確に認識されない。

○全社的な品質保証活動に取り組まないため、開発側と品質保証側で、品質保証活動に対する認識が異なるなど、品質保証活動の効率が悪い、あるいは効果が現れない。

○重大クレームの早期発見と万が一の発生時への対応が場当たり的となり、必要な情報が顧客あるいは経営トップ層に早く・正しく・漏れなく伝えられない。

○開発の日常の問題、開発プロセスに対する運用状況の不適合など習慣化した不適正事態を、監査などで見つけ、全社的に是正する仕組みができあがらない。

5.1 ソリューション例（取り組みやすい例）

○品質保証で常に同等の基準を適用できるように、あらかじめプロジェクトが実施する活動のチェックリストを作成し、また品質保証担当者の権限を明示しておく。
○プロジェクトの計画に品質保証の実施のタイミングを明記しておく。プロジェクト実施体制に品質保証担当をアサインする。
○品質保証活動を有効なものとして維持するため、ヒヤリングやインタビューなどの活動からチェックリストを定期的または随時に改訂することで、開発者に身近に役に立つものとする。
○開発者のキャリアパスに品質保証活動を追加し品質保証活動の意義や難しさを理解させる。
○品質保証活動の客観性を保つために、開発組織とは独立した組織により実施する。
○品質保証のための検査ツール等を系統的に導入する。
○品質測定データを系統的に蓄積し、品質チェックに活用する。

5.2 ソリューション例（深く取り組む例）

○プロジェクトの規模や難易度が異なる場合、以下のような基準により、品質保証活動の深さや範囲を決めることにより、品質保証活動の効率を向上させる。
・開発規模や金額から判断したリスク値の程度 / プロジェクトの月次報告書の提出遅延数の程度 / プロジェクトのスケジュール効率指数（SPI）、コスト効率指数（CPI）の値 / 品質メトリクスの評価基準・合格判定基準
○品質保証活動は、プロジェクト管理だけで運用・改善できないため、独立の品質管理部門が成果物の第三者レビューやプロセスの順守確認（＝監査）を適宜計画する。
○品質保証管理者は、プロジェクトや品質に関するトラブルが発生した場合、関係者や社会への影響度を判断して、あらかじめ定義した経路・手順で、経営トップ、関係機関、関係者へ報告するとともに、品質管理委員会等の対策実施機能を立ち上げ、問題の根本的原因を追究し、是正処置を行う。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S4-1(検証計画)
・WS-S5-1(妥当性確認計画)
・WS-S3-2(品質保証活動実施)
・SPEAK-IPA：S.3 品質保証
・CMMI® Ver1.2：プロセスと成果物の品質保証 (PPQA)
・共通フレーム 2007：2.3 品質保証
・ESPR 2.0：SUP2 品質保証

7. ふり返り等で参考となる文献等

・ソフトウェア品質保証の考え方と実際：保田 勝通 日科技連出版社 1995 年
・ソフトウェア品質保証の考え方と実践：服部 克己 Unisys 技報 28(4)、395-412、2009-02 日本ユニシス
・解説：アーンド・バリュー・マネジメント：冨永 章 プロジェクトマネジメント学会 2003 年

(WS-S3-2 品質保証活動実施)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

品質保証活動が実施されている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○プロセスや開発中の作業成果物の適切な可視性を提供することで、プロセスおよび作業成果物の品質を向上できる。

○プロセスや開発中の作業成果物が標準や手順に則っていることを、レビューや監査を通じて検証することで、属人性を排して、ソフトウェア品質を向上させる仕組みが構築できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○製品が、規格、顧客要件に合致しているかどうかを客観的に検証し保証する。

○プロジェクトのプロセスと活動が、規格、手順、顧客要件を守っているかどうかを客観的に検証し保証する。

○プロジェクトの作業成果物やプロセスに関し、顧客要件に対する問題点や社内基準不適合を特定し、記録する。

○品質保証の実行計画を順守する。

○品質保証活動の証拠を残し、それを維持する。

○品質保証活動が組織方針に基づいて実施されていることを監査する。

3．現実には実現することが難しい点

○開発側はプロフィットセンター、品質保証担当はコストセンターなので、両者の意見が対立した場合に、品質保証が権限を行使して工程を止めたり、やり直しさせるのは実際には難しい。上位管理者にエスカレーションする場合でも、同様に難しい。

○品質保証担当は、何年も開発現場から離れているようなケースが多く、上流工程から品質保証レビューに参加しても、シンタックス・チェック以上の突っ込んだアドバイスはできないことが多い。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○プロジェクトが第三者視点で検証されず、開発者自身では発見しにくかった欠陥を含んだまま出荷して、市場ではじめて客観的な視点で評価されることになる。

○市場で問題が発生したときに、確かに開発工程で品質を確保するための活動を実施した、という証拠を提示できなくなり、法的・社会的な批判にさらされる。

5.1 ソリューション例（取り組みやすい例）

○品質保証活動の中核にレビュー実施とその記録を位置づけ、プロジェクト立ち上げ時、各工程完了時、出荷判定時など重大なイベントでは品質保証レビューを実施し、品質保証担当による承認を経ないと次のステップへ進めない。

○品質保証レビューでは、作業成果物のレビューばかりでなく、社内プロセスや標準が順守されていることを評価する。そのためには各プロセス実施上のチェックリスト、コーディング規約、GUI ガイドラインなど、プロジェクトが順守するべき組織の規程を適用する。

○レビューの形態は、標準化している。具体的には、プロジェクトからレビュー依頼書を提出しレビューを実施し、レビュー終了後は品質保証担当がレビュー実施報告書を作成してレビュー内容と是正の証拠を残す。

5.2 ソリューション例（深く取り組む例）

○品質保証活動の状況を把握するために、レビュー実施回数 (予定 / 実績)、レビュー順守率、レビューの実施工数、各レビューアーの投入工数、レビュー会議（会議形式の場合）実施工数、各 SQA レビューの判定結果、組織単位でのレビュー判定結果（同意 / 不同意）数の割合等を計測して、組織とプロジェクトのプロセスを統計的に分析する。

○顧客の本番業務でシステムが満足に動作することを保証するために、システム運用業務（ITIL など参照）や保守業務（共通フレーム 2007 を参照）についての品質点検基準を定め、その作業内容と開発業務・運用・保守業務間の適切な品質保証コミュニケーションを行う。

○レビュー活動記録は、社内外の監査員が定期的に監査して、監査結果を文書にまとめ上級管理層に報告する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S4-2(成果物レビューとテスト)
・WS-S5-2(妥当性確認実施)
・WS-S3-1(品質保証計画)
・SPEAK-IPA：S.3 品質保証
・CMMI® Ver1.2：プロセスと成果物の品質保証 (PPQA)
・共通フレーム 2007：2.3 品質保証
・ESPR 2.0：SUP2 品質保証

7. ふり返り等で参考となる文献等

・ソフトウェア品質保証の考え方と実践：服部 克己 Unisys 技報 28(4)、395-412、2009-02 日本ユニシス
・品質保証レビューにおける品質評価手法：中辻 等 UNISYS TECHNOLOGY REVIEW 第 99 号、FEB. 2009 日本ユニシス

(WS-S4-1 検証計画)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで作成する作業成果物の検証に関する組織方針や基準を策定している。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○組織方針に基づき検証活動を推進することで、各プロジェクトで統一的な検証計画書ができる。

○レビュー方針、品質保証方針などの検証基準を明確化することで、検証活動に投入する労力と品質レベルのバランスを取ることができる。

○ステップあたりの想定する欠陥数、実施するテストケースの件数などの検証基準が明確になる。

○作業成果物のタイプやライフサイクル上の役割に合った検証方法が採用できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○検証活動を推進するための組織方針を立てる。

検証方法（検証対象、実施タイミング、体制、作業内容、作業成果物など）を、検証推進責任者が作成し、周知する。

○要求されているすべての作業成果物に対し、検証のための基準を設定する。

○検証戦略に基づいて、検証の実行計画を立てる。

投入できる工数、検証項目の量、テストケース、運用シナリオなどを調整し、組織に応じた、プロジェクトに適したものにする。

3．現実には実現することが難しい点

○データが集まっていない、様々な業務特性が存在するなど、適切な基準を設定することが難しい。

○プロジェクトの開発規模やソフトウェアの重要性（たとえば、クリティカルソフトウェアと呼ばれる人命に関わるようなソフトウェアか否か）によって適切な検証方法や基準を決定すべきだが、確立された決定方法がない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○必要なタイミングで必要な検証活動が行われず、作業成果物が正しいことの保証や、早期に欠陥を発見し改修することができない。

