トレーサビリティ確保におけるソフト開発データからの

効果検証

実施報告書

2013 年 2 月

独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

はじめに

IPA/SEC では、ソフトウェア品質説明力を強化すべく様々な観点からの検討を実施してきました。その一環として、ソフトウェア品質を説明するための手法等について具体的な実施方法、そのための作業量、実施にあたっての課題等を整理し、実際にソフトウェア品質を説明する際の参考とできるようにするために、公募により、観点ごとに分けられた実験を別々に実施しました。本書は、それらの結果を、実験ごとにまとめた報告書のうちの１つです。

本報告書の実験は、「2011 年度システムエンジニアリング実践拠点事業」として、東芝情報システム株式会社に委託し実施しました。

「トレーサビリティ確保におけるソフト開発データからの効果検証」
【報告書】

独立行政法人情報処理推進機構

目次

図目次

表目次

用語定義

用語	解説
PRISMY	PRISMY（Project Information Sharing and Managing System） 不具合情報管理システム http://www.tjsys.co.jp/page.jsp?id=742
トレーサビリティ	考慮の対象となっているものの履歴、適用又は所在を追跡できること。[※ISO 9000（JIS Q 9000）の用語の定義] 本報告書では、ソフトウェア開発のトレーサビリティを示す。
トレーサビリティマトリクス	追跡可能性マトリクス。 プロジェクトが「要件のすべてを正しく設計し、実装し、評価し、納入した」と保証するためのトレーサビリティ（追跡可能性）を維持する仕組み。上位要件と下位要件との追跡可能性や、要件と下流の作業成果物との追跡可能性を、格子表（マトリクス）に整理したもの。
ESQR	ESQR（Embedded System development Quality Reference） 「組込みソフトウェア開発向け　品質作り込みガイド」

１．背景

ソフトウェア品質説明力は、社会的に極めて重要な課題として認識されてきており、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・センターでは、その強化のための取り組みを実施している。本実験では、説明力を強化すべき品質として、「製品の利用品質の確認や向上への利用者情報や利用情報の活用」という観点に注目し、それに伴い実施する作業とコストの関連についての検証を行った。

本報告書では、以下の構成にて実験結果について報告を行う。

- 実験の目的とその概要
- トレーサビリティについて
- 適用レベルとトレーサビリティの範囲について
- 実験における調査概要
- 実験における調査結果
- 調査結果における考察と評価
- 参考

２．実験の目的とその概要

ソフトウェア開発は、オフショア化や競争のグローバル化、ソースコード作成の自動化など、コスト削減の方策の取り込みが進んでいる。特に機器に組み込まれるソフトウェアでは、頻繁なソフトウェア更新が困難であるため、コスト削減とともに一層の品質向上が望まれている。

ソフトウェア開発のV字モデルにおいては、コスト削減及び品質向上の施策として、文書（仕様書／設計書）間のトレーサビリティを確保し、仕様からソースコードまでを追跡できるような仕組みが必要であると考えられる。

そのため、今回の実験では、高めの品質レベルが要求される通信ソフトウェアを対象に、システム設計から製造までの工程間で、トレーサビリティを確保することで実現できる工数削減（コスト削減）の評価を行い、その結果を、ソフトウェア品質説明力強化に向けての参考データとして提供するものである。

２．１　実験テーマ

トレーサビリティ確保によるコスト削減（手戻り作業の削減）効果を検証する。

２．２　実験分野

今回の実験では、以下の実験分野を選択した。

大分類（イ）：コスト評価

小分類（B）：利用品質の確認や向上への、利用者情報や利用情報の活用

２．３　実験対象

今回の実験対象は、以下の 2 つのソフトウェア開発である。

- 一般組込み機器製品に搭載される通信ソフトウェア
- 車両搭載用通信プロトコルスタックソフトウェア

２．４　実験尺度

トレーサビリティ確保による後戻り工数のコスト（削減分）、トレーサビリティ対策による増加（付加）工数の割合を測定する。

２．５　実験目標

トレーサビリティ確保及び上流工程での品質確保による効果を明確にする。

３．トレーサビリティについて

３.１ソフトウェア開発とトレーサビリティ

組込みソフトウェア開発は、組込み製品の多機能化・多様化に伴い、開発するソフトウェアは肥大化、複雑化している。組込み製品の多様化への対応は、派生開発によるものも多く、１ つの機能が発展しながら複数の製品に展開されることも少なくない。そのため、ある機能で1つの不具合が発生すると、その影響は派生開発された類似製品にも影響することになり、不具合が想定外の範囲に拡がることも考えられる。

ソフトウェアの品質を保証する上で、ソフトウェア提供者は機能的で信頼性が高く、効率的なソフトウェアを提供することを考えなくてはならない。また、機能性を追及するだけではなく、高性能で安全性（信頼性）が高い等、高い品質を保証した製品開発を行う必要がある。

そのためには、ソフトウェアへの要求を的確に定義し、これを製品の中に確実に作り込むことが重要になる。そこでは、欠陥を作り込むことを予防することがより重要であり、これにはソフトウェアのトレーサビリティが深くかかわってくる。

ソフトウェアの品質保証では、以下の要素を担保することが求められている。

（1）有益であり安全性が高いこと

（2）システム設計は要求仕様を満足していること

（3）実装がシステム設計に合致していること

（4）要求仕様がシステムテストで確認されること

これらの要素を、ソフトウェア開発におけるV字モデルでの各仕様書／設計書の関連図（図 1）で示す。

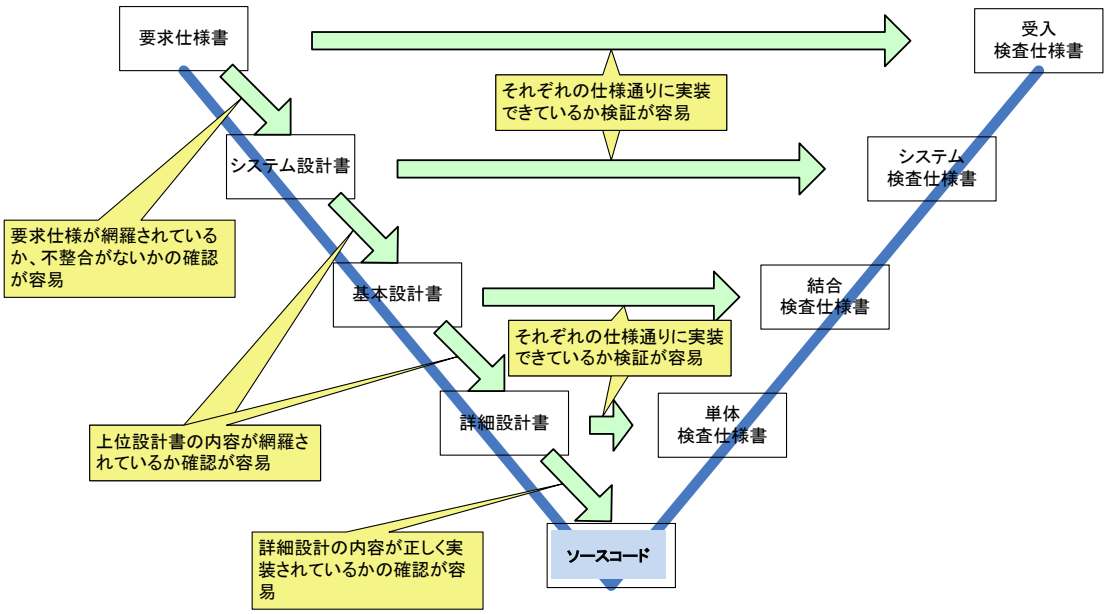


図 1 V字モデルでの設計書／仕様書間のトレーサビリティの例

開発の各工程の成果物間のトレーサビリティが確保されることで、成果物が関連付けされ、その結果、内容の確認が容易となり、レビュー時間が短縮できると予測できる。また、各段階での仕様の漏れや不整合の検出が容易になる。

その結果として、各工程のレビューで前工程の要件が正しく取り込めているか否かが確認でき、欠陥（不具合）の作り込み防止にもつながる。さらに、品質が向上することで、確認の手間（工数）を削減できる可能性がある。

３.２ トレーサビリティとは

一般には、ソフトウェアのトレーサビリティとして、以下の 2 点が挙げられる。

（1）製品に関するトレーサビリティ

（2）文書及びデータに関するトレーサビリティ

（1）は、ソフトウェアアイテム（ソフトウェアの部品の最小構成単位）が、製品にどのように組み込まれているかを追跡できることである。（2）は、ソフトウェアの要件が製品のどの部分で実現されているかを追跡できることを意味し、逆にできあがっているソフトウェアの構成部品（実行モジュール等）が、どのソフトウェア要件を実現するためのものかを追跡できることも意味する。本実験では、「（2）文章及びデータに関するトレーサビリティ」について検証を行う。

ソフトウェア開発では、開発途中での仕様変更は多いので、変更に関連している他の部分を修正しなければならない。また、ソースコードの作成中に設計上の問題を発見した場合、その基になっている設計にまで立ち返って仕様を修正することになる。

このようにソフトウェア開発では、個々の成果物がお互いにどのように影響し合っているかを把握することが重要であり、この特性がトレーサビリティである。

トレーサビリティを確保することで、ソフトウェアの一部を修正したときに、その影響を受けるソフトウェアの構成部品及び関連する箇所が特定可能なため、必要なソフトウェアの修正漏れを防ぐことができる。

トレーサビリティが確保された文書の作成工数は、トレーサビリティなしの文書の作成工数に比べて、多くなることが推測される。

しかし、トレーサビリティを意識して作成された文書は、上位文書との関連付けができているので、漏れや不整合が少なくなり品質が向上すると考えられる。また、関連する文書の参照も容易であることから、第 3 者によるレビューも容易となり、レビューの質や効率が向上することも考えられる。

これらの効果により、品質が向上した文書は、製造工程や検証工程での不具合発生を予防することになり、不具合対応工数の削減効果があると考える。

３.３ トレーサビリティ対策工数の試算

今回の実験にあたり、成果物にトレーサビリティ情報を付加するための工数（トレーサビリティ対策工数）の見積りルールを策定した。基本的に、トレーサビリティ情報を確認／付加する工数は、従来工数の 30%と仮定した。

ベースとなる工数の算出にあたって、コード作成の工数については、SEC BOOKS「ソフトウェア開発データ白書 2010-2011」の生産性を参考とし、設計書作成ならびにレビューの工数を決めるための品質指標については、SEC BOOKS「組込みソフトウェア開発向け 品質作り込みガイド」(以下、ESQR)の品質指標を参考とした。

各作業工数の算出式を表 1に示す。

表 1 トレーサビリティ対策工数見積り

作業	算出式	備考
コード作成	コード作成時間 =（処置ステップ数/8.47）- （処置ステップ数/12.1）	参考：ソフトウェア開発データ白書 2010-2011 「主開発言語別の SLOC 生産性基本統計量（改良開発）」の C 言語
設計書作成	設計書作成時間 ＝処置ステップ数×28.6	参考：ESQR 設計書ボリューム率 処置ステップ数：KLOC 単位 【前提：1 ページ／時間】
設計書 レビュー	設計書レビュー時間 =処置ステップ数×13.42	参考：ESQR 設計レビュー作業実施率 処置ステップ数：KLOC 単位
コードレビュー	コードレビュー時間 ＝処置ステップ数×6.71	参考：ESQR コードレビュー作業実施率 処置ステップ数：KLOC 単位

なお、算出の基礎となる「処置ステップ数」が極めて小さく、前述の品質指標では適切な値を算出できない場合、該当工数は 0.5 時間と仮定する。

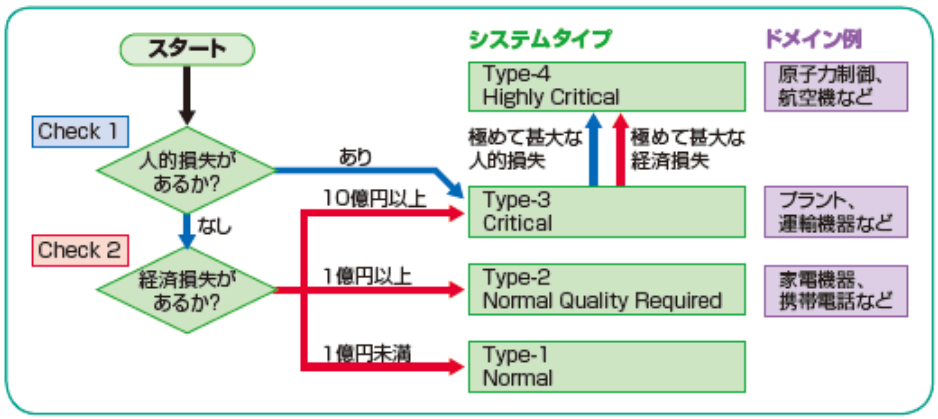
それぞれの算出式の定義手順については、次ページ以降の3.3.1～3.3.5を参照のこと。

３.３.１ 品質指標の定義

ESQR の品質指標を参考するにあたり、ESQR の手順に従ってシステムプロファイリング（SCP：System Characteristics Profiling）及びプロジェクトプロファイリング（PCP：Project Characteristics Profiling）を実施した。

（1）システムプロファイリング（SCP）

プロジェクトプロファイリングでは、図 2の手順に従いシステムタイプを決定する。



*「組込みソフトウェア開発向け 品質作り込みガイド」

図 2 SCP チェック手順

今回の調査の対象プロジェクトは、車両搭載や携帯電話搭載を想定した通信システムであるため、以下のように仮定し、**Type-2[Normal Quality Required(NQ)]**と定義した。

- 人的損失：なし
- 経済損失：1 億円以上

システムタイプについては、図 3を参照のこと。

システムのタイプ：SCP (System Characteristics Profiling)

Type-1：Normal：通常レベルの品質や信頼性が求められるシステム

Type-2：Normal Quality Required：通常以上に高い品質や信頼性が求められるシステム

Type-3：Critical：高い品質や信頼性が求められるシステム

Type-4：Highly Critical：きわめて高い品質や信頼性が求められるシステム

*「組込みソフトウェア開発向け 品質作り込みガイド」

図 3 システムのタイプ分け

（2）プロジェクトプロファイリング（PCP）

次に図 4のプロジェクトプロファイル表を使用し、品質指標の補正係数を求める。

今回の調査では、ファクター①～③をプラス補正、その他を基本と仮定し、**補正係数を+3** と定める。

ファクター		マイナス補正（－1）		基本		プラス補正（＋1）
①	ソフトウェア規模	□	極めて小さい	□ 普通	□	極めて大きい
②	ソフトウェアの複雑さ	□	極めて単純	□ 普通	□	極めて複雑
③	システム制約条件の厳しさ	□	制約ゆるい	□ 普通	□	制約厳しい
④	仕様の明確度合い	□	極めて明確	□ 普通	□	明確になっていない
⑤	再利用するソフトウェアの品質レベル	□	極めて高品質	□ 普通	□	極めて品質低い
⑥	開発プロセスの整備度合い	□	整備できている	□ 普通	□	整備できていない
⑦	開発組織の分業化・階層化の度合い	□	開発組織が単純	□ 普通	□	開発組織が複雑
⑧	開発メンバのスキル	□	メンバスキル高い	□ 普通	□	メンバスキル低い
⑨	プロジェクトマネージャの経験とスキル	□	PMスキル高い	□ 普通	□	PMスキル低い
⑩	システム障害時のメーカ側損失額	□	極めて小さい	□ 普通	□	極めて大きい
小計						
合計ポイント数						

*「組込みソフトウェア開発向け 品質作り込みガイド」

図 4 プロジェクトプロファイル表

3.3.2 コード作成時間

「ソフトウェア開発データ白書 2010-2011」に記載されている「主開発言語別の SLOC 生産性基本統計量（改良開発）」（表 2参照）の C 言語の標準偏差の値（12.1SLOC/人時）を参考に、コード作成工数を算出する。

なお、コードへトレーサビリティ情報を付加するための工数は、この生産性が 30%低下する（3.3参照）と仮定して算出した。したがってトレーサビリティを確保するためのコード作成時間を以下のように求める。

【トレーサビリティ情報付加時生産性】=12.1×0.7

＝8.47

【コード作成時間】=（処置ステップ数/8.47）-（処置ステップ数/12.1）

表 2 主開発言語別 SLOC 生産性の基本統計量（改良開発）

[SLOC ／人時]

主開発言語	N	最小	P25	中央	P75	最大	平均	標準偏差
b：COBOL	87	0.0	2.3	4.6	6.4	40.8	6.1	7.4
g：C	67	0.1	1.4	4.1	6.9	69.9	7.8	12.1
h：VB	56	0.0	1.7	4.8	7.9	39.9	7.8	9.5
q：Java	132	0.0	2.2	4.0	7.4	27.0	6.0	6.0

*「ソフトウェア開発データ白書 2010-2011」

３.３.３ 設計書作成時間

図 5に記載の ESQR の「設計書ボリューム率」を参考に、設計書枚数を不具合情報管理システム（以下、PRISMY）の「処置ステップ数」から算出する。

ＩＤ	PD11				
名称	設計書ボリューム率				
略称	RDDV				
名称（英語表記）	Ratio of the Design Document Volume				
参考値	N	NQ	C	HC	補正ベース値
	9	19	29	39	10.00
参考値の範囲	0.00 ～ 19.00	9.00 ～ 29.00	19.00 ～ 39.00	29.00 ～ 49.00	
計測単位	Page/KLOC				
許容誤差	有効数字上位２桁まで （14.8 Page/KLOC の場合、14Page/KLOCとする）				
指標値の意味	● 設計書類のボリュームをプロジェクト規模とのバランスで表します。 ● 本指標値が小さい場合にはドキュメンテーションが不十分であると考えることができます。				
計算方法	設計書ボリューム/ソースコード全行数 RDDV = VODD/TLOC				
指標の利用法	● 本指標値が参考値より小さい場合には、設計ドキュメントが不足している可能性があります。設計ドキュメントが不足する理由としては、①そもそも設計の検討が不十分でドキュメントが作成できていない ②設計の検討などは十分なものの、作業結果としての整理が十分でないなどが考えられます。いずれの場合でも、再度ドキュメントのレビューを行うなどして、ドキュメント内容の妥当性を確認することが良いでしょう。				
備考： 参考値の解釈	● 本指標の参考値では、あまり高い品質を要求されない普通のソフトウェア（Type-1：N）を考えた場合、要求仕様書の1ページに対して、その内容をきちんと設計に反映させた場合、設計書のボリュームは要求仕様書の3倍程度が適切との立場で参考値を設定しています。 ● また、システムの特質によってもボリュームが変わります。たとえば、ユーザインタフェースの多いシステムでは設計書のボリュームもまた大きくなる傾向にあるでしょう。				

図 ５ 設計書ボリューム率

- 【システムタイプ】Type-2：NQ
- 【プロジェクトプロファイル補正係数】+3 （ソフトウェアの規模、複雑性、制約条件）
- 【設計書ボリューム率】22
- 【トレーサビリティ情報付加分加算】22 × 1.3=28.6
- 【ページ数（設計書作成時間）】=28.6 × 処置ステップ数（※）

※ 設計書へトレーサビリティ情報を付加するための工数は、1 ページあたり 1 時間と仮定した。

３.３.４ 設計書レビュー時間

図 6に記載の ESQR の「設計レビュー作業実施率」を参考に、設計書レビュー時間を PRISMY の「処置ステップ数」から算出する。

ＩＤ	PR21
名称	設計レビュー作業実施率
略称	ERDR
名称（英語表記）	Execution Ratio of the Design Review

参考値	N	NQ	C	HC	補正ベース値
	7.20	9.60	12.00	14.40	2.40
参考値の範囲	4.80 ～ 9.60	7.20 ～ 12.00	9.60 ～ 14.40	12.00 ～ 16.80	
計測単位	人時／KLOC				
許容誤差	有効数字上位3桁まで				
指標値の意味	● 設計のレビューにどれだけ工数をかけているかをプロジェクト規模とのバランスで表します。 ● 安全性、信頼性を要求されるシステムほどレビューにかける工数は多くなります。				
計算方法	設計レビュー工数/ソースコード全行数 ERDR＝REDE/TLOC				
指標の利用法	● 設計のレビューには一定以上の適切な工数をかけて行うことが求められます。 ● 本指標値が参考値より小さい場合には、設計のレビューが甘く結果として設計の漏れや抜けが残りやすくなります。 ● 逆に、本指標値が参考値より大きい場合には、仕様が曖昧なために設計レビューがしづらくなっている、設計構造が複雑すぎる、機能が多い、といったプロジェクト上のリスクが入っていることが考えられます。				
備考： 参考値の解釈	● 本指標の参考値はすべて新規に作成した場合を想定して算出しています。 ● ソフトウェアの再利用率が高い場合はレビュー対象が規模に対して少なくなると考えられるので、参考値より少なく見積もることができます。 ● また、本指標はプロジェクトの求める品質レベル、構造の複雑さ、異常系の処理などにより、値が変化します。				

図 ６ 設計レビュー作業実施率

- 【システムタイプ】Type-2：NQ
- 【プロジェクトプロファイル補正係数】+3 （ソフトウェアの規模、複雑性、制約条件）
- 【設計レビュー作業実施率】10.32
- 【トレーサビリティ情報付加分加算】10.32 × 1.3=13.42
- 【設計書レビュー時間】=13.42 × 処置ステップ数

今回の対象データでは、設計書レビュー時間に関するデータが少ないため、ここで算出した設計書レビュー時間を、設計書レビュー時のトレーサビリティ情報を確認するための工数と仮定した。

３.３.５ コードレビュー時間

図 7に記載の ESQR の「コードレビュー作業実施率」を参考に、コードレビュー時間を PRISMY の「処置ステップ数」から算出する。

ＩＤ	PR22				
名称	コードレビュー作業実施率				
略称	ERCR				
名称（英語表記）	Execution Ratio of the Code Review				
参考値	N	NQ	C	HC	補正ベース値
	3.60	4.80	6.00	7.20	1.20
参考値の範囲	2.40 ～ 4.80	3.60 ～ 6.00	4.80 ～ 7.20	6.00 ～ 8.40	
計測単位	人時／ KLOC				
許容誤差	有効数字上位３桁まで				
指標値の意味	● ソースコードのレビューにどれだけ工数をかけているかをプロジェクト規模とのバランスで表します。 ● 安全性、信頼性を要求されるシステムほどレビューにかける工数は多くなります。				
計算方法	コードレビュー工数/ソースコード全行数 ERCR＝RECO/TLOC				
指標の利用法	● ソースコードのレビューには一定以上の適切な工数をかけて行うことが求められます。 ● 本指標値が参考値より小さい場合には、対象ソースコードの必要な部分に十分目が行き届かずソースコード上の不具合が残りやすくなります。 ● 逆に、本指標値が参考値より大きい場合には、ソースコードの理解性、保守性等が良くない、レビューの効率が良くないなどのプロジェクト上のリスクが入っていることが考えられます。				
備考： 参考値の解釈	● 本指標の参考値はすべて新規に作成した場合を想定して算出しています。 ● ソフトウェアの再利用率が高い場合はレビュー対象が規模に対して少なくなると考えられるので、参考値より少なく見積もることができます。 ● また、本指標はプロジェクトの求める品質レベル、構造の複雑さ、異常系の処理などにより、値が変化します。				

図 7 コードレビュー作業実施率

- 【システムタイプ】Type-2：NQ
- 【プロジェクトプロファイル補正係数】+3 （ソフトウェアの規模、複雑性、制約条件）
- 【設計レビュー作業実施率】5.16
- 【トレーサビリティ情報付加分加算】5.16 × 1.3=6.71
- 【コードレビュー時間】=6.71 × 処置ステップ数

今回の対象データでは、コートレビュー時間に関するデータが少ないため、ここで算出したコードレビュー時間を、コードレビュー時のトレーサビリティ情報を確認するための工数と仮定した。

４．適用レベルとトレーサビリティの範囲について

今回の実験結果を評価するにあたり、プロジェクトの各成果物間のトレーサビリティを取る範囲を5段階に分類し、これを「適用レベル」として、表 3のように定義した。

ソフトウェアのトレーサビリティは、本来であればV字モデルの左側の設計工程の成果物と右側の検証工程の成果物のトレーサビリティも含まれる。しかし、今回の実験では設計要因の不具合が対象なので、設計工程のみの成果物のトレーサビリティにフォーカスするため、検査工程の成果物とのトレーサビリティは作業項目からは除外した。

適用レベル L1～L4 については、トレーサビリティの始点を要求仕様書としているが、今回の調査対象プロジェクトでは、要求仕様書は実験の対象外としているため、適用レベル L1 については、トレーサビリティが確保されているものとして評価を行う。適用レベル L0 は調査対象プログラムの現状なので、今回の評価対象の適用レベルは L1／L2／L3／L4 となる。

表 3 適用レベルとトレーサビリティの範囲

適用レベル	トレーサビリティの範囲
L0	トレーサビリティ：全くなし
L1	要求仕様書⇒システム設計書
L2	要求仕様書⇒システム設計書⇒基本設計書
L3	要求仕様書⇒システム設計書⇒基本設計書⇒詳細設計書
L4	要求仕様書⇒システム設計書⇒基本設計書⇒詳細設計書⇒ソースコード

トレーサビリティの範囲が拡がることに比例して、成果物の作成工数は増加すると推測される。一方、各成果物の品質は向上するため検証工程での不具合発生件数は減少し、不具合対応工数も減少する。

成果物にトレーサビリティ情報を付加するための工数（トレーサビリティ対策工数）と不具合対応工数の差を今回の実験の評価の尺度とする。

品質向上の観点からは、トレーサビリティはできるだけ広範囲で確保するべきだが、プロジェクトで求められる品質と開発コストのバランスで、トレーサビリティ確保の範囲は変化する。

特定の条件下ではあるが、品質向上に充てられる工数増分と品質向上効果による不具合対応工数の削減分の関係を明らかにする。

４.１適用レベル L0 のトレーサビリティの範囲

適用レベル L0 では、トレーサビリティを確保していない状態として扱う。

４.２適用レベル L1 のトレーサビリティの範囲

適用レベル L1 では、図8に示す通り要求仕様書～システム設計書間のトレーサビリティが確保されている状態として評価する。本来、顧客要求を取りまとめて要求仕様書を作成する要求管理の部分についてもトレーサビリティを確保することが好ましいが、今回の実験対象プロジェクトではこの部分のデータが存在しないため、対象外とする。

適用レベル L1 では、顧客の要求する機能仕様が漏れなく記述されている要求仕様書に対して、システムがどのように対処するのか、適切な解決方法が矛盾なく、漏れなくシステム設計書に記述されている状態である。

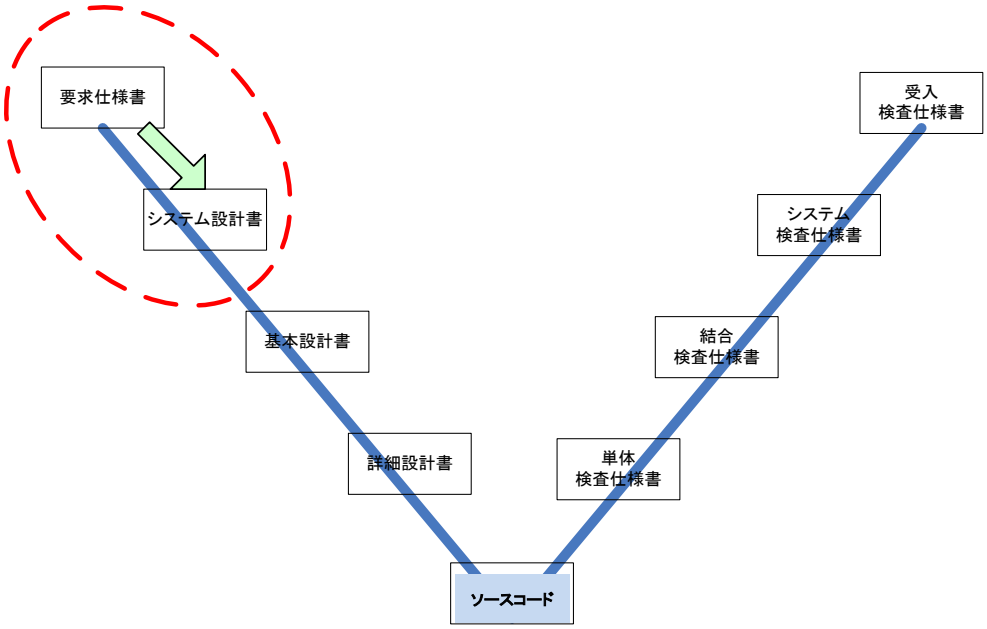


図 ８ 適用レベル L1 のトレーサビリティ範囲

４.３適用レベル L2 のトレーサビリティの範囲

適用レベル L2 では、図9に示す通り要求仕様書～基本設計書間のトレーサビリティが確保されている状態として評価する。

このレベルでは、システム設計書に記述されている解決方法に対して、システムの構築方法と解決方法の具体的な形が矛盾なく、漏れなく基本設計書に記述されている状態である。