○検証戦略や基準がないため、「重箱の隅を突付く」ような検証や、逆に必要な検証作業を省いてしまうなど、品質・生産性の低下を招くことになる。

5.1 ソリューション例（取り組みやすい例）

組織方針を以下のように記載する。

○検証の方法や基準は、下記のようにライフサイクルのＶ字型モデルに対応させる。

・要件定義書の要件はシステムテストで、ソフトウェア設計は結合テストで検証する。

・コードレビューを実施する。

・前工程の要件が当該工程で反映されていることをレビューする。

○検証活動（レビュー、テストなど）は、適切な要員を割当て、実施し、記録を残す。

・適切な要員は、有識者、第三者、検証内容によっては利害関係者を考慮する。

○基準

・設計書についてデータ白書を参考にページあたりのレビュー基準（時間）を設け、結合テスト、システムテストについて基準となるテスト密度を設けて、レビューとテストを実施する。

・実績データが蓄積された後、それぞれの基準値を策定し、各プロジェクトにて利用する。

5.2 ソリューション例（深く取り組む例）

○レビュー技術も様々な方法が提案されているので、世間で使われている技術を調査して、プロジェクトにあったものを選べるようにする（アジャイルインスペクション、ＱＩ、リスクベースドレビュー等）。

○検証活動は担当者のスキルに依存するため、レビュー手法・テスト技法の習得には訓練が必要である。活動に先立ち、使用する手法・技法のトレーニングプログラムを立ち上げる。

○設計レビューおよびテスト結果を分析し、不具合の早期抽出するための品質向上策を検討する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S4-2(成果物レビューとテスト)

・WS-JRev-1(共同レビュー)

・WS-S5-1(妥当性確認計画)

・WS-S3-1(品質保証計画)

・WS-O16-1(測定活動)

・SPEAK-IPA：S.4 検証

・CMMI® Ver1.2：検証

・共通フレーム 2007：2.4 検証

7. ふり返り等で参考となる文献等

・ソフトウェアインスペクション：Tom Gilb、Dorothy Graham、伊土 誠一、富野 寿訳 構造計画研究所 1999 年

・ピアレビュー：Karl E.Wiegers、大久保 雅一訳 日経 BP ソフトプレス 2004 年

・レビュープロセスの継続的改善方式：林 章浩、片岡 信弘、木野 泰伸、津田 和彦 品質管理学会論文誌、Vol.40、No.3、pp.71-80、2010

(WS-S4-2 成果物レビューとテスト)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで作成する作業成果物のレビューやテストを実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○作業成果物やサービスは、その仕様（あるいはニーズ、要求事項）や規定要求事項を適切に反映していることを確認できる。

○ライフサイクルに適した検証の方法を採用することにより、レビューやテストなどで欠陥の発見と是正を早期に実施し、不具合を的確に修正できるため、リリース製品の品質を高められる。

○検証活動の結果、摘出した問題点や不適合箇所を特定し記録し分析することで、関係者が問題の根本原因を把握し、問題解決に利用することができる。

○検証担当者の自己流の検証ではなく、組織として一貫した検証を実施できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○検証計画に基づいて、検証を行う。

○検証時に検出した不具合を特定し、記録する。

○検証活動の結果、検証基準に達していることを確認する。

○検証の結果を、関係者で共有し、活用する。

3．現実には実現することが難しい点

○検証活動は実際のプロジェクトでは状況により計画に沿って実行できない場合がある。

○検証結果は、検証担当者のスキル・経験に依存するため、バラツキが生じる。

○記録の仕方によっては、検証結果も活用に至らないことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○作業成果物（設計書、ソースコードなど）の不具合、作業成果物間の不整合などの発見が遅れ、手戻り作業の発生、進捗遅延、稼動後の障害発生など、リリース製品のＱＣＤが確保できない。

○レビューやテストなどの検証結果が報告されず、重要な課題を関係者で共有できなくなる。

○開発された作業成果物について、求められている品質要件を満たしているか確信がもてない。

5.1 ソリューション例（取り組みやすい例）

○レビュー

・最新の要件や設計書との一貫性を確認する。

・チェックリストに基づき、レビューを実施する。

・レビュー記録に対象成果物、参照文書、レビュー参加者、レビュー日時および実施時間と指摘事項を記録する。

・コードインスペクションを計画に沿って実施する。

○テスト

・テスト仕様書に基づきテストを実施し、その結果を記録する。検出した不具合は、原因を特定し、不具合管理票に記録する。

・不具合修正後に、回帰テストを実施する。

○レビュー密度、指摘率、テスト密度、バグ発生率を元に品質評価を行い、各種の基準値（参考値）に達しているかを確認する。

5.2 ソリューション例（深く取り組む例）

○早期の不具合予測を行うためにアジャイルインスペクションを採用する。

○重要な成果物に対しては、有識者がインスペクションまたはモデルベース検証を行う。

○品質リスク度合いによって、レビュー方法を選択する。

○ペアプログラミングにより、検証も兼ねソースコードを作成した。

○テストファーストやテスト駆動開発としてテストコードを書いてからプログラミングを行う。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-Test-1(テスト戦略)

・WS-S4-1(検証計画)

・WS-S3-2(品質保証活動実施)

・SPEAK-IPA：S.4 検証

・CMMI® Ver1.2：検証

・共通フレーム 2007：2.4 検証、2.6 共同レビュー

・ESPR 2.0：SUP8 共同レビュー

7. ふり返り等で参考となる文献等

・ソフトウェア・テストの技法 (第 2 版)：G.J. Myers, et al. 長尾 真他訳 近代科学社 2006 年

・ソフトウェアのテスト技法：玉井 哲雄、松田 茂広、三嶋 良武 共立出版 1988 年

・不具合の分析とフィードバックによるテストの設計：西 康晴 JaSST03 予稿集 2003 年

・ISO9126-1 および JIS X 0129-1

(WS-S5-1 妥当性確認計画)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

妥当性確認の組織方針や基準を策定する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○開発したシステムが特定の意図された用途に合致していることを実証する計画を明確にできる。

○要件定義や設計など上流工程の妥当性確認は、発注するプロジェクトに対して、投資をしてもよい、コストをかけてもよい、と判断する基準にできる。

○受入れ検査に伴う妥当性確認は、発注者が求める IT への投資効果が得られるか否かを確認し、効果が実現するか否かを判断できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○妥当性確認を推進するための組織方針を立てる。

確認方法（対象、実施タイミング、体制、作業内容、作業成果物など）を、妥当性確認の推進責任者が作成し、周知する。

○対象となる作業成果物に対し、妥当性確認のための基準を設定する。

○妥当性確認に対する組織方針に基づいて、妥当性確認の実行計画を立案する。

投入できる工数、確認項目の量、テストケース、利用シナリオなどを調整し、組織に応じた、プロジェクトに適したものにする。

3．現実には実現することが難しい点

○実績データが蓄積されていない、様々な利用状況や用途が存在するなど、適切な基準を設定することが難しい。

○プロジェクトの開発規模やソフトウェアの重要性（たとえば、クリティカルソフトウェアと呼ばれる人命に関わるようなソフトウェアか否か）によって適切な妥当性確認方法や基準を決定すべきだが、確立された決定方法がない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○必要なタイミングで必要な妥当性確認が行われず、成果物が顧客の利用状況に適切であることの保証や、早期に問題を発見し改修することができなくなる。

○妥当性確認の戦略や基準が策定されない場合、顧客の機能、非機能要求、あるいは開発側の提案活動によって採用された要求事項に対して、思いつき、または場当たり的な確認に終始してしまう。あるいは必要な確認作業を省いたり、プロジェクトの最後に確認が集中し、用途に合わない・使いにくいシステムになったり、納品遅延や仕様変更の多発、顧客不満足などを招くことになる。

5.1 ソリューション例（取り組みやすい例）

組織方針を以下のように記載する。

○妥当性確認の方法や基準には、以下の内容を含める。

・システム化によりあらかじめ特定していた課題・問題事項が解決されることを、実稼働環境やそれに近い疑似環境で確認する。

・上流工程の早期に利用方法、画面イメージなどのユーザインターフェースを明確化し、確認するためのプロトタイピング、顧客との共同レビュー、操作性を確認するユーザビリティテスト、正式リリース前のβ版公開による評価など、プロジェクト特性により適切な手段を採用する。

○妥当性確認（レビュー、テストなど）は、適切な要員を割当て、実施し、記録を残す。

・適切な要員は、顧客、エンドユーザ、有識者、第三者などの利害関係者を考慮する。

○参考値として妥当性確認対象について、データ白書を参考にページあたりのレビュー基準（時間）や、システムテストについて基準となるテスト密度を設けて、レビューとテストを実施する。