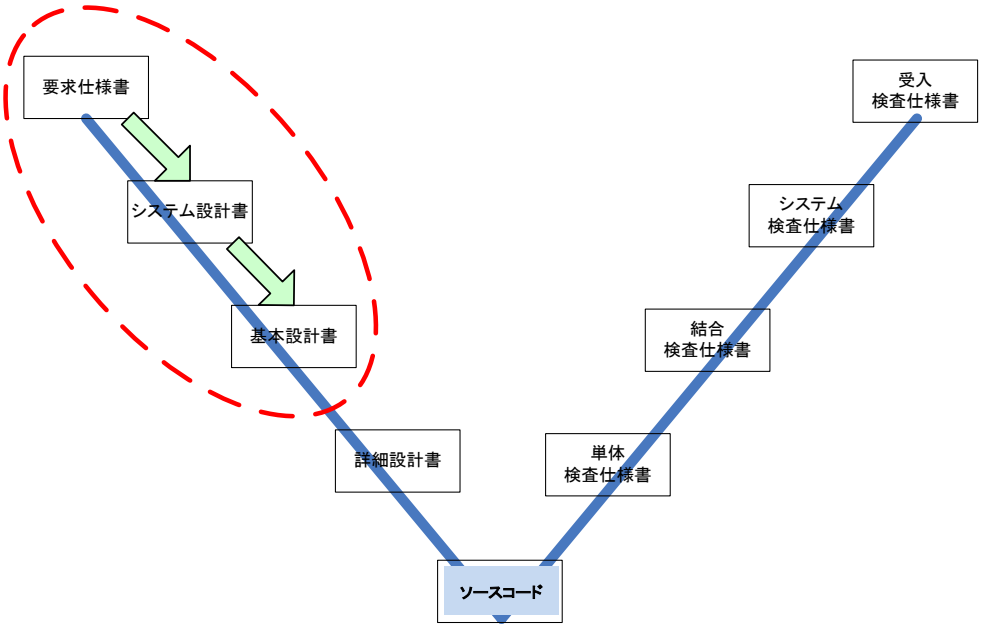


図 ９ 適用レベル L2 のトレーサビリティ範囲

４.４適用レベル L3 のトレーサビリティの範囲

適用レベル L3 では、図１０に示す通り要求仕様書～詳細設計書間のトレーサビリティが確保されている状態として評価する。

このレベルでは、基本設計書に記述されている解決方法の仕組みに対して、それぞれの仕組みの具体的な実現方法が矛盾なく、漏れなく詳細設計書に記述されている状態である。

プログラム作成にあたっては、詳細設計書の他に外部との関係を記述したインターフェース仕様書等が必要な場合もあるが、今回の適用レベルの定義ではこれらの仕様書／設計書は詳細設計書に包含するものとする。

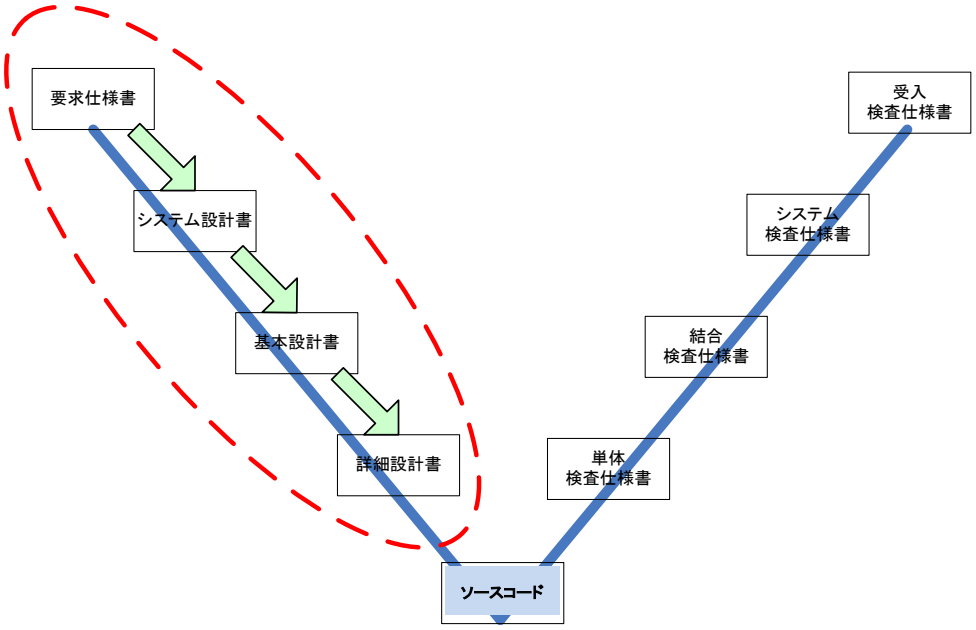


図　１０　適用レベル L3 のトレーサビリティ範囲

４.５適用レベル L4 のトレーサビリティの範囲

適用レベル L4 では、要求仕様書～ソースコード間のトレーサビリティが確保されている状態として評価する。

このレベルでは、詳細設計書に記述されている実現方法が、漏れなくプログラムコードに変換されている状態である。

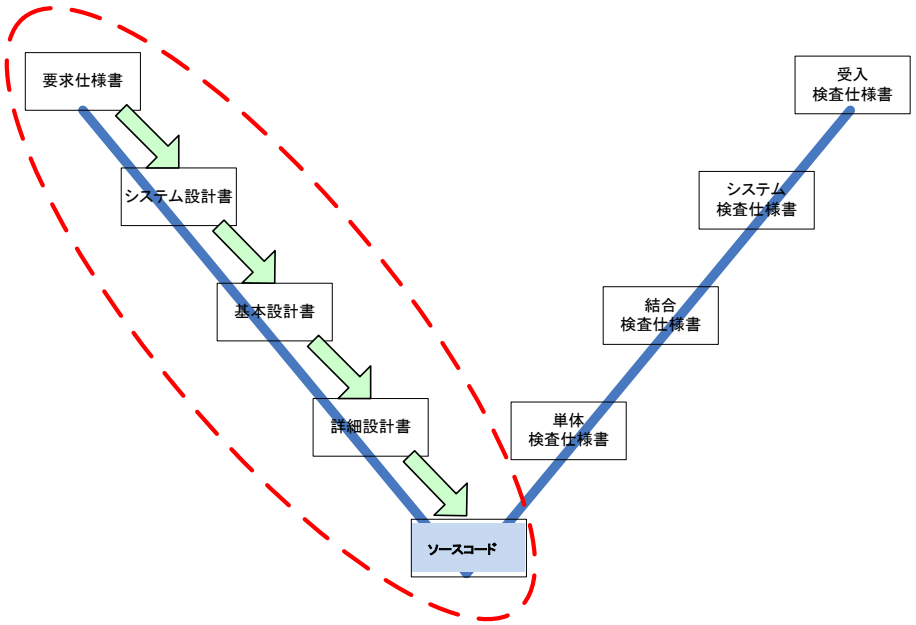


図 11 適用レベル L4 のトレーサビリティ範囲

５．実験における調査概要

５.１調査概要

【調査対象】一般組込み機器製品に搭載される通信ソフトウェア

車両搭載用通信プロトコルスタックソフトウェア

【ライフサイクルの範囲】 システム設計～製造

【調査の目的】 トレーサビリティ確保によるコスト削減及び品質向上効果の検証

【調査の分類】 コスト評価

【評価の尺度】 人月： 以下の工数の割合を測定

・トレーサビリティ確保による後戻り工数（削減分）

・トレーサビリティ対策のための増加（付加）工数

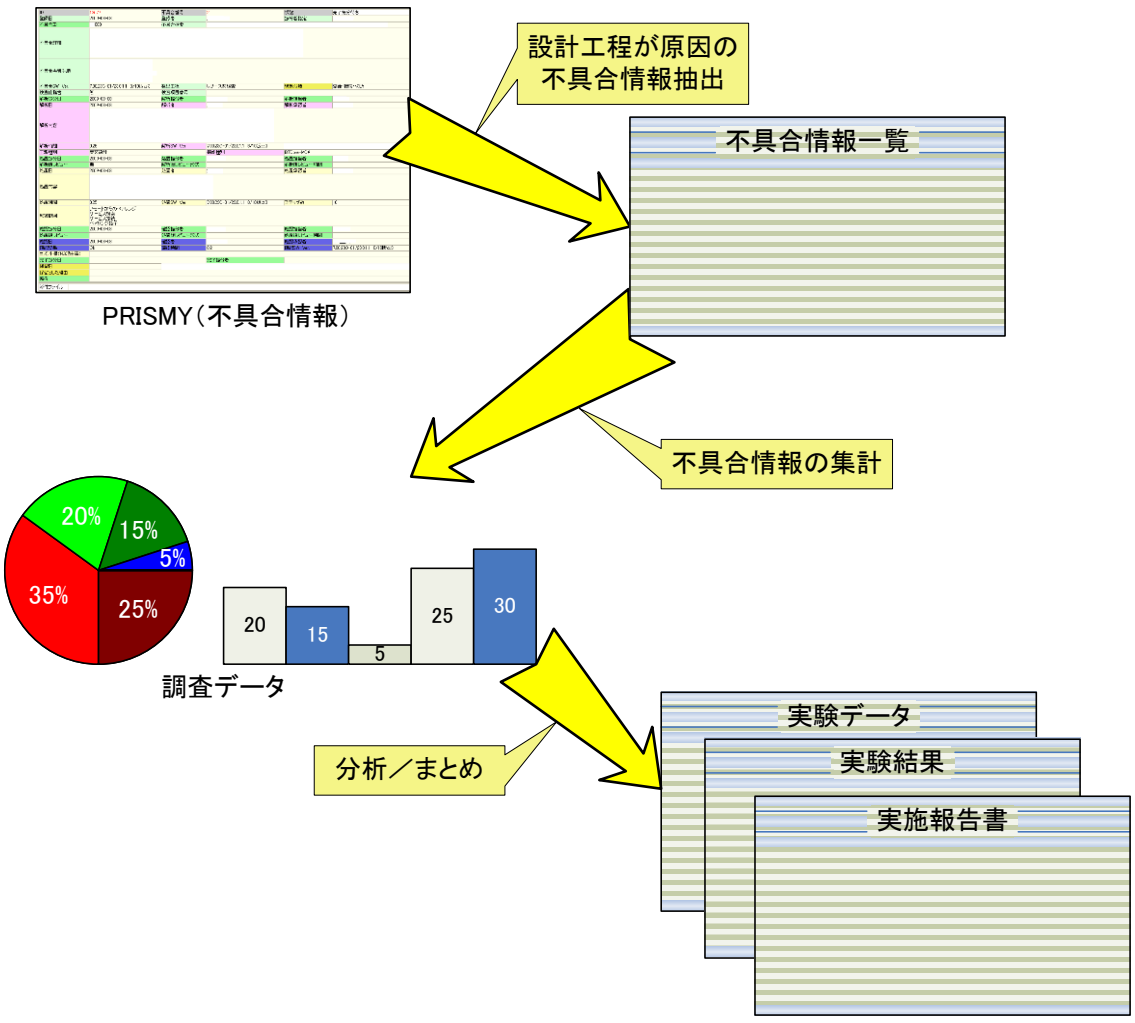


図 12 調査概要図

５.２調査対象データ

今回の調査対象データは、以下の 2 つのソフトウェア開発における不具合データである。

- 一般組込み機器製品に搭載される通信ソフトウェア
- 車両搭載用通信プロトコルスタックソフトウェア

これらのデータは PRISMY で管理されており、設計工程に起因する不具合データを PRISMY 情報から抽出して使用する。

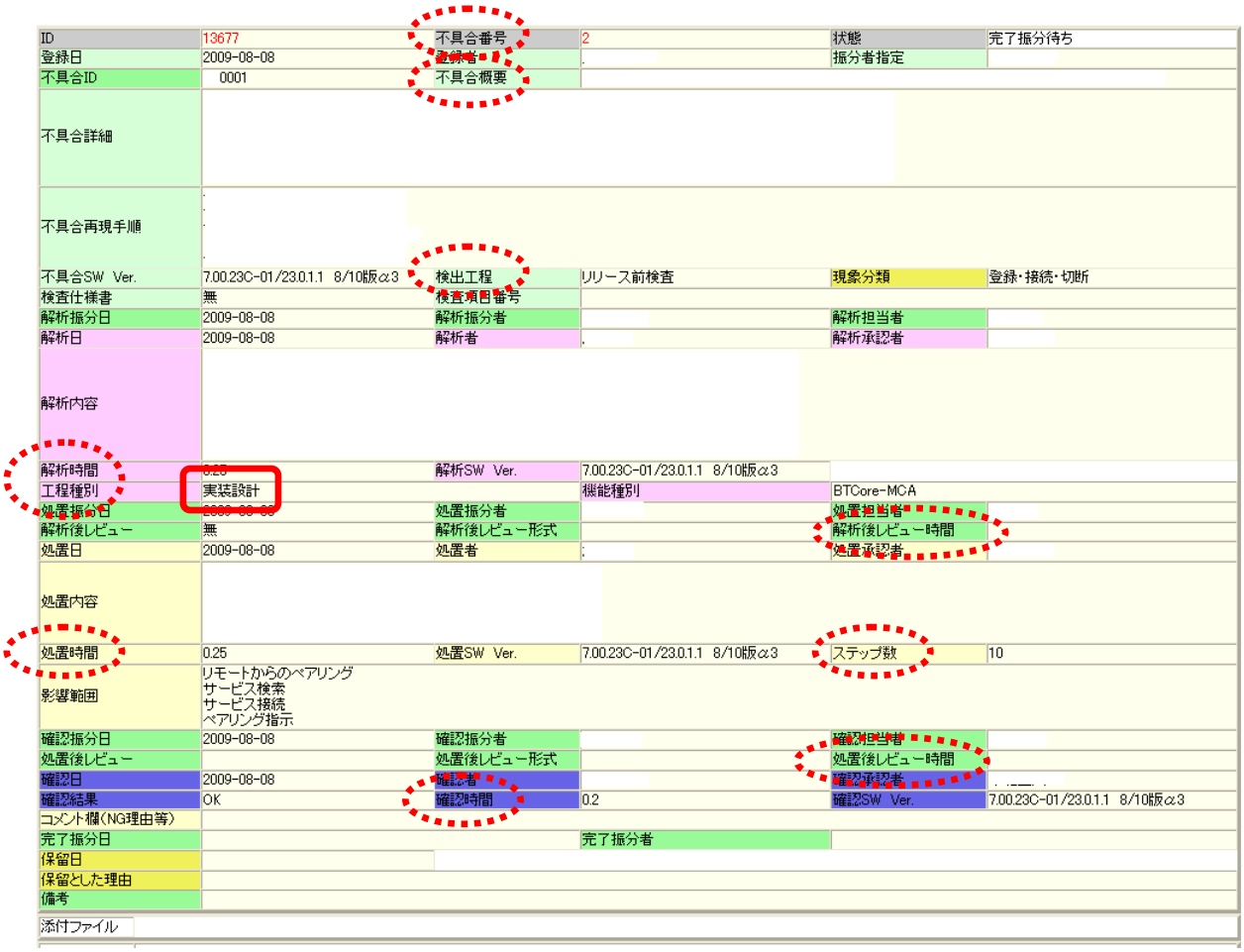
PRISMY 情報からは、不具合管理情報として表 4のようなデータ項目を抽出した。

表中の「解析時間」～「確認時間」までを、不具合対応工数として集計する。

表 4 不具合管理情報

No.	抽出データ項目	説明
1	不具合管理番号	PRISMY 上の管理番号
2	工程種別	不具合の原因工程
3	不具合概要	不具合の概要
4	検出工程	不具合を検出した工程
5	解析時間	解析作業の実績時間
6	解析後レビュー時間	解析内容レビューの実績時間
7	対処時間	対処作業の実績時間
8	対処後レビュー時間	対処内容レビューの実績時間
9	確認時間	対処後の動作確認作業の実績時間
10	ステップ数	対処の際に修正したステップ数

PRISMY では、図 13のような管理画面で不具合情報の管理を行っている。



ID	13677	不具合番号	2	状態	完了振分待ち
登録日	2009-08-08	登録者		振分者指定	
不具合ID	0001	不具合概要			
不具合詳細					
不具合再現手順					
不具合SW Ver.	7.00.23C-01/23.0.1.1 8/10版α3	検出工程	リリース前検査	現象分類	登録・接続・切断
検査仕様書	無	検査項目番号			
解析振分日	2009-08-08	解析振分者		解析担当者	
解析日	2009-08-08	解析者		解析承認者	
解析内容					
解析時間	0.25	解析SW Ver.	7.00.23C-01/23.0.1.1 8/10版α3		
工程種別	実装設計		機能種別	BTCore-MCA	
処置振分日	2009-08-08	処置振分者		処置担当者	
解析後レビュー	無	解析後レビュー形式		解析後レビュー時間	
処置日	2009-08-08	処置者		処置承認者	
処置内容					
処置時間	0.25	処置SW Ver.	7.00.23C-01/23.0.1.1 8/10版α3	ステップ数	10
影響範囲	リモートからのペアリング サービス検索 サービス接続 ペアリング指示				
確認振分日	2009-08-08	確認振分者		確認担当者	
処置後レビュー		処置後レビュー形式		処置後レビュー時間	
確認日	2009-08-08	確認者		確認承認者	
確認結果	OK	確認時間	0.2	確認SW Ver.	7.00.23C-01/23.0.1.1 8/10版α3
コメント欄(NG理由等)					
完了振分日		完了振分者			
保留日					
保留とした理由					
備考					
添付ファイル					

図 13 PRISMY データ管理画面

今回使用した PRISMY 情報では、「工程種別」等に設定されている設計工程の名称が、機能設計／詳細設計／実装設計の 3 つのいずれかであり、一般的な名称であるシステム設計／基本設計／詳細設計と異なっている。したがって、本報告書では、表 5のように設計工程の名称の対応付けを行い、一般的な名称を使用する。

表 5 設計工程の名称の対応付け

PRISMY 情報の名称	一般的な名称
機能設計	システム設計
詳細設計	基本設計
実装設計	詳細設計

５.３調査対象プロジェクト

対象の 2 つのソフトフェア開発は 2005 年から継続しており、今回の対象プロジェクトは 2007～2011 年の間に実行されたプロジェクトである。

調査にあたり、表 6のように該当する 9 つのプロジェクトから、適正な品質レベルを示した 4 つのプロジェクトを選定した。

表 ６ 対象プロジェクト選定

No.	プロジェクト名	品質レベル	選定
1	MT 社向け管理	低い	
2	AL 社向け開発	適正	プロジェクト 1
3	A 社向け開発	高い	
4	HY 社開発	適正	プロジェクト 2
5	M 社向け開発	高い	
6	T 社向け開発	適正	プロジェクト 3
7	A Model 開発	低い	
8	AI 社向け開発 1	低い	
9	AI 社向け開発2	適正	プロジェクト 4

調査の便宜上、選定したプロジェクトは、これ以降プロジェクト 1／プロジェクト 2／プロジェクト 3／プロジェクト 4 と呼ぶこととする。各プロジェクトの概要は、表 7のようになっている。

表 ７ 選定プロジェクトの概要

プロジェクト	開発区分	流用率	開発規模
プロジェクト 1	派生開発	93%	大規模
プロジェクト 2	派生開発	71%	中規模
プロジェクト 3	派生開発	86%	中規模
プロジェクト 4	派生開発	82%	中規模

なお、開発規模については、今回の調査では表 8の定義を使用した。

表 8 開発規模の定義

開発規模	開発工数(人月)	開発Step数(KLOC)
大規模	80以上	300以上
中規模	50 ～ 80未満	100 ～ 300未満
小規模	50未満	100未満

５.３.１ プロジェクト１概要

プロジェクト 1 の開発規模は 348KLOC の大規模プロジェクトである。

新規／変更／流用の比率を図 １4に示す。派生開発で流用率が 94%と非常に高い割合を占めている。

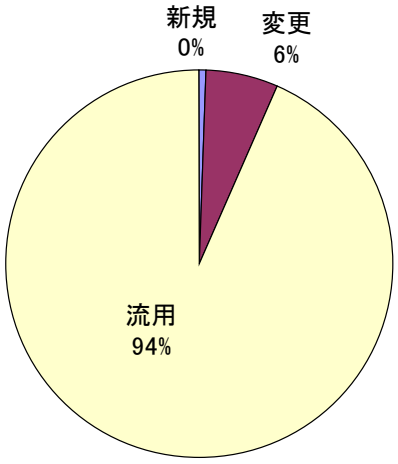


図 １4 プロジェクト１ コード比率

プロジェクト 1 の開発工数は、12,721 時間で、設計／製造／検査／不具合対応の各工程の比率を図 １5に示す。このプロジェクトの結合検査項目は 14,000 件で検査工程が 25%を占めており、不具合対応が 10%を占めている。

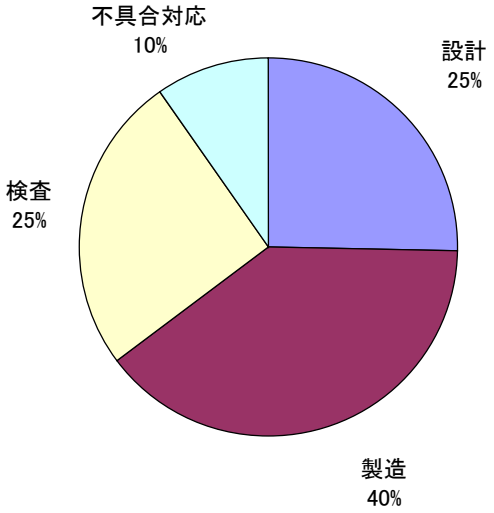


図 １5 プロジェクト１ 工数比率

５.３.２ プロジェクト２概要

プロジェクト 2 の開発規模は 180KLOC の中規模プロジェクトである。

新規／変更／流用の比率を図 16に示す。派生開発で流用率が 71%と高い割合を占めている。

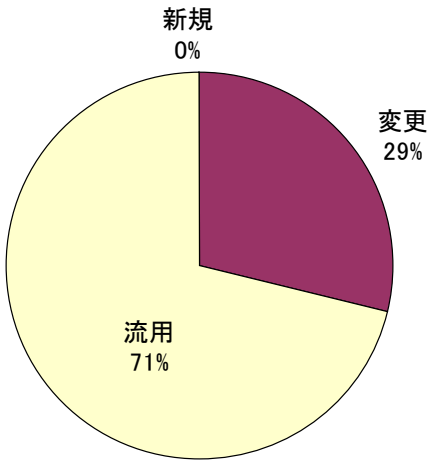


図 16 プロジェクト 2 コード比率

プロジェクト 2 の開発工数は、11,470 時間で、設計／製造／検査／不具合対応の各工程の比率を図 17に示す。このプロジェクトの結合検査項目は 8,700 件で検査工程が 33%を占めており、不具合対応が 10%を占めている。

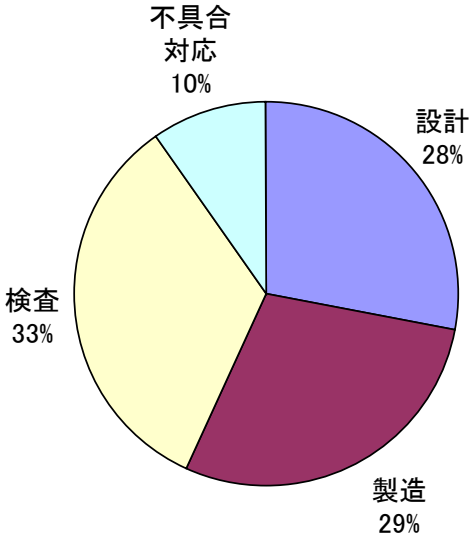


図 17 プロジェクト 2 工数比

５.３.３ プロジェクト３概要

プロジェクト 3 の開発規模は 173.7KLOC の中規模プロジェクトである。

新規／変更／流用の比率を図　１８に示す。派生開発で流用率が 87%と高い割合を占めている。

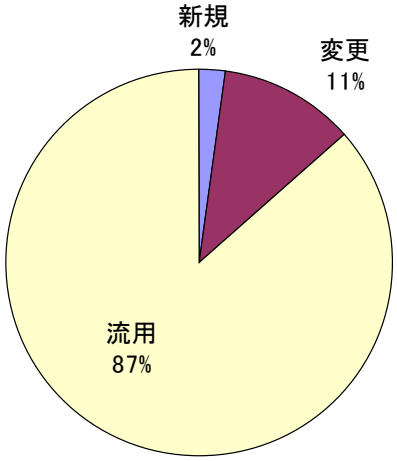


図　１８　プロジェクト３　コード比率

プロジェクト 3 の開発工数は、9,932 時間で、設計／製造／検査／不具合対応の各工程の比率を図　１９に示す。このプロジェクトの結合検査項目は 3,600 件で検査工程が 33%を占めており、不具合対応が 10%を占めている。

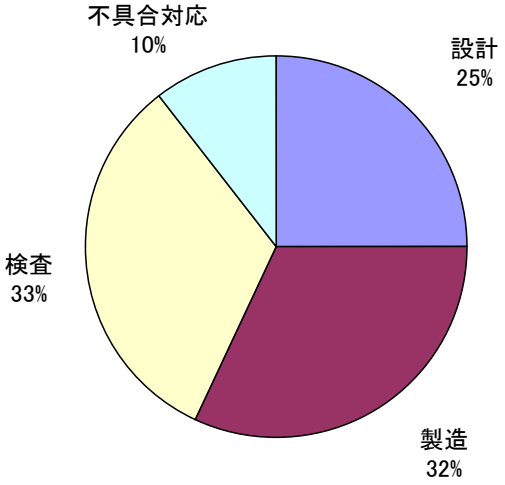


図　１９　プロジェクト３　工数比

５.３.４ プロジェクト４概要

プロジェクト 4 の開発規模は 220KLOC の中規模プロジェクトである。

新規／変更／流用の比率を図 20に示す。派生開発で流用率が 81%と高い割合を占めている。

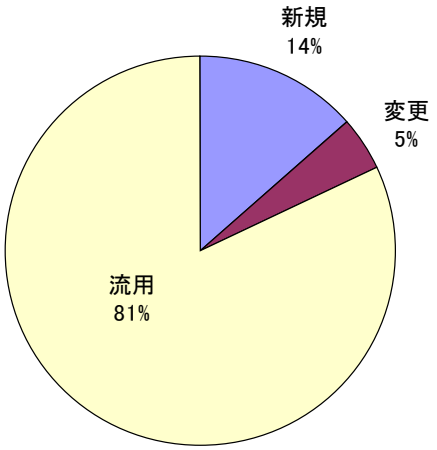


図 20 プロジェクト 4 コード比率

プロジェクト 4 の開発工数は、14,051 時間で、設計／製造／検査／不具合対応の各工程の比率を図 21に示す。このプロジェクトの結合検査項目は 7,500 件で検査工程が 31%を占めており、不具合対応が 4%を占めている。

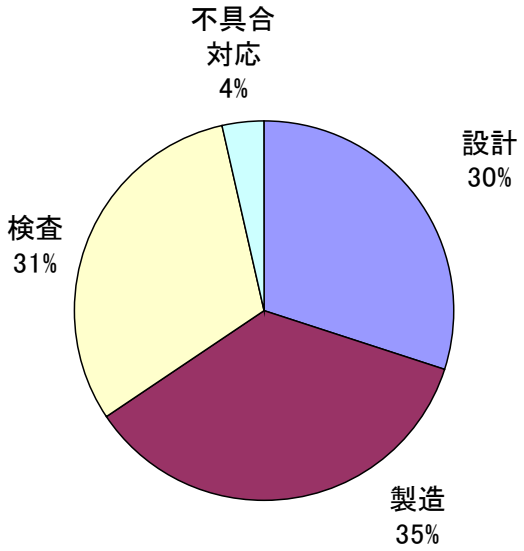


図 21 プロジェクト 4 工数比

６．実験における調査結果

６.１プロジェクト１のデータ分析

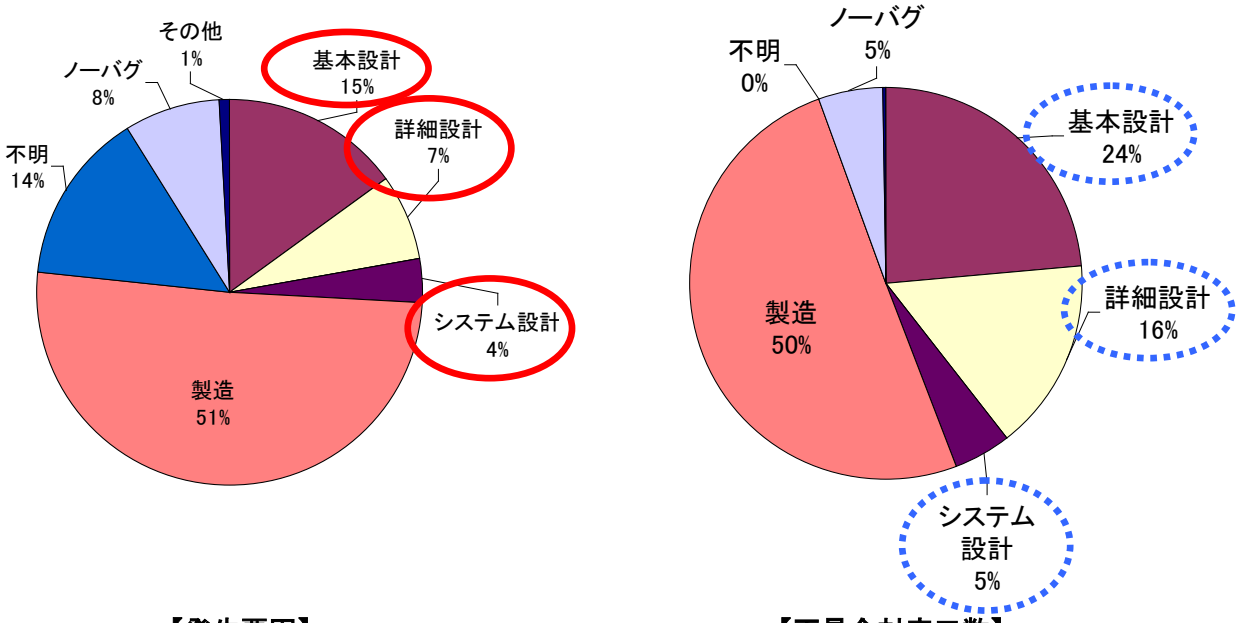
プロジェクト 1 のデータ分析の結果を以下に記す。

６.１.１ プロジェクト１の不具合データ

プロジェクト 1 における不具合件数は 421 件で、不具合対応工数の合計は 1251 時間であった。発生要因で分類した場合の件数の割合、及び不具合対応工数の割合を図 22に示す。

発生要因工程としては「製造」の割合が 51%と最も多い。しかし、今回の調査では設計工程を要因とした不具合を対象としているため、ここでは「システム設計」、「基本設計」、「詳細設計」の 3 つの工程に注目する。これら 3 工程を要因とする不具合は、合わせて全体の 26%を占めている。

不具合対応工数で見ると、「システム設計」、「基本設計」、「詳細設計」の割合が、合わせて 45%となり、件数の割合に比較して大きくなっていることから、設計工程が要因の不具合は不具合対応工数が大きくなることを示している。



【発生要因】　　【不具合対応工数】

図 22 プロジェクト1 不具合発生要因及び不具合対応工数

また、これらの不具合が検出された工程を分析した結果を図 23に示す。

上流工程で検出される割合が極端に少なく、検査工程で検出される割合が、合計で 51%と半数を占めている。また、他社や他プロジェクトなど外部からの指摘も、合計で43%と大きな割合を占めている。