・実績データが蓄積された後、それぞれの基準値を策定し、各プロジェクトにて利用する。

5.2 ソリューション例（深く取り組む例）

さらに妥当性確認の手段の選択肢として以下のようなものがある。

○顧客自らテストを実施するのが困難な場合、開発側が顧客や上級管理者の前で要求仕様書をまとめた提案書を提示し、説明することによって代替する。その場合には、検証で用いたテスト項目を妥当性の確認に流用するだけではなく、顧客参加の公式レビュー議事録などを参照して、顧客の判断基準を事前によく理解する。

○プロジェクトの早期に顧客のニーズ、期待、および制約を引き出し、分析した結果、妥当性を確認し、顧客を含めた関係者の意思疎通を行うため、ペルソナ・シナリオ法や段階リリース、アジャイル開発などの手法がある。

○システム全体の中から優先度順に部分機能を構築・リリースしながら段階的に統合して仕上げる段階リリース方式を採用し、利用状況や用途を明確化しながら進める。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-Test-1(テスト戦略)
・WS-S4-1(検証計画)
・WS-S5-2(妥当性確認実施)
・WS-S3-1(品質保証計画)
・SPEAK-IPA：S.5 妥当性確認
・CMMI® Ver1.2：妥当性確認
・共通フレーム 2007：2.5 妥当性確認

7. ふり返り等で参考となる文献等

(WS-S5-2 妥当性確認実施)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

妥当性確認（顧客が本来やりたいことが実装できているかを確認すること）を実施する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○必要な妥当性確認のアクティビティが実施される。
○妥当性確認テストで出た問題が特定され、かつ記録される。
○開発されたソフトウェア作業成果物が、その意図された利用について適当であることの証拠が提供される。
○妥当性確認の結果が、顧客およびその他の関連関係者に対して利用可能になる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○妥当性確認の実行計画に基づき、成果物または成果物構成要素の妥当性を確認する。
○妥当性確認の結果を分析する。
○妥当性確認の結果、検出した問題を特定し、記録する。検出した問題を解決する。
○妥当性確認の結果、対象成果物が意図された利用について適切である証拠を確認する。
○妥当性確認の結果を、顧客およびその他の関連関係者に対して利用可能にする。

3．現実には実現することが難しい点

○利用者と利用環境（状況）対して適切な要求仕様を明確化することが必要であるが、現実の要求仕様書には、開発者視点の要求しか書かれていないことが多い。
○要求仕様書を根拠にして妥当性確認のテスト項目を作成しても、必要なテスト項目をすべて網羅することは難しい。
○妥当性確認の環境は、リリース後の製品の運用環境とは異なるため、環境との相性、運用条件や運用シナリオなどが原因で生じる問題を予見することは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○顧客の機能要求、非機能要求、あるいは開発側の提案活動によって採用にされた要求事項が、リリース後の実際の環境で意図したとおり運用できない。
○プロダクトを実際に利用する際、予期しない問題が生じ、実際の利用に支障を来す、あるいは「使えない・役に立たないソフトウェア」「宝の持ち腐れ」になる。
○発注側の視点から見ると、想定していた機能や性能と異なるソフトウェア開発に対価を払っていた、などの問題が起こる。

5.1 ソリューション例（取り組みやすい例）

○開発プロジェクトの終結段階で成果物や構成要素の妥当性確認が NG 判定になると手遅れになるため、要件仕様の妥当性確認、設計の妥当性確認など、早い段階から妥当性を確認することが重要となる。そのために、マイルストーンとなる共同レビューでは必ず顧客を加え、画面イメージなどユーザインターフェースを定期的に確認する。

○上流工程での妥当性確認には、要求レビュー、プロトタイピング、モデルの妥当性確認、などの手段がある。これらの実施可能性を早期に判断した上で、模擬環境、実機環境、本番運用環境などでプロトタイピング等を行う。

○操作性を確認するユーザビリティテスト、正式リリース前のβ版公開による評価を行う。

5.2 ソリューション例（深く取り組む例）

○顧客自らテストを実施するのが困難な場合、開発側が顧客や上級管理者の前で要求仕様書をまとめた提案書を提示し、説明することによって代替する。その場合には、検証で用いたテスト項目を妥当性の確認に流用するだけではなく、顧客参加の公式レビュー議事録などを参照して、顧客の判断基準を事前によく理解する。

○プロジェクトの早期に顧客のニーズ、期待、および制約を引き出し、分析した結果、妥当性を確認し、顧客を含めた関係者の意思疎通を行うため、ペルソナ・シナリオ法や段階リリース、アジャイル開発などの手法がある。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S5-1(妥当性確認計画)
・WS-S3-2(品質保証活動実施)
・SPEAK-IPA：S.5 妥当性確認
・CMMI® Ver1.2：妥当性確認
・共通フレーム 2007：2.5 妥当性確認

7. ふり返り等で参考となる文献等

(WS-S8-1 問題解決管理)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで起きた問題について系統的に対処する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○開発の過程で生じる様々な不具合や課題を集め、それらへの対策や解決状況を関係者間で明確化・共有化できる。

○商品、技術、設計・開発プロセスの共通性を見える化することで、それまで個人や管理者に埋没していた、問題の根本原因を明確にできる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクトで問題として取り扱う対象を特定する。

○担当者は、問題の原因を分析し、分析内容を記録する。

○記録された問題を確認し、解決する担当者を割り当てる。

○分析結果を検討し、以下のように対処方法を判断する。

・変更する場合は、変更依頼を発行し、完了まで管理する。

・変更しない場合は、そのまま問題を完了する。

・プロジェクト内で解決できない場合は上位の管理層に課題を報告し、可決を依頼する。

○問題の傾向を分析し、根本原因をプロジェクトや組織内で共有する。

3．現実には実現することが難しい点

○プロジェクトで発生した課題は、大きくなってから報告される。

○発見された欠陥は修正の可否の検討なしに個人に依存して対応される。

○発見された欠陥と対応策、対応の範囲を超えて、周辺の変更を合わせて実施してしまう。

○業務範囲やセクショナリズムの壁が大きく、広く公平な視野を持てない場合、プロジェクト単位で解決できないような大きな問題は、論理的に正しいアプローチでも理解されないことがある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○問題に対して事象を裏返したような対症療法的な解決策を選択するため、問題の明確な定義と根本原因を除去できず同じような問題が再発する。

（たとえば）

「リソースが不足していた」という問題に対して「リソースを追加する」のような対策を行い、真因が見積りの精度にあると分析しなかったため、別のプロジェクトでも同じことが起きる。

○経験と勘に頼った問題解決を行い、後でふり返ってなぜそのような判断をしたのか、なぜ判断を間違えたのかの理由を辿れなくなる。

5.1 ソリューション例（取り組みやすい例）

○プロジェクトで発生する問題を『課題』『レビュー指摘』『テスト欠陥』に分けて管理する。
○課題管理表やレビュー記録、欠陥管理表とテンプレートを作成する。
・発見された課題、レビュー指摘、テスト欠陥は課題管理表、レビュー記録、欠陥管理表に記録し、追跡管理する。
・課題はプロジェクトの進捗会議などで課題のクローズ状況や新たな課題の発生を確認する。
・発見されたレビュー指摘やテスト欠陥は直ちに修正するのではなく、原因を分析した後、修正するかどうかは関係者と合議の上で決定する。
○記載原因、埋め込まれたプロセス、検出されたプロセスなどの傾向分析を行い、根本原因を追究する。

5.2 ソリューション例（深く取り組む例）

○ Web ツールの導入による関係者と内容の共有を図り、対応を確実にする。
○ QC 7つ道具や新 QC7 つ道具を用いて傾向の分析を実施する。
○なぜなぜ分析を活用し、問題の真因を見出す。
・設計者や管理者などの視点で発見された内容（結果）とその原因、またその結果と連続的に因果関係を洗い出し、最後には真因を見つける。場当たり的対応にならないように、組織やプロジェクトで再発防止策をプロセスに織り込む。
※リスク分析との棲み分けを明確にして管理にあたる。
・リスク分析は問題が実際に発生する前に、対応策（リスク軽減策）を実施して問題が発生しないようする活動を取り扱う。リスク管理については、該当するヒントシートを参照のこと。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-PP-03(リスク計画)
・WS-PMC-02(課題管理)
・WS-PMC-03(リスク管理)
・SPEAK-IPA：S.8 問題解決
・CMMI® Ver1.2：プロジェクトの監視と制御 (PMC)、統合プロジェクト管理 (IPM)、リスク管理 (RSKM)
・共通フレーム 2007：2.8 問題解決
・ESPR 2.0：SUP6 問題解決管理