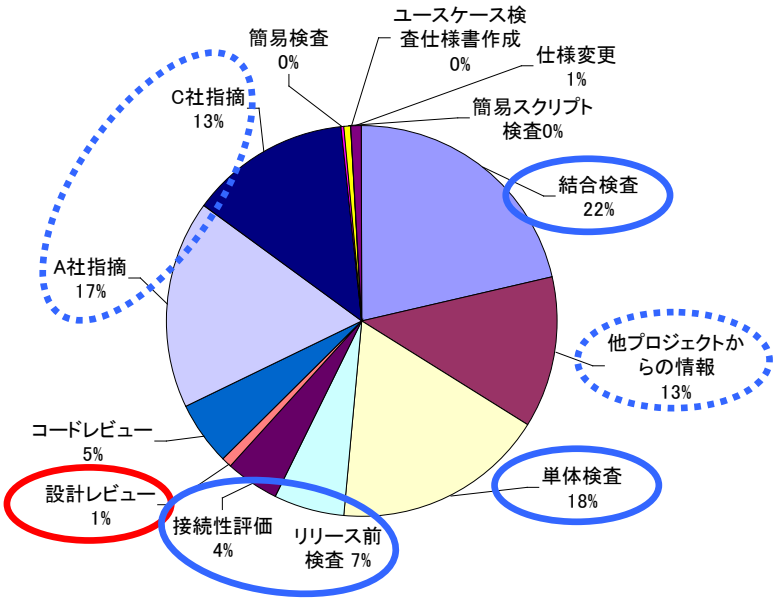


図 23 プロジェクト1 不具合検出工程

これらの不具合のうち、設計要因の不具合件数は 107 件で、その不具合対応工数の合計は552.35 時間であった。プロジェクト 1 における、設計要因の不具合対応工数の内訳を表 9に示す。

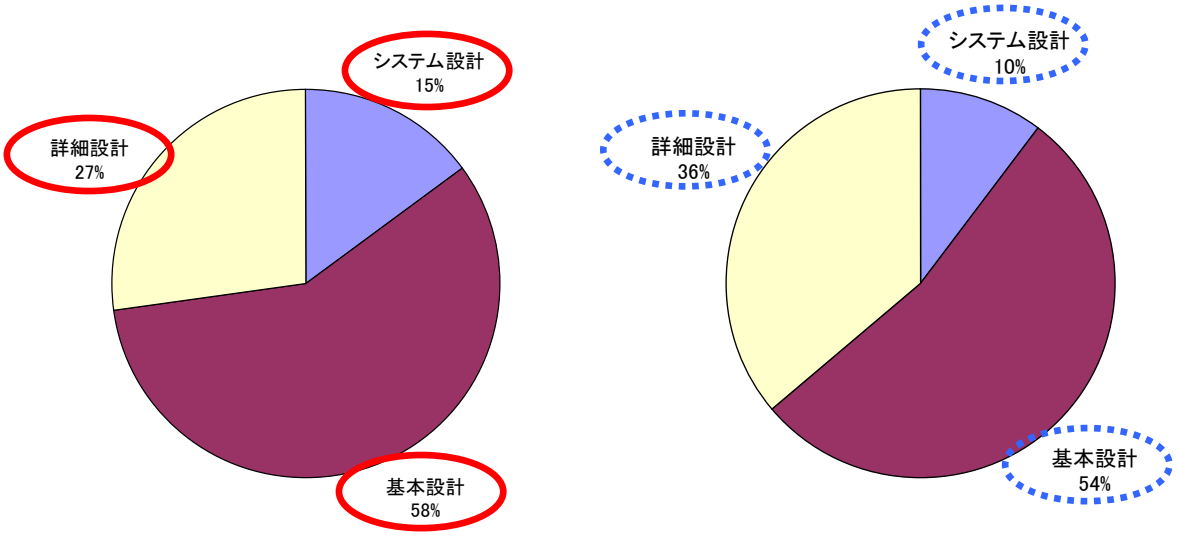
表 9 プロジェクト1 不具合対応工数の内訳

工程種別	不具合対応工数(H)					
	解析時間	解析後レビュー	処置時間	処置後レビュー	確認時間	対応工数合計
システム設計	20.00	0.00	37.00	0.00	0.25	57.25
基本設計	148.75	0.25	143.85	0.75	2.50	296.10
詳細設計	113.50	0.00	85.00	0.00	0.50	199.00
小計	282.25	0.25	265.85	0.75	3.25	552.35

不具合件数を「システム設計」、「基本設計」、「詳細設計」、の 3 工程で分類した結果、及びそれぞれの不具合対応工数の割合を図 24に示す。

不具合件数では、基本設計が 58%、詳細設計が 27%となり、この 2 つの要因が、合わせて 8 割強を占めている。

また、不具合対応工数については、詳細設計の不具合対応工数の割合が不具合件数に比べて約 10%増加しており、詳細設計の不具合対応工数が大きくなる傾向にある。



【不具合件数】　　【不具合対応工数】

図 24 プロジェクト1 設計要因の不具合件数及び設計要因の不具合対応工数

次に、不具合ごとの不具合対応工数の分布を図 25に示す。

不具合対応工数が 5 時間未満のものが 71%と大半を占めている。工数の削減にはこの部分への対処が不可欠である。

また、不具合対応工数が 30 時間以上のものが 2 件あり、1 件はデグレードであり、もう 1 件は設計誤りが原因であった。この種の不具合は、発生すると対応により多くの工数を要すると考えられる。

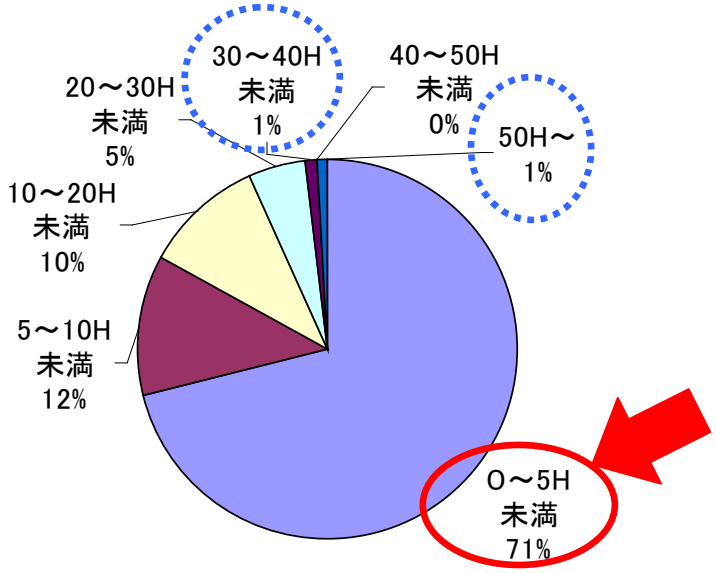


図 25 プロジェクト1 不具合ごとの不具合対応工数

６.１.２ プロジェクト１ 不具合原因の内訳と対策工数

抽出した設計要因の不具合の原因を分析した結果を表 10に示す。なお、一つの不具合が複数の原因に区分されることもあるため、表中の不具合件数の合計は実際の件数よりも多くなっている。

表 10 プロジェクト1 不具合の原因区分

工程種別	原因区分(件)							
	設計漏れ	設計誤り	検査漏れ	検査仕様誤り	レビュー不足	トレーサビリティ	外部要因	不明
システム設計	8	0	1	3	5	4	1	0
基本設計	26	32	1	1	12	5	0	1
詳細設計	0	0	0	0	29	0	0	0
小計	34	32	2	4	46	9	1	1

また、それぞれの割合を図 26に示す。

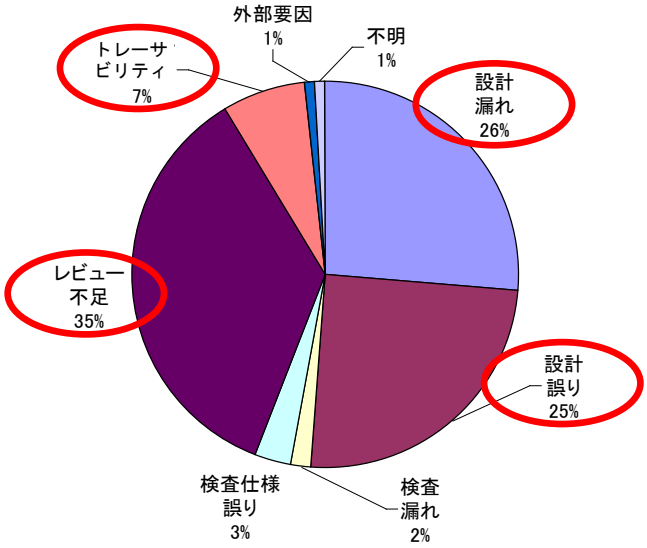


図 26 プロジェクト1 不具合の原因区分比

「設計漏れ」、「設計誤り」、「レビュー不足」、「トレーサビリティ(欠如)」の 4 項目が合計で 93%と大半を占めている。

トレーサビリティを確保することで、上位及び下位の要件との関連付けが容易になり、記述の漏れや不整合を防止し、レビューの質の向上が期待できるので、いずれの原因も、トレーサビリティを確保することが有効な対策になる。

それぞれの不具合に対して、トレーサビリティ対策工数を見積もった。

表 1の見積り単位で、不具合ごとのトレーサビリティ対策工数を見積もった結果を表 11に示す。

プロジェクト 1 における、トレーサビリティ対策工数の合計は 258 時間となった。

表 11 プロジェクト1 トレーサビリティ対策工数

工程種別	トレーサビリティ対策工数(H)				
	設計書作成	コード作成	設計書レビュー	コードレビュー	対策工数合計
システム設計	9.50	9.50	6.50	5.00	30.50
基本設計	52.50	53.50	36.00	28.00	170.00
詳細設計	16.50	15.00	14.00	12.00	57.50
合計	78.50	78.00	56.50	45.00	258.00

このトレーサビリティ対策工数と設計要因による不具合対応工数の合計(552.35 時間)との比較を図 27に示す。

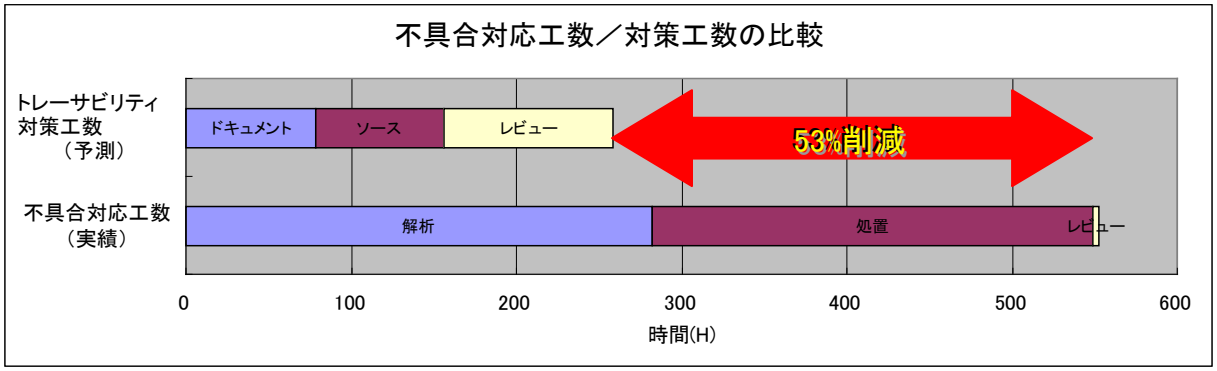


図 27 プロジェクト1 不具合対応工数とトレーサビリティ対策工数比

この結果から、プロジェクト 1 では 53%の工数削減が期待できることになる。

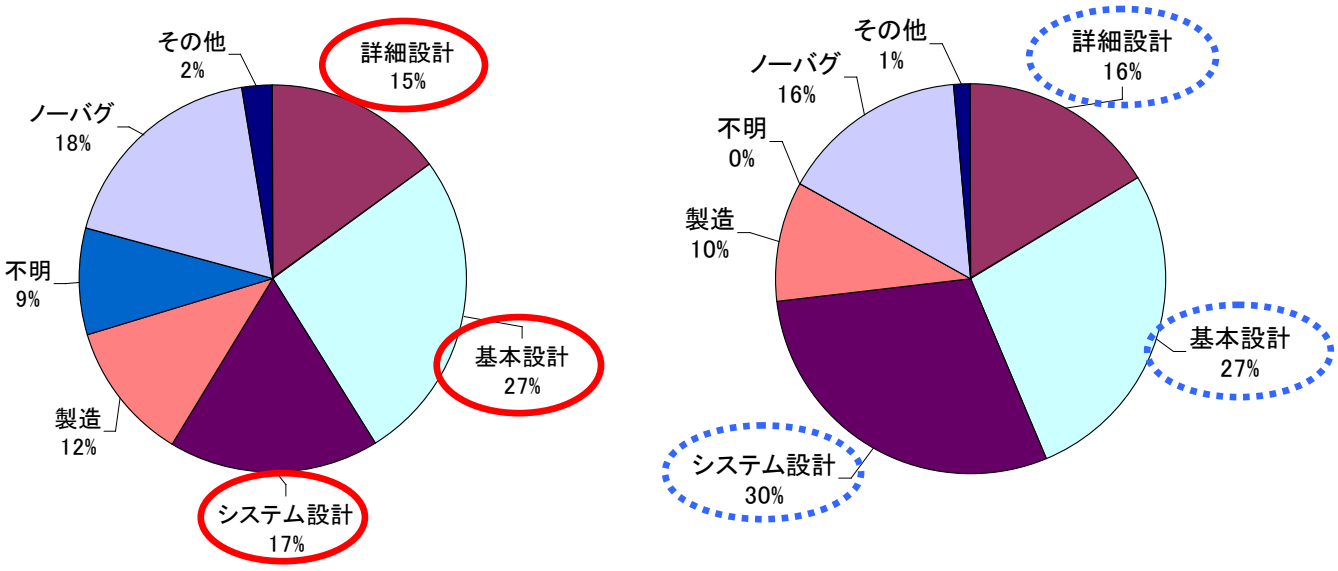
６.２プロジェクト２のデータ分析

６.２.１ プロジェクト２の不具合データ

プロジェクト 2 における不具合件数は 294 件で、不具合対応工数の合計は 993 時間であった。発生要因で分類した場合の件数の割合、及び不具合対応工数の割合を図 28に示す。

発生要因工程としては、今回の調査で対象とした設計工程を要因としている不具合の割合が多い。「システム設計」、「基本設計」、「詳細設計」の 3 つの工程を要因とする不具合は、合わせて全体の 59%を占めている。

不具合対応工数で見ると、前述の 3 工程の割合が、合計で 73%となり、件数の割合に比較して大きくなっていることから、設計工程が要因の不具合は不具合対応工数が大きくなることを示している。



【発生要因】　　【不具合対応工数】

図 28 プロジェクト 2 不具合発生要因及び不具合対応工数

また、これらの不具合が検出された工程を分析した結果を図 29に示す。

上流工程で検出される割合が極端に少なく、検査工程で検出される割合が、合計で 59%と半数以上を占めている。また、他社や他プロジェクトなど外部からの指摘も、合計で33%と大きな割合を占めている。

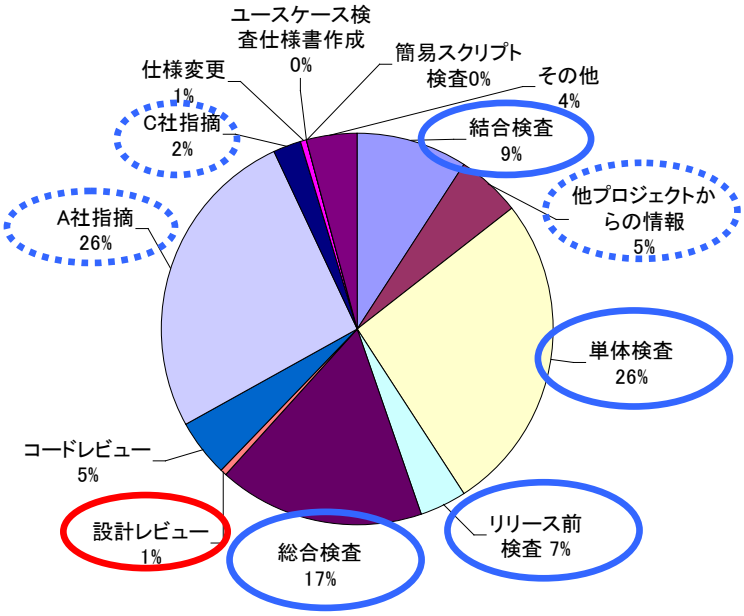


図 29 プロジェクト 2 不具合検出工程

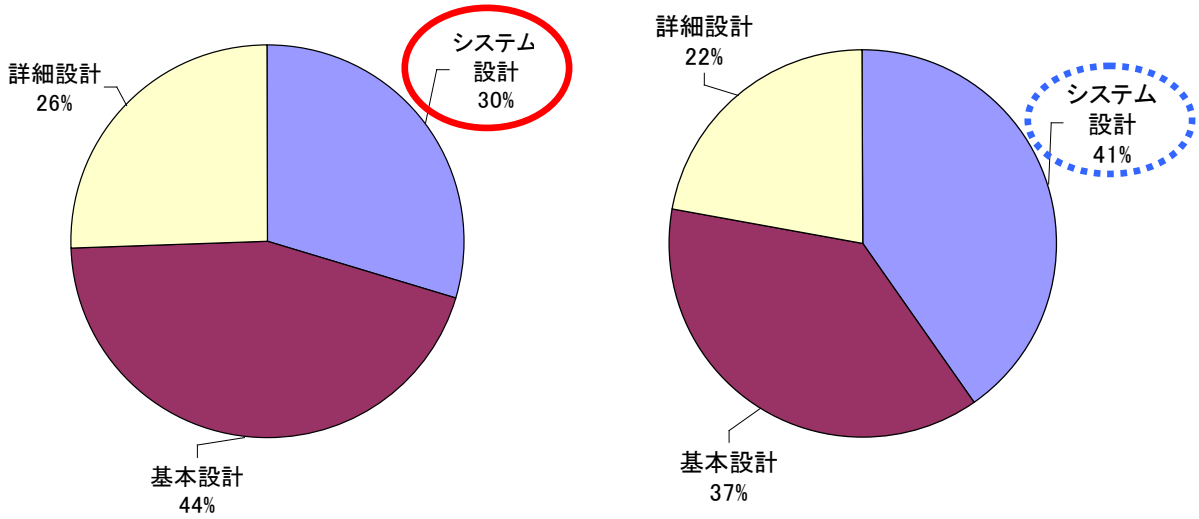
これらの不具合のうち、設計要因の不具合件数は 172 件で、その不具合対応工数の合計は 726.25 時間であった。プロジェクト 2 における、設計要因の不具合対応工数の内訳を表 12に示す。

表 12 プロジェクト 2 不具合対応工数の内訳

工程種別	不具合対応工数(H)					
	解析時間	解析後レビュー	処置時間	処置後レビュー	確認時間	対応工数合計
システム設計	127.50	0.50	164.75	0.00	0.00	292.75
基本設計	119.00	1.50	150.75	0.00	0.00	271.25
詳細設計	64.50	0.00	97.75	0.00	0.00	162.25
小計	311.00	2.00	413.25	0.00	0.00	726.25

不具合件数を「システム設計」、「基本設計」、「詳細設計」の 3 工程で分類した結果、及びそれぞれの不具合対応工数の割合を図 30に示す。

システム設計については、不具合件数では 30%であるが、不具合対応工数では 41%と 11%増加している。逆に基本設計、詳細設計については、不具合対応工数での割合が減少している。



【不具合件数】　　【不具合対応工数】

図 30 プロジェクト 2 設計要因の不具合件数及び設計要因の不具合対応工数

次に、不具合ごとの不具合対応工数の分布を図 31に示す。

不具合対応工数が 5 時間未満のものが 68%と大半を占めている。工数の削減にはこの部分への対処が不可欠である。

また、不具合対応工数が 20 時間以上のものが 3 件あり、いずれも設計誤りが原因であった。この種の不具合は、発生すると対応により多くの工数を要すると考えられる。

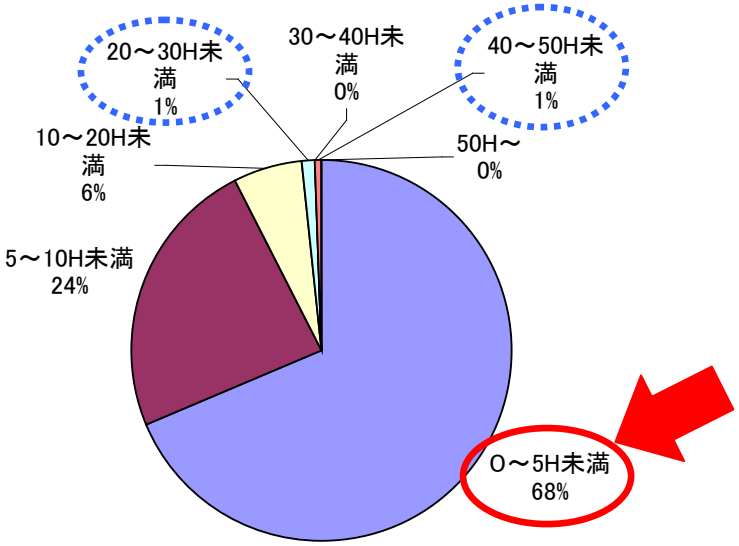


図 31 プロジェクト 2 不具合ごとの不具合対応工数

６.２.２ 不具合原因の内訳と対策工数

抽出した設計要因の不具合の原因を分析した結果を表 13に示す。なお、一つの不具合が複数の原因に区分されることもあるため、表中の不具合件数の合計は実際の件数よりも多くなっている。

表 13 プロジェクト２ 不具合の原因区分

工程種別	原因区分(件)							
	設計漏れ	設計誤り	検査漏れ	検査仕様誤り	レビュー不足	トレーサビリティ	外部要因	不明
システム設計	5	34	－	1	7	－	7	－
基本設計	7	60	－	－	5	2	2	5
詳細設計	8	26	－	1	12	－	－	1
小計	20	120	0	2	24	2	9	6

また、それぞれの割合を図 32に示す。

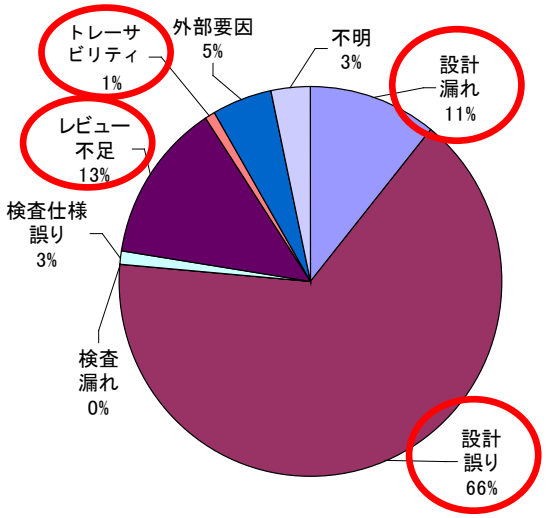


図 32 プロジェクト２ 不具合の原因区分比

「設計漏れ」、「設計誤り」、「レビュー不足」、「トレーサビリティ(欠如)」の 4 項目が合計で 91%と大半を占めている。

トレーサビリティを確保することで、上位及び下位の要件との関連付けが容易になり、記述の漏れや不整合を防止し、レビューの質の向上が期待できるので、いずれの原因も、トレーサビリティを確保することが有効な対策になる。

それぞれの不具合に対してトレーサビリティ対策工数を見積もった。

表 1の見積り単位で、不具合ごとのトレーサビリティ対策工数を見積もった結果を表 14に示す。

プロジェクト 2 における、トレーサビリティ対策工数の合計は 563.25 時間となった。

表 14 プロジェクト 2 トレーサビリティ対策工数

工程種別	トレーサビリティ対策工数(H)				
	設計書作成	コード作成	設計書レビュー	コードレビュー	対策工数合計
システム設計	77.00	87.00	44.00	32.00	240.00
基本設計	61.50	60.25	44.00	37.00	202.75
詳細設計	36.50	35.50	26.00	22.50	120.50
合計	175.00	182.75	114.00	91.50	563.25

このトレーサビリティ対策工数と設計要因による不具合対応工数の合計(726.25 時間)との比較を図 33に示す。

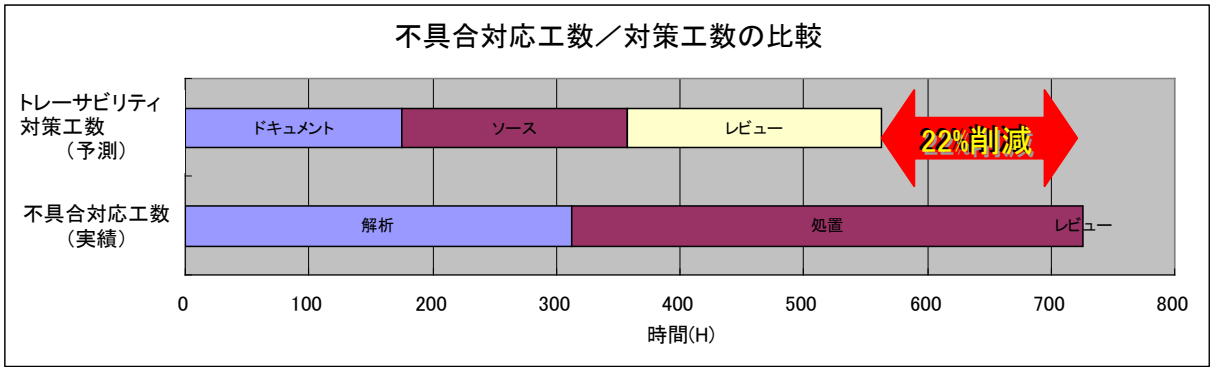


図 33 プロジェクト 2 不具合対応工数とトレーサビリティ対策工数比

この結果から、プロジェクト 2 では 22%の工数削減が期待できることになる。

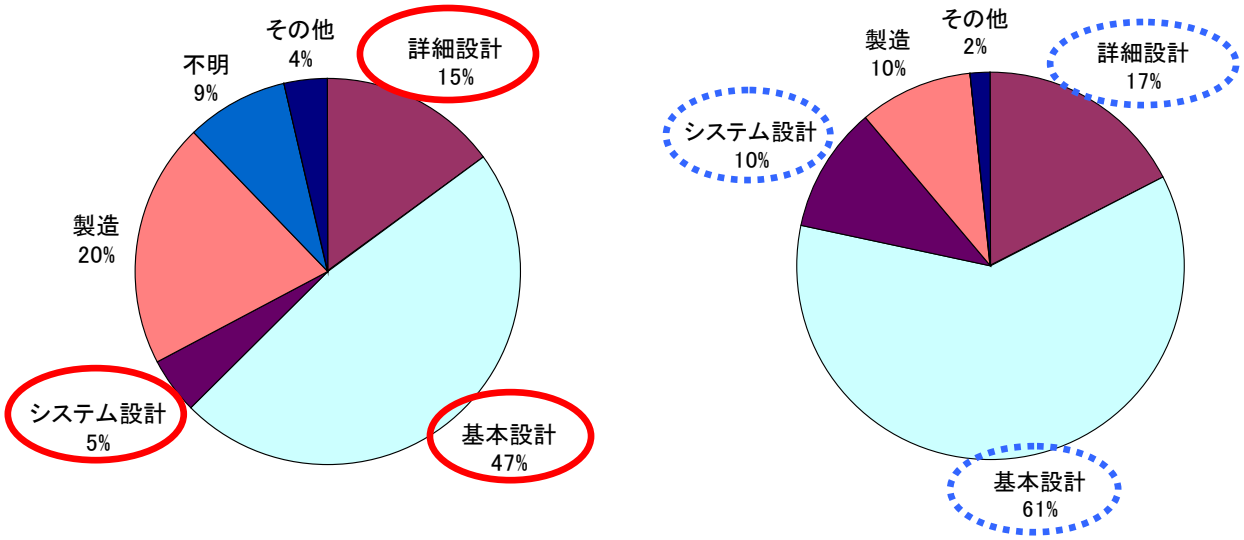
６.３プロジェクト３のデータ分析

６.３.１ プロジェクト３の不具合データ

プロジェクト 3 における不具合件数は 395 件で、不具合対応工数の合計は 1039 時間であった。発生要因で分類した場合の件数の割合、及び不具合対応工数の割合を図 34に示す。

発生要因工程としては、今回の調査で対象とした設計工程を要因としている不具合の割合が多い。「システム設計」、「基本設計」、「詳細設計」の 3 つの工程を要因とする不具合は、合わせて全体の 67%を占めている。

不具合対応工数で見ると、前述の 3 工程の割合が、合計で 88%となり、件数の割合に比較して大きくなっていることから、設計工程が要因の不具合は不具合対応工数が大きくなることを示している。



【発生要因】　　【不具合対応工数】

図 34 プロジェクト 3 不具合発生要因及び不具合対応工数

また、これらの不具合が検出された工程を分析した結果を図 35に示す。

上流工程で検出される割合が 6%と少なく、検査工程で検出される割合が、合計で 52%と半数以上を占めている。また、他社や他プロジェクトなど外部からの指摘も、合計で33%と大きな割合を占めている。