7. ふり返り等で参考となる文献等

・プロセス改善ナビゲーションガイド　～ベストプラクティス編～：ソフトウェア・エンジニアリング・センター
・新・管理者の判断力：C.H. ケプナー、B.B. トリゴー、上野 一郎訳 産能大出版
・ライト、ついてますか：ドナルド・C・ゴース、G.M. ワインバーグ著 木村 泉訳 共立出版

(WS-PP-01 規模の見積り)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトの作業範囲に基づき規模を見積もっている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○見積もった結果を提示することで、プロジェクトの規模、および工数やコストの根拠を提供できる。

○顧客と契約する際、プロジェクトの予算、スケジュール、納期などの数値を提供できる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○見積り前提の明確化と関係者による合意

・WBS（プロジェクト全体を細かい作業に分割した構成図）を作成して、プロジェクトで行う作業の範囲を定義する。

○合意した作業範囲を前提に見積もる。

・外部調達・再利用する成果物または成果物の構成要素を特定する。

・適切な手法を使用して、見積りに使用する成果物やタスクの属性を決定する。

・妥当性が確認されたモデルや履歴データに基づき見積もる。

○見積り結果の合意

・プロジェクト責任者の承認

・顧客を含めた関係先との合意

3．現実には実現することが難しい点

○見積り時点で、プロジェクトの開発範囲がすべて決まらない場合が多く、適切な見積りができない。

○プロジェクトの WBS 作成が不十分な場合、見積り作業のみによって精度を向上させるのは難しい。

○プロジェクトのコストや期間が決められている場合、分析的な見積りプロセスを実施しても、見積り時に先入観が入ってしまう。

○見積り時点で、仕様変更や欠陥対応など、追加作業が見積もられていないため、計画が破綻する。

○エンジニアの「勘と経験と度胸」など、見積りの方法の暗黙知を形式知として継承することは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○見積りの根拠が不明確なため、プロジェクト管理の目標値が信頼できなくなり、納期遅れやコスト超過などが常態化し、顧客の信頼を失う。

○見積り結果について関係者の合意がないため、プロジェクトの目標（ターゲット）が、いつのまにか顧客や上位管理者による役割（コミットメント）と変わってしまい、根拠がなく無理なプロジェクト計画ができあがる。

5.1 ソリューション例（取り組みやすい例）

○見積り技法、手順の確立

・ソフトウェア規模の見積りは、ソースコード、ファンクションポイント、仕様数、要求項目数、メソッド数、クラス数などがあり、一般的にソースコード行やファンクションポイントを用いる。
・過去の類似プロジェクトでかかった工数などを参考に実施に必要な工数を算出する。
・要求仕様件数とプロジェクトで作成する成果物のページ数を明確にし、それを適度な粒度に分割し、見積りの単位とする。
・見積りは、数える＞計算する＞判断する、の順に精度が上がり信頼できる。
・プロジェクトの見積りは、スキルをもった専門家に担当させる方法もある。

5.2 ソリューション例（深く取り組む例）

○見積り結果の検証をルール化

・信頼できる見積りを行うため、次の３ステップを利用する。
　ⅰ）何らかの見積り手法で対象システムの開発規模を見積もる。
　ⅱ）別の見積り手法を用いてこれを検証し、結果が大きく異なれば最初に戻ってプロジェクトの対象範囲などの前提条件を含めて見直す。
　ⅲ）結果がほぼ同じだったら、WBS により作業範囲の過不足と見積りの妥当性を裏付ける。

○見積りタイミングによる手法の使い分け

・プロジェクトの進捗に従い、見積り精度要求も高くなるので、タイミングによって手法を使い分ける。
・システム提案時：三点見積り法、デルファイ法、ファジー論理法、など
・システム要求分析時：ファンクションポイント法、モデル化技法、類似法、外挿法、ボトムアップ法

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-P31-1(顧客ニーズ抽出)
・WS-P32-1(システム要件のまとめ)
・WS-PP-02(計画立案)
・WS-PP-03(リスク計画)
・WS-O16-1(測定活動)
・SPEAK-IPA：O.1.3 プロジェクト管理
・CMMI® Ver1.2：プロジェクト計画策定 (PP)
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・ソフトウェア見積り：スティーブ・マコネル 日経 BP ソフトプレス 2006 年

(WS-PP-02(計画立案)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクト実行の計画を立案している。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○プロジェクトの初期段階で、プロジェクトの実現可能性を判断する材料に利用できる。
○プロジェクトの利害関係者間で、スケジュール、マイルストーン、WBS、依存関係、予算、資源、制約条件、成果物、リスクなどの共通理解のベースにできる。
○プロジェクトを進捗管理しコントロールするための基本計画として利用できる。
○プロジェクトの予実の差分を把握し、是正処置をとる際の基盤にできる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○各作業の WBS に基づき、プロジェクトと組織の間のインターフェース、各種管理目標としての基準値を明確にする（例：推進体制、QCD 目標値など）。
○作業項目単位で必要とする資源（ハードウェア、ソフトウェア、要員、スキル、期間など）を定量化し、プロジェクトのスケジュールを作成する。
○各作業の役割分担や責任を明確にし資源を投入し、進捗管理の仕組みを構築する。
○プロジェクト計画について、利害関係者により合意する。

3．現実には実現することが難しい点

○何年も自己流で開発している場合には、「計画書など作成しても意味がない」などと考える傾向があり、啓蒙するのが難しい。
○変更が発生する都度、毎回プロジェクト計画書を更新し、内外に周知して構成管理するのは、煩雑であり難しい。
○作成した計画の維持管理を実施しないため、プロジェクトの実態と乖離し使われない。
・プロジェクト管理ツールは、プロジェクト計画の主要な情報を前提とするが、プロジェクト計画とツールの情報が乖離した場合、区別して管理するのは難しい。
・プロジェクト開始時に計画を策定するが、その後進捗管理に使用しない場合がある。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○プロジェクトの予算やスケジュールなどの管理ができない。
○問題発生時の対応策に関する承認ルールなど計画されず、組織的プロジェクト運営ができない。
○プロジェクト計画をレビューしないため、組織パフォーマンスを改善し向上できない。
○プロジェクト情報の調査、実績の分析・評価を行うことが難しい。

5.1 ソリューション例（取り組みやすい例）

○プロジェクト計画書の記載項目

・プロジェクト計画書には、プロジェクトで必要となる以下の項目が記述される。
プロファイル、概要、方針、体制と役割、追跡のためのパラメータ、見積り、マスタスケジュール、WBS、品質計画、キャパシティとパフォーマンスの計画、構成管理計画、コミュニケーション計画、依存関係、作業成果物一覧、リスク、要員計画、課題管理、外部委託、重要決定事項の識別、トレーニング計画

○ EVM によるプロジェクト管理

・EVM を導入している組織では、過去の類似プロジェクトの AC(Actual Cost) を参照して、実現可能なプロジェクト計画が作成できる。

(Tips)

・最近の政府調達案件では、WBS（ワークブレークダウン・ストラクチャ）と EVM（アーンドバリューマネジメント）は、プロジェクト計画の必須項目である。

・WBS は、プロジェクトのコスト計画、スケジュール計画、品質保証計画、リスク管理計画の主要な入力情報となる。

5.2 ソリューション例（深く取り組む例）

○ PERT によるプロジェクト計画の確認

・プロジェクトのマスタスケジュールとマイルストーンを設け、WBS に基づいてクリティカルパスと依存関係を作成し、さらにプロジェクトのイベントなどの主要な情報を加えて、プロジェクト全体を俯瞰する PERT（Program Evaluation and Review Technique）図を作成する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-PP-01(規模の見積り)
・WS-PP-03(リスク計画)
・WS-PMC-01(プロジェクト進捗管理)
・SPEAK-IPA：O.1.3 プロジェクト管理
・CMMI® Ver1.2：プロジェクト計画策定 (PP)
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・絶対に遅延しないプロジェクト進捗管理：岡村 正司 日経 BP 社 2010 年
・組込みソフトウェア向けプロジェクトマネジメントガイド [計画書編]：ソフトウェア・エンジニアリング・センター
・プロジェクトマネジメント知識体系ガイド（PMBOK® ガイド）第 4 版：Project Management Institute, Inc. 2008 年