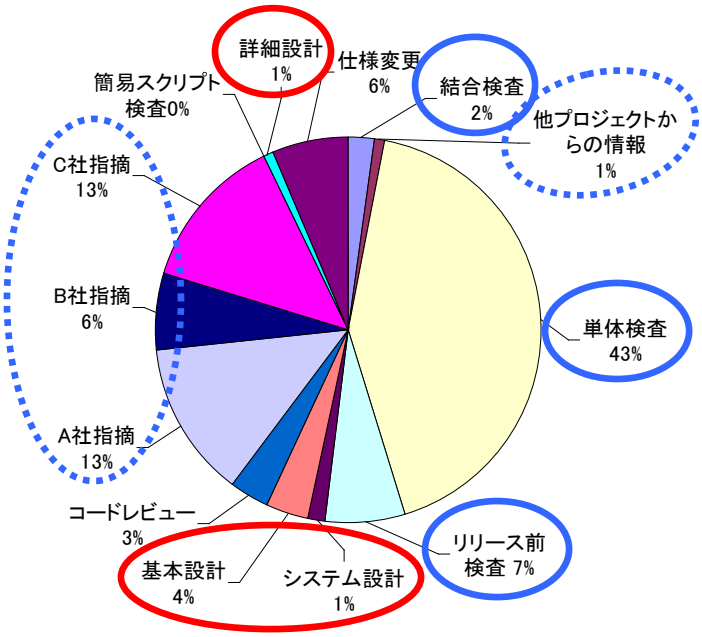


図 35 プロジェクト 3 不具合検出工程

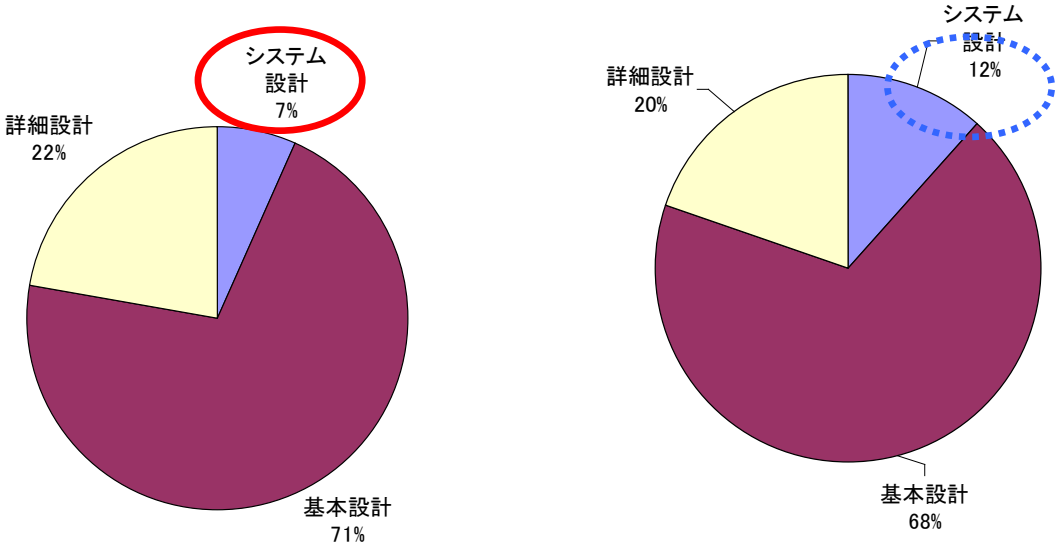
これらの不具合のうち、設計要因の不具合件数は 265 件で、その不具合対応工数の合計は 922.45 時間であった。プロジェクト 3 における、設計要因の不具合対応工数の内訳を表 15に示す。

表 15 プロジェクト 3 不具合対応工数の内訳

工程種別	不具合対応工数(H)					
	解析時間	解析後レビュー	処置時間	処置後レビュー	確認時間	対応工数合計
システム設計	21.10	0.00	85.50	0.00	1.10	107.70
基本設計	279.00	0.00	334.20	0.00	19.80	633.00
詳細設計	80.60	0.00	91.10	0.00	10.05	181.75
小計	380.70	0.00	510.80	0.00	30.95	922.45

不具合件数を「システム設計」、「基本設計」、「詳細設計」の3工程で分類した結果、及びそれぞれの不具合対応工数の割合を図 36に示す。

システム設計については、不具合件数では 7%であるが、不具合対応工数では 12%となり 5%増加している。基本設計及び詳細設計についてのそれぞれの割合を比較すると、それぞれ 3%、2%と僅かではあるが減少している。



【不具合件数】　　【不具合対応工数】

図 36 プロジェクト 3 設計要因の不具合件数及び設計要因の不具合対応工数

次に、不具合ごとの不具合対応工数の分布を図 37に示す。

不具合対応工数が 5 時間未満のものが約 79%と大半を占めている。工数の削減にはこの部分への対処が不可欠である。

また、不具合対応工数が 40 時間以上のものが 2 件あり、いずれも設計誤りが原因であった。この種の不具合は、発生すると対応により多くの工数を要すると考えられる。

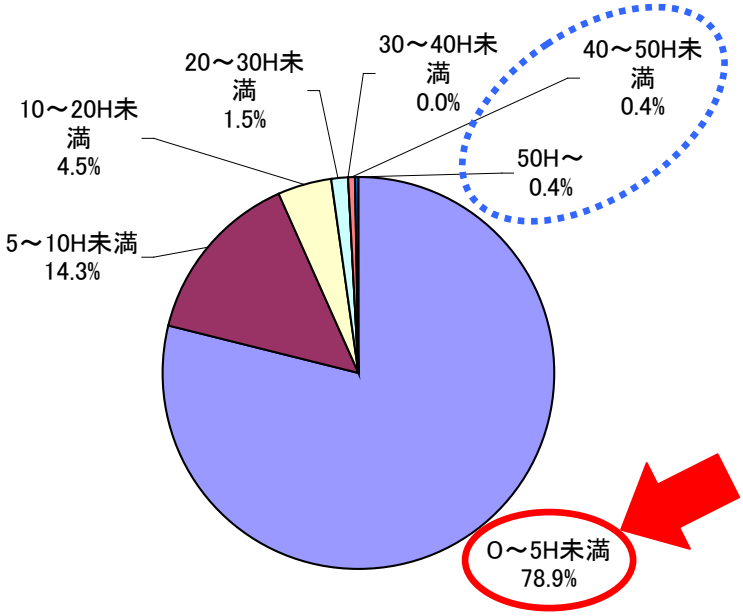


図 37 プロジェクト 3 不具合ごとの不具合対応工数

６.３.２ 不具合原因の内訳と対策工数

抽出した設計要因の不具合の原因を分析した結果を表 16に示す。なお、一つの不具合が複数の原因に区分されることもあるため、表中の不具合件数の合計は実際の件数よりも多くなっている。

表 16 プロジェクト３ 不具合の原因区分

工程種別	原因区分(件)							
	設計漏れ	設計誤り	検査漏れ	検査仕様誤り	レビュー不足	トレーサビリティ	外部要因	不明
システム設計	1	11	-	-	1	0	-	6
基本設計	16	123	-	4	21	3	19	18
詳細設計	18	27	-	-	15	2	1	6
小計	35	161	0	4	37	5	20	30

また、それぞれの割合を図 38に示す。

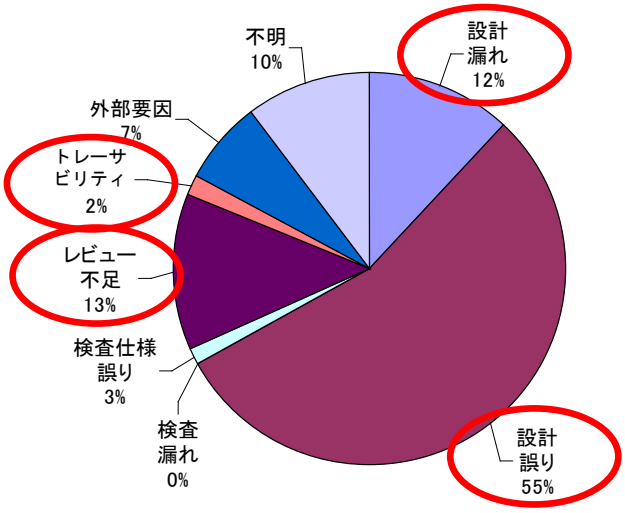


図 38 プロジェクト３ 不具合の原因区分比

「設計漏れ」、「設計誤り」、「レビュー不足」、「トレーサビリティ(欠如)」の 4 項目が合計で 82%と大半を占めている。

トレーサビリティを確保することで、上位及び下位の要件との関連付けが容易になり、記述の漏れや不整合を防止し、レビューの質の向上が期待できるので、いずれの原因も、トレーサビリティを確保することが有効な対策になる。

それぞれの不具合に対してトレーサビリティ対策工数を見積もった。

表 1の見積り単位で、不具合ごとのトレーサビリティ対策工数を見積もった結果を表 17に示す。

プロジェクト 3 における、トレーサビリティ対策工数の合計は 516.5 時間となった。

表 17 プロジェクト 3 トレーサビリティ対策工数

工程種別	トレーサビリティ対策工数(H)				
	設計書作成	コード作成	設計書レビュー	コードレビュー	対策工数合計
システム設計	9.50	11.00	6.50	5.50	32.50
基本設計	102.00	96.25	79.00	71.00	348.25
詳細設計	40.00	37.75	31.00	27.00	135.75
合計	151.50	145.00	116.50	103.50	516.50

このトレーサビリティ対策工数と設計要因による不具合対応工数の合計(922.45 時間)との比較を図 39に示す。

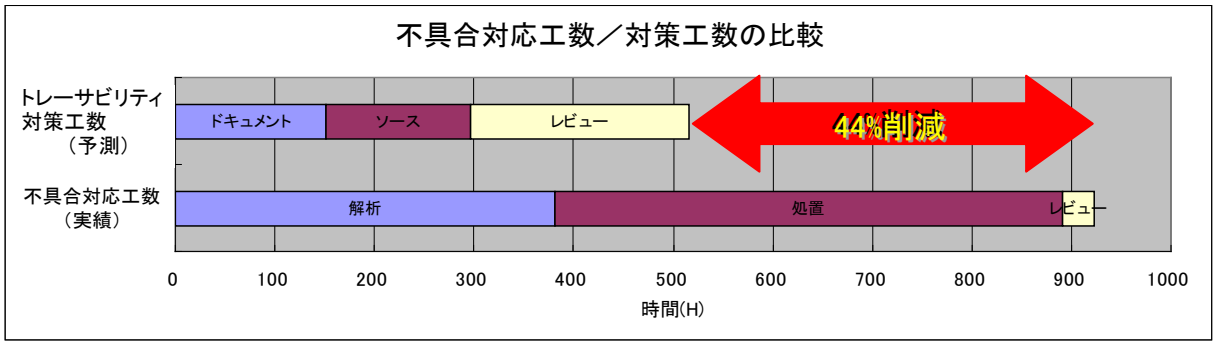


図 39 プロジェクト 3 不具合対応工数とトレーサビリティ対策工数比

この結果から、プロジェクト 3 では 44%の工数削減が期待できることになる。

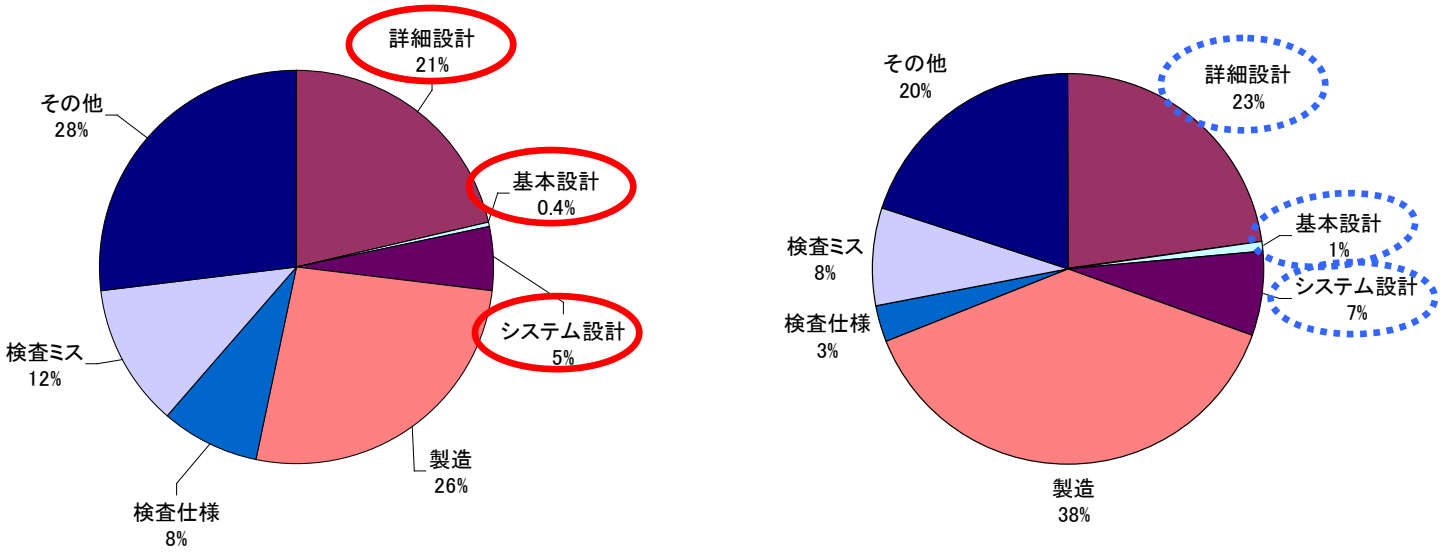
６.４プロジェクト４のデータ分析

６.４.１ プロジェクト４の不具合データ

プロジェクト 4 における不具合件数は 272 件で、不具合対応工数の合計は 497 時間であった。発生要因で分類した場合の件数の割合、及び不具合対応工数の割合を図 40に示す。

発生要因工程としては、今回の調査で対象とした設計工程を要因としている不具合の割合が多い。「システム設計」、「基本設計」、「詳細設計」の 3 つの工程を要因する不具合は、合わせて全体の 26.4%を占めている。

不具合対応工数で見ると、前述の 3 工程の割合が、合計で 31%となり、件数の割合に比較して大きくなっていることから、設計工程が要因の不具合は不具合対応工数が大きくなることを示している。



【発生要因】　　【不具合対応工数】

図 40 プロジェクト 4 不具合発生要因及び不具合対応工数

また、これらの不具合が検出された工程を分析した結果を図 41に示す。

プロジェクト 4 については、上流工程で検出されている不具合はなく、検査工程で検出される割合が、合計で 99%とほぼ全件を占めている。また、他社や他プロジェクトなど外部からの指摘も 1%とごくわずかになっている。