(WS-PP-03 リスク計画)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトのリスクを特定し、分析し、対応策を決定する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○計画段階からリスクを特定し対策を策定しておくことで、トラブルや事故時の損失などを回避もしくは、それらの低減を図ることができる。
○リスク予防策を実施するための予算を計上することで、リスクが健在化してもプロジェクトが赤字にならないように費用管理できる。
○リスクを特定し文書化することで、リスクの発生状況、リスク管理の変更について、関係者に報告することができる。
○リスクの発生に伴い、不測事態対応計画を実行することができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○リスク管理を行うためのガイドラインを示したリスク管理方針を定義する。
○リスク管理の責任を負うリスク管理者を割り当てる。
○各リスクの発生確率および影響度を見積もる。
○各リスクのしきい値を評価し、設定する。
○リスク監視のタイミングとトリガーポイントを設定する。
○リスク対応の優先度を決め資源を投入する。
○各リスクに対して受容、軽減、転嫁、回避など対応を決定する。
○リスクが発生した際の具体的な対応策とその有効性を定義する。
○リスク対応計画書として文書化する。

3．現実には実現することが難しい点

○リスクが顕在化しない場合には、リスク対策の策定に費やした工数は無駄になるので、十分な時間を投入してリスク管理をするのは難しい。
○経験のないプロジェクトマネージャは、リスクが現実のものになるとは思わず、現実味ある対応ができない。
○個々のエンジニアが抱え込んだリスクは暗黙知なので、洗い出すのは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○リスクが顕在化した場合の適切な対策やプロジェクトに与える影響度を予測できず、事故・災害時の不測事態対応計画を立てることができない。
○社員にリスク管理に関する意識が浸透せず、リスク管理の手法やスキルが蓄積されない。
○インシデントや事故が起きた際、自己流の思い付き的な危機対応、責任転嫁、影響を見ぬ振りをするなど、リスクに対する危機管理ができない。
○リスクを受容する場合でも、関係者が明確に把握できない。

5.1 ソリューション例（取り組みやすい例）

○組織レベルに応じたリスク管理

・リスクは、組織レベル、プロジェクトレベル、個人レベルに区別し、共通のリスクは、標準リスク管理テンプレートを作成して、発生確率、影響、トリガーポイント、監視のタイミング、対策などを予め検討しておく。

・組織レベルのリスクは、PMO が識別・評価し、金額や日程に換算して契約内容に反映させるとよい。設計書やプログラムのバックアップ、機密情報漏えいを防ぐための対策も組織のリスクである。

・プロジェクト計画書にはプロジェクト固有のリスクを特定して記載し、リスクごとの単票を作成するなどして、公式レビュー時に関係者間でリスク対応の役割と責任について合意する。

・個人レベルのリスクは列挙しにくく、担当者任せにすると、本来リスクとして記入すべき事象が欠落する危険があるので、気になったことをすべて挙げさせて、リーダやチームメンバが内容を検討して対処する。

5.2 ソリューション例（深く取り組む例）

○技法を用いたリスク管理

・プロジェクトの工程上のリスクは、クリティカルパスを探索し、すべてのタスクが最も早く着手できるスケジュール、遅くとも終わらなくてはならないスケジュールなどを特定し、納期へ影響する要素を特定する（クリティカルパス分析）。

・リスクの識別には、ブレーンストーミング、デルファイ法、インタビュー法、QC 手法、SWOT 分析、などを用いる。

・リスク分析には、シナリオ分析、デシジョンツリー分析、モデル、シミュレーションなどを用いる。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-PP-01(規模の見積り)
・WS-PP-02(計画立案)
・WS-PMC-03(リスク管理)
・SPEAK-IPA：O.1.3 プロジェクト管理
・CMMI® Ver1.2：プロジェクト計画策定 (PP)
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・プロジェクト・リスクマネジメント：ポール・S. ロイヤー、峯本 展夫訳 生産性出版 2002 年
・プロジェクトマネジメント「リスク」と「人材」の管理が成功への道、情報システム開発
・プロジェクトにひそむ３４のリスク：玉置 亮太、鶴岡 弘之、特集実践 日経ＩＴプロフェッショナル ２００３年８月号 PP.2

(WS-PMC-01 プロジェクト進捗管理)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトの進捗を確認する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○プロジェクトの進捗を確認することで、プロジェクトが全体のどの程度まで進んでいるかを把握し、このまま進めば納期に間に合うのか、間に合わない場合にはどの程度遅れるかが予測できる。

○プロジェクトの進捗率を測定することで、実績が計画から著しく乖離する前に予防的な措置を取れる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクトの進捗を監視し、進捗会議で関係者に報告すると共に、顧客やプロジェクト承認者への報告を行う。

○進捗会議ではスケジュールに加えて、作業工数、コスト、規模、トレーニング状況、情報保護関係者の関与状況についても報告する。

○進捗管理の報告は、前回の報告以降に測定したマイルストーン、活動、作業成果物、EVM のパラメータ、リスク・課題、資源、プロジェクトメンバのスキル、変更に関する追加情報が含まれる。

○スケジュールや作業工数の進捗報告は週一回、その他は少なくとも月一回で開催する。

3．現実には実現することが難しい点

○定例の進捗会議を週一回行う場合、作業の遅れを関係者が認識するのに一週間近く遅れることもある。一方、毎朝の会議など、あまり頻繁に定例会議を開催すると実際の作業時間がなくなり、進捗に悪い影響を及ぼすので、バランスをとるのが難しい。

○見積りのパラメータと進捗管理のパラメータを一致させるのは難しい。たとえば、規模の見積りにファンクションポイント（FP) を用いた場合でも、FP で進捗を報告するのは現実的ではない。多くは「このタスクは 3 日遅れ」などと、見積りとは異なる尺度を用いて進捗（スケジュール）管理をしている。

○進捗率が計画値を下回ったことで分かるのは進捗を遅らせる問題が発生したことであり、問題を予防できるわけではない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○早めに対処すれば防止できるプロジェクトの問題にもかかわらず、その問題を放置することで、開発現場が混乱して制御ができなくなり納期遅延を起こす。

○進捗を実際に監視していない場合、あるタスクが「90％終わっています」と報告した後、2 週間経って進捗が無いにもかかわらず、それでも上位管理者には「進捗は順調です」と報告するような状況になる。

5.1 ソリューション例（取り組みやすい例）

○進捗管理は、クリティカルパスを意識して、タスクの遅延がプロジェクト全体に及ぼす影響を考えて実施する。その際、リスク管理の視点から進捗率と生産性（標準の生産性ではなく現実の作業のトレンド）を考えて管理することが重要である。

○プロジェクトメンバのスキルが高くない場合、プロジェクトの進捗状況を正しく測定できず、問題への対応が後手になる。このような事態を回避するために、WBS のタスクは５人日以下に細分化し、タスクごとに担当者を割り当て、タスクの進捗率の測定方法をプロジェクトで統一する。

（たとえば）

着手 20% コーディング完了 50%、単体テスト完了 80%、最終レビュー終了 100%

○「若干遅れていますが、十分リカバリ可能です」など、実績工数がない進捗報告をされた場合には、実際の現場を監察する。

5.2 ソリューション例（深く取り組む例）

○進捗確認の際には、実績工数も記入できる WBS を作成し、常に正しい情報を用いて進捗を判断する。EVM を導入しているプロジェクトでは、進捗会議に合わせて EV、AC を入力して SPI、CPI を計算し、しきい値管理すると「見える化」「測れる化」による進捗管理が可能である。

○プロジェクト管理ツールを使うなどして、リアルタイムに進捗状況を共有（見える化）することで、報告の手間が省け、次の進捗確認までに問題が更に拡大するのを防ぐことが可能である。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-PMC-02(課題管理)
・WS-PMC-03(リスク管理)
・WS-O16-1(測定活動)
・SPEAK-IPA：O.1.3 プロジェクト管理
・CMMI® Ver1.2：プロジェクトの監視と制御
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・絶対に遅延しないプロジェクト進捗管理：岡村 正司 日経 BP 社 2010 年

(WS-PMC-02 課題管理)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

プロジェクトで起こった問題の是正処置を行う。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○目標が達成されてないときに、計画からの差異を是正し、プロジェクトで識別された問題の再発を防止する行動が取ることができる。

○発生した問題を是正することにより、プロジェクトをもともとの計画どおりに進めることができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクト計画書で計画した進捗パラメータの計画と実績の乖離を把握する。

○乖離の大きさを分析して、是正処置の必要性を判断する。

○特定された課題に取り組むために必要な是正処置を意思決定する。

○実施した是正処置の状況を利害関係者と共にレビューする。

○コミットメント（主に、コストと納期）の変更を協議する。

○是正処置を終結まで管理し、その有効性を判断する。

3．現実には実現することが難しい点

○小規模プロジェクトを経験的なプロジェクト管理だけで乗り切ってきたプロジェクト管理者に対して、プロジェクト運営の概念を変えるのは難しい。

○あまり厳密ではないプロジェクト管理の場合、作業成果物を軸とした WBS ではなく、所有物が担当者に所属する形になることが多く、成果物や WBS が見えない。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○プロジェクト計画書と現実の乖離がますます大きくなり、顧客ニーズ、要件、計画に対してまるで違った結果になる。

○若いプロジェクトメンバが、PDCA によるプロジェクト管理の技法を習得せず、場当たり的なプロジェクト管理をする悪い習慣を身にまとう。

5.1 ソリューション例（取り組みやすい例）

ガントチャートを用いた進捗管理のソリューション

○タスク依存関係を考慮してマスタースケジュールを作成し、日程表上にガントチャートを作成する。

○進捗会議やマイルストーンごとにガントチャートに縦線を入れ、タスクの進捗具合に合わせて稲妻曲線を入れて、各タスクの進捗度合いを把握（「このタスクは4日遅れ」など）する。

○プロジェクトや組織の経験則で、挽回可能な日数の「しきい値」を決めて、この値を超えた場合には、是正処置を発動する。

○対応策としては、軽い順に、以下のようになる。

・プロジェクトの範囲内での対応（時間外勤務など）→社内での対応（増員、交代、予算追加、火消しなど）→顧客と交渉（優先順位の変更、仕様変更、納期延期など）→サービスインの遅延→プロジェクト中止

5.2 ソリューション例（深く取り組む例）

EVM を用いた成果物の作成工数をベースラインとするソリューション

○作成する成果物単位ごとに作成工数を見積り、出来高の計画値（PV）を決定し、WBS を作成する。

○ WBS をベースラインとして利用し、タスクの進捗があるたびにベースラインの出来高（EV）とコスト実績値（AV）を入力して、実績 WBS を作成する。

○進捗確認時点（週一が望ましい）で計画値と実績の乖離を把握するため、スケジュール差異（SV）、スケジュール効率指数 SPI、コスト効率指数（CPI）を把握する。

○ SV がマイナス、SPI、CPI 両方の数値が悪化、BAC に対して EAC が増加傾向、などの場合には、乖離が発生していると判断する。

○問題、原因、解決策の立案（あるいは用意してあったプランを提示）

○先を予測して意思決定を行い、是正処置を履行する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S8-1(問題解決管理)

・WS-PMC-01(プロジェクト進捗管理)

・WS-PMC-03(リスク管理)

・SPEAK-IPA：O.1.3 プロジェクト管理

・CMMI® Ver1.2：プロジェクトの監視と制御

・共通フレーム 2007：3.1 管理

・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・絶対に遅延しないプロジェクト進捗管理：岡村 正司 日経 BP 社 2010 年

・解説：アーンド・バリュー・マネジメント：冨永 章 プロジェクトマネジメント学会 2003 年

(WS-PMC-03 リスク管理)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

リスクの状況を監視し、軽減策を実行する。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○プロジェクトが合意した責任範囲、品質目標、スケジュール、コスト目標を達成できない等の、プロジェクトに影響を与える潜在的なリスク要因を特定し、取り除くことができる。

○リスクの顕在化が不可避な場合にも、あらかじめ用意しておいたリスク対応策を履行することにより、速やかにリスクに対処できる。

○プロジェクト計画時にリスクを特定する習慣をつけることで、プロジェクトでの課題を明確にして、プロジェクト計画の策定に役立てることができる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○リスク管理計画の策定

○リスクの出所と区分を決定して、リスクパラメータを定義

○リスクを特定して、定性的・定量的なリスク分析

○リスクを分類し、優先付け

○リスク対応計画の作成

○リスクの監視して、リスク対応計画を履行

○リスクのクローズ

3．現実には実現することが難しい点

○リスク管理は失敗プロジェクトをなくすために実施される。しかしながら統計的に成功プロジェクトは 30％に満たないといわれるように、リスク管理を実施しても失敗プロジェクトを撲滅することは難しい。

○ IT 開発の共通のリスクなどの研究は進んでいるので、利用すると役に立つ。しかし、未知のものを含めて、すべてのリスクを予測し特定することは難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○リスクを見落とし、手当てが遅れてトラブルが拡大し、経営的に大きな影響を与える。

○あらかじめ準備しておけば、かえってプラスになったかもしれない良いリスクを見逃し、機会損失を被る。

○リスクのプラス面とマイナス面を理解することで、かえって不要なことをしなければよかった、というようなリスクを回避できる。

5.1 ソリューション例（取り組みやすい例）

プロジェクトレベルのリスク管理ソリューション

○プロジェクトが発足したら、リスク評価シートを準備して、リスク項目ごとにプロジェクトに影響を与える度合いを「高」「中」「低」で評価し、「中」以上のものに項番を付け、リスク内容を具体的に記述する。

○明確化した各リスクに対し、リスクの発生確率と発生した場合の影響の大きさから優先度を決定し、リスクのしきい値、監視のタイミング、リスクのトリガー、および管理担当者を選定する。

○リスクの対応戦略として、悪いリスクに対しては、回避、軽減、転嫁の対応策を選択し、良いリスクに対しては活用、共有、強化の対応策を選定する。

○プロジェクトの進行に従って、進捗会議、工程管理、公式レビューなどで陽にリスクを監視し、リスクがしきい値を超えた場合には、リスク対応策を履行する。

○リスクへの対応が完了した、または環境変化でリスクが消滅もしくは軽減された場合にクローズする。

5.2 ソリューション例（深く取り組む例）

組織レベルのリスク管理ソリューション　～火の見櫓編～

○組織傘下のプロジェクトのリスク発生を定点観測する「火の見櫓」を設置する。

○傘下プロジェクトが発足する際にプロジェクト計画書を受け取り、それぞれのプロジェクトのリスクについて合意して、トリガーやしきい値を確認する。

○「火の見櫓」は、プロジェクトの進行に従い、プロジェクト計画書の PERT 図やマスタースケジュールからクリティカルパスに影響するポイントを特定して、リスクを監視する。

○プロジェクトの進捗状況としきい値の合計が基準値を超えた場合に、「煙が出た」、「火がついた」とみなして、「火の見櫓」が「火消し隊」を投入して「消化」にあたる。

○特に、「火の見櫓」がプロジェクトの報告よりも先にリスクの顕在化を検知したケースは、必ず Lessons Learned として共有し、プロジェクトのしきい値の適正化を検討する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-PP-03(リスク計画)
・WS-PMC-01(プロジェクト進捗管理)
・WS-PMC-02(課題管理)
・SPEAK-IPA：O.1.3 プロジェクト管理
・CMMI® Ver1.2：プロジェクトの監視と制御
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・プロジェクトマネジメント知識体系ガイド（PMBOK® ガイド）第 4 版：Project Management Institute, Inc. 2008 年
・NTT ソフトウェアにおけるプロセス改善活動事例：亀田 康雄 SPES2004 2004 年

(WS-O16-1 測定活動)

1．目的と成果目標

(1) 取り上げた業務（プロセス）の目的

測定活動が実施されている。

(2) これを首尾よく達成すると何が実現できるか（めざす成果目標）

○測定と分析のためのメトリクスを示すことで、改善状況を客観的に捉え、プロセスの改善結果を定量的に測定できる。

※メトリクスの例:インスペクション実施率、規模当たり欠陥件数、工数当たり欠陥検出率、テスト欠陥検出率、テスト工数当たり欠陥検出率など

○「現状よりも大変な作業はしたくない」「何が良くなるのか実感できない」といった開発現場でありがちな雰囲気に対して、良くなっていることを実感させ、改善活動を奨励づけられる。

○プロセスの問題分析や解決、改善を実施するために必要な測定データの収集と分析の方法を明確にできる。

2．この業務（プロセス）を実現するために必要な実施事項（典型的な活動）

○プロジェクトで使用するメトリクスを組織的に確認する。

○上級管理者等の情報ニーズから導き出した測定目標を確立する。

○測定目標を基に、目標に対しての達成状況が判断できるメトリクスを確定する。

○測定に必要となる活動を特定し実施する。

○測定したデータを収集、格納、分析する。

○分析結果を評価し、意思決定に用いる。

○手順に従い、各関係者に測定と分析の結果を報告する。

3．現実には実現することが難しい点

○プロジェクトの効率だけを優先したデータを測定するのは、組織全体の観点から望ましくない。

○測定・分析には、数える、測る、などの人手による活動が不可避であり、ツールなどを活用しても自動化することは難しい。

○短期プロジェクトでは、データの測定は行えても、その結果を収集･分析し、次回のプロジェクトへ反映させるのは工数的に難しい。

4．この業務（プロセス）をうまく実施しなかった場合、起こるだろう問題

○測定できないものは、制御できない。

○改善効果を見える化できないと、「また作業が増えた」「何が良くなるの」といった現場のやらされ感が残る。改善効果を具体的に報告もできない。

○上位層が現状の改善目標や改善状況を客観的に把握できない。

5.1 ソリューション例（取り組みやすい例）

○測定の基本

・プロジェクト計画書に、測定の責任者の割当て、測定計画（目的、メトリクスの定義、実施方法など）を記載する。

・測定実施し、プロジェクト完了報告書に測定したメトリクス、分析結果を残す。

○ CMMI® 流の測定分析

・組織の情報ニーズ一覧を基に測定計画を作成し、プロジェクト責任者がレビュー・承認する。
（期間・工数・コスト・開発規模に関する情報、インスペクション、ピアレビュー情報、欠陥情報など）

・情報ニーズを特定し、目的を明確にすることで、必要なメトリクスを識別できる。

・プロジェクトの進行と測定計画に従ってメトリクスを測定し、結果を記録・格納する。組織ごと、プロジェクトごとに、しきい値や UCL/LCL などを定め、測定したメトリクスの値が目標値から大きく乖離している場合には、その原因を分析する。

・測定計画にしたがって、利害関係者に測定記録や分析結果を報告する。

・プロジェクト完了時にプロジェクト完了報告書の一部として測定実績を作成し、プロジェクト責任者がレビュー・承認し、組織の測定リポジトリに蓄積し、組織的な分析に利用する。

5.2 ソリューション例（深く取り組む例）

○ GQM 手法

・目的の設定：何がしたいのか、何のためにデータを収集するのか目的を明確化し、次にプロセス視点で各プロセスにて何を行えば改善になるかをゴールとして設定する。

・質問の設定：そのゴールは何を以って定量的に測定し判断できるのかを質問する。

・メトリクスの設定：観測するべきメトリクスを設定 (選定) する。

・メトリクス検証:プロジェクト期間を通じて、設定した各メトリクスを測定し、正しくゴールを導き出すかを検証する。

6. 関連する他の改善検討ワークシート、参考プロセス項目

・WS-S4-1(検証計画)
・WS-PP-01(規模の見積り)
・WS-PMC-01(プロジェクト進捗管理)
・SPEAK-IPA：O.1.6 測定
・CMMI® Ver1.2：測定と分析 (MA)
・共通フレーム 2007：3.1 管理
・ESPR 2.0：SUP1 プロジェクトマネジメント

7. ふり返り等で参考となる文献等

・21 世紀へのソフトウェア品質保証技術：菅野 文友、吉澤 正 監修　日科技連ソフトウェア品質管理研究会編、日科技連 1994

8.　改善計画・実績シート、ふり返りシート

改善計画・実績シート

改善目的・Goal・目標：	
効果確認方法・確認タイミング：	結果（実績）：

TASK	担当	カレンダー					

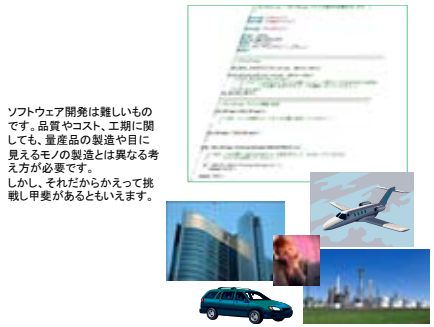
改善実施実績・改善結果へのふり返り

ふり返り観点： ①改善手段の効果／目標達成実績　②改善計画内容+実践実績　③対応中に感じたこと	
Keep：良かったこと・継続したいこと	Try：次にやりたいこと・次はどうすればよいか
Problem：うまくいかなかったこと・問題点	

おわりに

ソフトウェアエンジニアリングの現場定着は、いろいろな面で困難さを抱えています。しかし、現在の社会や産業がITとその中枢にあるソフトウェアで支えられていることは紛れもない事実で、その品質・生産性・確実性を向上させることは、ますます重要であると言えます。ITのセキュリティ問題は、いろいろな不幸が表に現れ、それなりに社会的な関心を呼びましたが、ITやソフトウェアのより広く深い諸問題がその背後にあることはなかなか認知されません。また、その結果としても、ソフトウェア開発の現場の混乱した状況やエンジニアへの過大な負荷もなかなか改善されません。

私たちのチームが開発したかったのは、現場への信頼感に基づいて、現場の創意性を高め、現場が仕事の成果に対する視野を広げていくのに役立つ支援ツールを開発することでした。また、これは現場の人たちに対する一つの挑戦状でもあると思っています。仕事の目標を明確にし、有効な方法をトライし、約束したことは実現するような仕事の仕組みづくりができるかどうか、ぜひ、実力を鍛え仕事環境を向上させることを楽しみながら、挑戦してみてください。私たちも、これからも現場の人たちと討議しながら、また自分自身もある種現場の一員として働きながら、考え、技術とスキルを向上させていきたいと思います。そのために、このメソッドのアプローチが役立つことを願います。



ソフトウェア開発は難しいものです。品質やコスト、工期に関しても、量産品の製造や目に見えるモノの製造とは異なる考え方が必要です。
しかし、それだからかえって挑戦し甲斐があるともいえます。

プロセス改善WG

委員一覧

主査	菊島　靖弘	東京海上日動システムズ株式会社
	赤坂　幸彦	エヌ・ティ・ティ・データ・システム技術株式会社
	安達　賢二	株式会社 HBA
	足立　久美	株式会社デンソー
	穴田　直也	株式会社大和コンピューター
	安倍　秀二	パナソニック株式会社
	臼杵　誠	富士通株式会社
	江崎　美保	株式会社日新システムズ
	尾形　俊彦	みずほ情報総研株式会社
	小川　清	名古屋市工業研究所
	串田　幸江	株式会社アジルコア
	倉持　俊之	IPA/SEC（TIS 株式会社）
	棗野　友靖	楽天株式会社
	小泉　浩	マイクロソフト株式会社
	神武　直彦	慶応義塾大学
	河野　文昭	株式会社アドヴィックス
	込山　俊博	日本電気株式会社
	近藤　聖久	三菱電機株式会社
	斎藤　吉正	新日鉄住金ソリューションズ株式会社
	阪本　太志	東芝デジタルメディアエンジニアリング株式会社
	塩谷　和範	VSE センター
	白坂　成功	慶應義塾大学大学院
	新谷　勝利	IPA/SEC
	砂塚　利彦	砂塚コンサルティングサービス株式会社
	田中　一夫	アイエックス・ナレッジ株式会社
	谷川　浩	トヨタ自動車株式会社
	徳永　享	富士ゼロックス株式会社
	戸村　哲	独立行政法人産業技術総合研究所

新井本　武士　株式会社東芝
錦見　美貴子　独立行政法人産業技術総合研究所
丹羽　武志　株式会社インテック
服部　祐二　ブラザー工業株式会社
伏見　諭　合同会社ソフデラ
堀田　勝美　株式会社コンピータジャパン
松原　友夫　松原コンサルティング
室谷　隆　IPA/SEC（TIS 株式会社）
和田　典子　ソニー株式会社

執筆者（敬称略）

安達　賢二　株式会社 HBA
安倍　秀二　パナソニック株式会社
桒野　友靖　楽天株式会社
斎藤　吉正　新日鉄住金ソリューションズ株式会社
阪本　太志　東芝デジタルメディアエンジニアリング株式会社
伏見　諭　合同会社ソフデラ
倉持　俊之　IPA/SEC（TIS 株式会社）
新谷　勝利　IPA/SEC
室谷　隆　IPA/SEC（TIS 株式会社）

編 者 紹 介

独立行政法人 情報処理推進機構 技術本部 ソフトウェア・エンジニアリング・センター

2004 年 10 月に独立行政法人 情報処理推進機構（IPA）内に設立されたソフトウェア・エンジニアリング・センター（SEC）は、エンタプライズ系ソフトウェアと組込みソフトウェアの開発力強化に取り組むとともに、その成果を実践・検証するための実践ソフトウェア開発プロジェクトを産学官の枠組みを越えて展開している。

［所在地］ 〒 113-6591 東京都文京区本駒込 2-28-8
文京グリーンコート センターオフィス
電話 03-5978-7543、FAX 03-5978-7517
http://sec.ipa.go.jp/index.html

●本書に記載されている「SPINA^{3}CH 自律改善メソッド」について

IPA ホームページよりダウンロードして利用できます。利用上の注意をご確認の上、ご利用ください。

●お問い合せについて

本書または掲載事例に関するお問い合せは、IPA ホームページの「お問い合わせ」からお願いします。「一般の問い合せ」のフォームに必須事項をご記入の上、送信してください。

ご記入の際には、お問い合せ内容とともに、冊子名、頁番号等をご記入いただくようお願いします。

SEC BOOKS
プロセス改善ナビゲーションガイド
～自律改善編～

平成 25 年 3 月 15 日　第 1 版第 1 刷発行

編　者 独立行政法人 情報処理推進機構 技術本部
ソフトウェア・エンジニアリング・センター

発 行 所 独立行政法人 情報処理推進機構

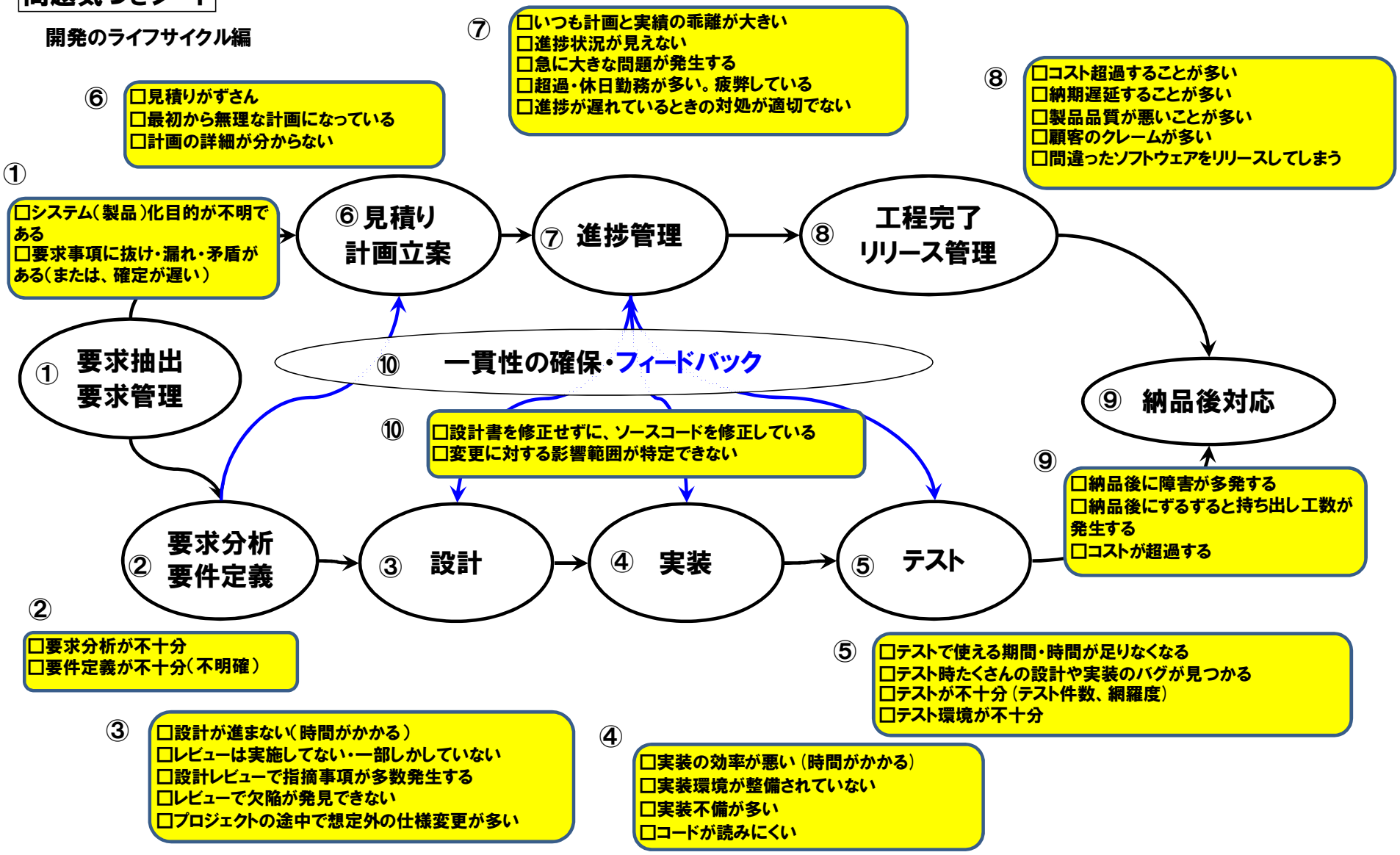
所 在 地 〒 113-6591
東京都文京区本駒込 2-28-8
文京グリーンコート センターオフィス
電話　03-5978-7501（代表）
ＵＲＬ　http://sec.ipa.go.jp/index.html

制作 株式会社バーテムズネットワークス　編集 株式会社トリフォリオ
ISBN 978-4-905318-20-0　Printed in Japan

問題気づきシート

2013/3/15 Ver.2.0

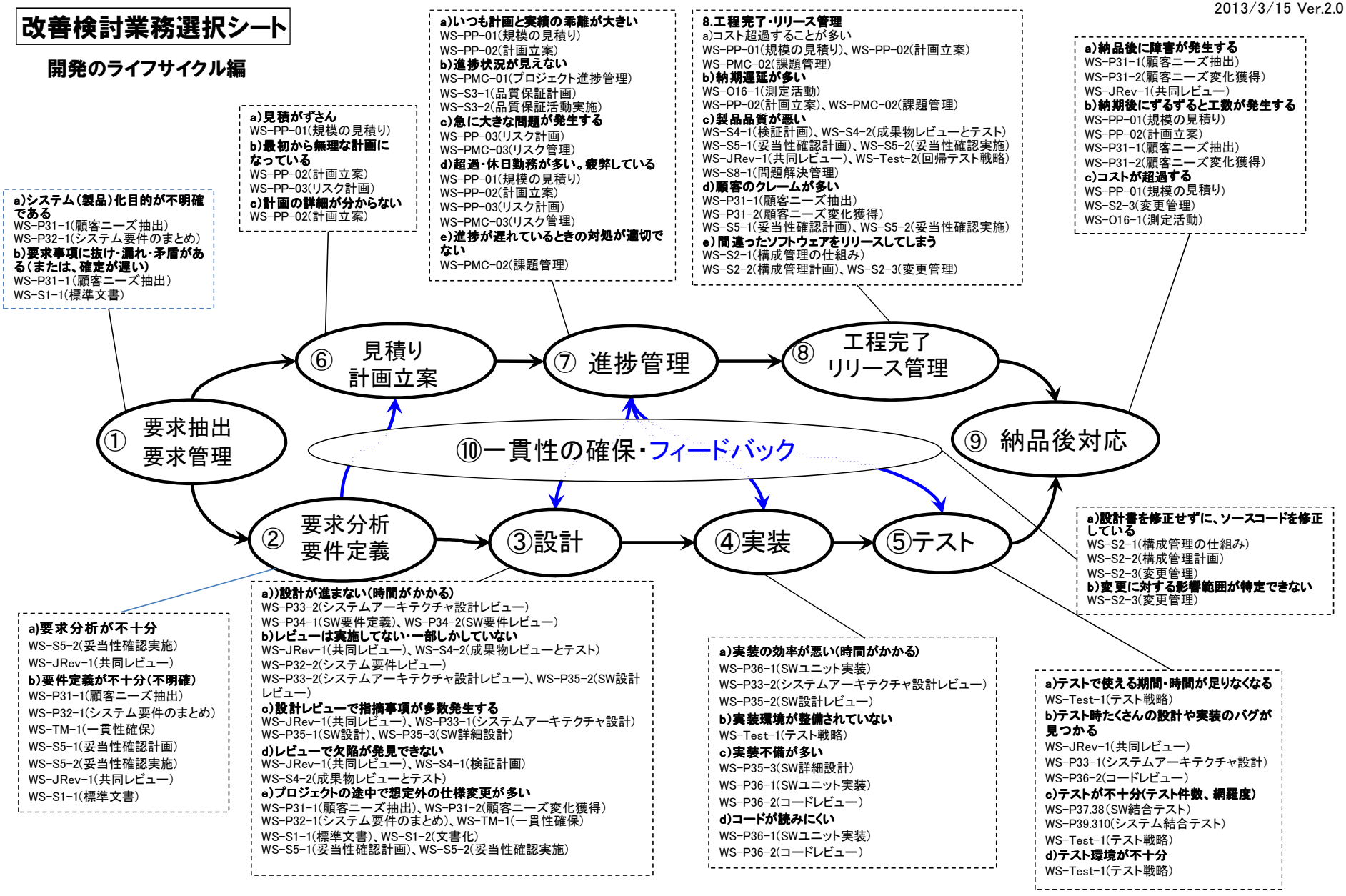
開発のライフサイクル編



改善検討業務選択シート

開発のライフサイクル編

2013/3/15 Ver.2.0



ISBN978-4-905318-20-0

C3055 ￥762E

9784905318200

定価（本体762円【税別】）

1923055007628

IPA 独立行政法人 情報処理推進機構
技術本部 ソフトウェア・エンジニアリング・センター

SEC−TN12−007