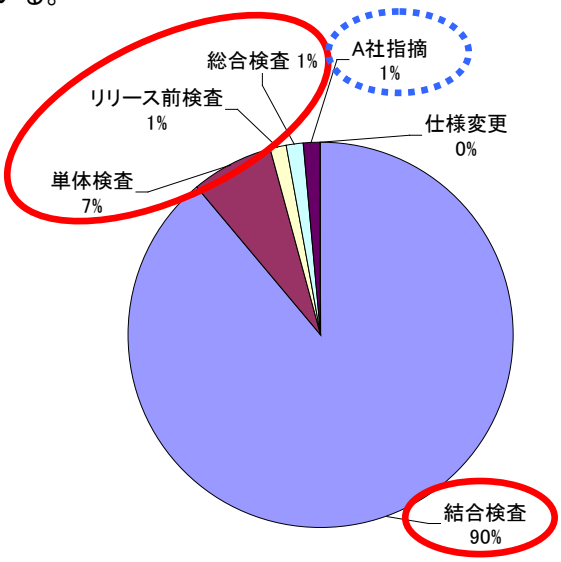


図 41 プロジェクト 4 不具合検出工程

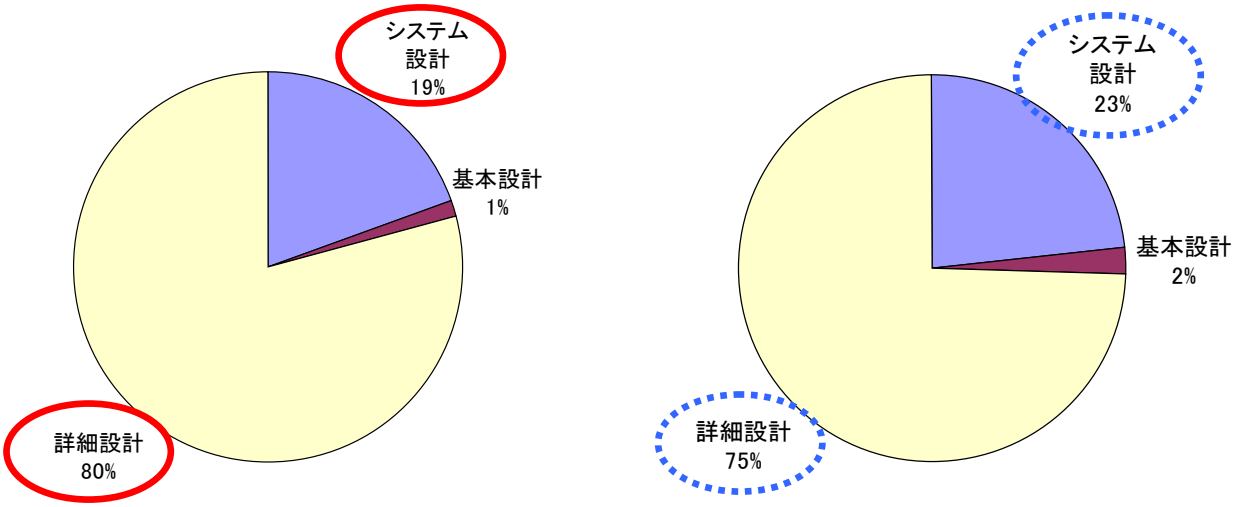
これらの不具合のうち、設計要因の不具合件数は 72 件で、その不具合対応工数の合計は 152.35 時間であった。プロジェクト 4 における、設計要因の不具合対応工数の内訳を表 18に示す。

表 18 プロジェクト 4 不具合対応工数の内訳

工程種別	不具合対応工数(H)					
	解析時間	解析後レビュー	処置時間	処置後レビュー	確認時間	対応工数合計
システム設計	19.60	0.00	13.00	0.00	3.00	35.60
基本設計	0.50	0.00	3.00	0.00	0.00	3.50
詳細設計	70.25	0.00	28.75	0.00	14.25	113.25
小計	90.35	0.00	44.75	0.00	17.25	152.35

不具合件数を「システム設計」、「基本設計」、「詳細設計」の 3 工程で分類した結果、及びそれぞれの不具合対応工数の割合を図 42に示す。

システム設計については、不具合件数では 19%であるが、不具合対応工数では 23%となり 4%増加している。逆に詳細設計の割合は、80%から 75%と 5%減少している。



【不具合件数】　　【不具合対応工数】

図 42 プロジェクト 4 設計要因の不具合件数及び設計要因の不具合対応工数

次に、不具合ごとの不具合対応工数の分布を図 43に示す。

不具合対応工数が 5 時間未満のものが 91%と大半を占めている。工数の削減にはこの部分への対処が不可欠である。

なお、このプロジェクトでは、不具合対応工数が 20 時間以上の不具合は存在しなかった。

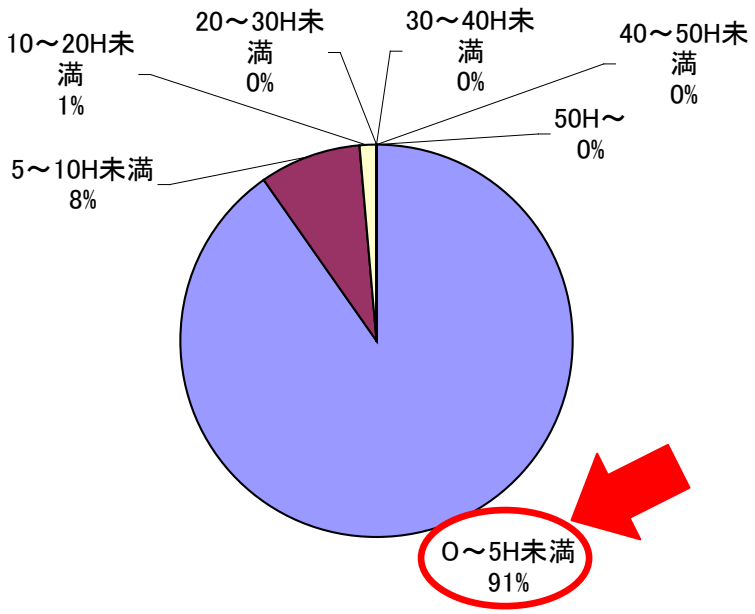


図 43 プロジェクト 4 不具合ごとの不具合対応工数

６.４.２ 不具合原因の内訳と対策工数

抽出した設計要因の不具合の原因を分析した結果を表 19に示す。

表 19 プロジェクト４ 不具合の原因区分

工程種別	原因区分(件)							
	設計漏れ	設計誤り	検査漏れ	検査仕様誤り	レビュー不足	トレーサビリティ	外部要因	不明
システム設計	1	5	-	-	1	-	2	5
基本設計	-	1	-	-	-	-	-	-
詳細設計	7	29	0	8	4	0	2	7
小計	8	35	0	8	5	0	4	12

また、それぞれの割合を図 44に示す。

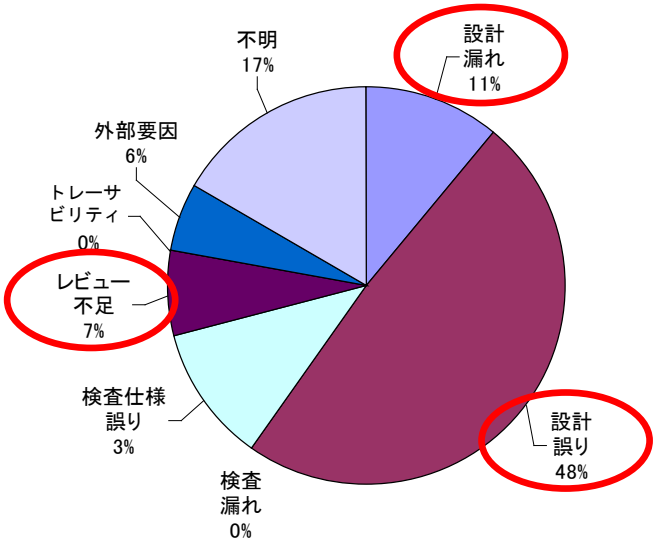


図 44 プロジェクト４ 不具合の原因区分比

「設計漏れ」、「設計誤り」、「レビュー不足」の 3 項目が合計で 66%と大半を占めている。

トレーサビリティを確保することで、上位及び下位の要件との関連付けが容易になり、記述の漏れや不整合を防止し、レビューの質の向上が期待できるので、いずれの原因も、トレーサビリティを確保することが有効な対策になる。

それぞれの不具合に対してトレーサビリティ対策工数を見積もった。

表 1の見積り単位で、不具合ごとのトレーサビリティ対策工数を見積もった結果を表 20に示す。

プロジェクト 4 における、トレーサビリティ対策工数の合計は 137.75 時間となった。

表 20 プロジェクト 4 トレーサビリティ対策工数

工程種別	トレーサビリティ対策工数(H)				
	設計書作成	コード作成	設計書レビュー	コードレビュー	対策工数合計
システム設計	6.00	6.50	3.50	3.00	19.00
基本設計	0.50	0.50	0.50	0.50	2.00
詳細設計	36.00	36.75	24.50	19.50	116.75
合計	42.50	43.75	28.50	23.00	137.75

このトレーサビリティ対策工数と設計要因による不具合対応工数の合計(152.35 時間)との比較を図 45に示す。

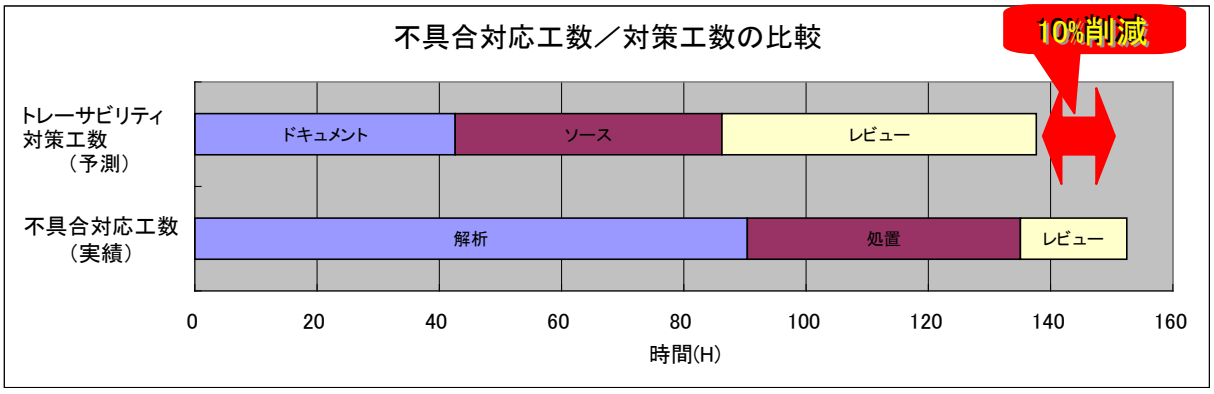


図 45 プロジェクト 4 不具合対応工数とトレーサビリティ対策工数比

この結果から、プロジェクト 4 では 10%の工数削減が期待できることになる。

７．調査結果における考察と評価

「６．実験における調査結果」の結果から、トレーサビリティ確保の範囲と工数削減の効果について考察する。

７.１トレーサビリティと工数削減

表 21は、今回調査した不具合対応工数と、その不具合を防止するためのトレーサビリティ対策工数をまとめたものである。

この表の「不具合対応工数」は、PRISMY 情報から抽出した、設計要因の不具合情報 1 件ごとの「解析時間」～「確認時間」を集計したものである。「解析時間」、「確認時間」等の意味については、表 4を参照のこと。「トレーサビリティ対策工数」は、その不具合情報を基に見積もったトレーサビリティを確保するため工数を表しており、「工数差」は、不具合対応工数とトレーサビリティ対策工数との差を表している。人月は、この工数差を 160 時間／人月で換算した値である。

また、「人月累計」は、「４．適用レベルとトレーサビリティの範囲について」で定義した、適用レベル L2／L3／L4 に従ってトレーサビリティの範囲が拡がることに伴う、工数の違いを表した値である。

表 21 トレーサビリティの工数削減効果

プロジェクト	項目	L2	L3	L4	合計
プロジェクト1	不具合対応工数(H)	57.25	296.10	199.00	552.35
	トレーサビリティ対策工数(H)	30.50	170.00	57.50	258.00
	工数差(H)	26.75	126.10	141.50	294.35
	工数(人月)	0.17	0.79	0.88	1.84
	人月累計(人月)	0.17	0.96	1.84	―
プロジェクト2	不具合対応工数(H)	292.75	271.25	162.25	726.25
	トレーサビリティ対策工数(H)	240.00	202.75	120.50	563.25
	工数差(H)	52.75	68.50	41.75	163.00
	工数(人月)	0.33	0.43	0.26	1.02
	人月累計(人月)	0.33	0.76	1.02	―
プロジェクト3	不具合対応工数(H)	107.70	633.00	181.75	922.45
	トレーサビリティ対策工数(H)	32.50	348.25	135.75	516.50
	工数差(H)	75.20	284.75	46.00	405.95
	工数(人月)	0.47	1.78	0.29	2.54
	人月累計(人月)	0.47	2.25	2.54	―
プロジェクト4	不具合対応工数(H)	35.60	3.50	113.25	152.35
	トレーサビリティ対策工数(H)	19.00	2.00	116.75	137.75
	工数差(H)	16.60	1.50	-3.50	14.60
	工数(人月)	0.10	0.01	-0.02	0.09
	人月累計(人月)	0.10	0.11	0.09	―

表 21の人月累計をグラフに表したのが、図 46である。

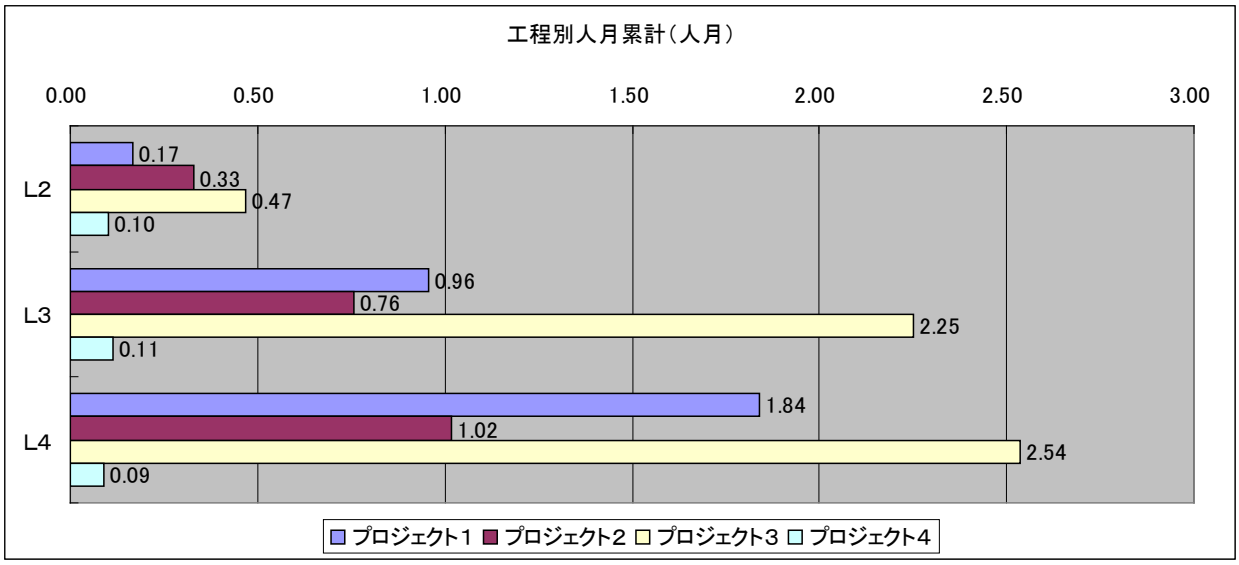


図 46 工程別人月累計

７.２効果の検証

図 ４６で示しているように、プロジェクト１～プロジェクト３については、工程の進行（トレーサビリティの範囲拡大）に従い工数差が拡大する傾向を示しており、トレーサビリティの確保が、コスト削減に有効であることを示している。

このような傾向になる事はあらかじめ予想していたが、実際の検証結果として、これだけの工数（コスト）差があることが明確になった。各プロジェクトにおける工数削減率を表 ２２に示す。

開発工数に対しての削減率は数%であるが、不具合対応工数に対してはプロジェクト１で 53%、プロジェクト２で 22%、プロジェクト３で 44%に相当するため、トレーサビリティ対応を開発当初から、あらかじめ想定して入れておくことで、不具合の大幅な削減効果が期待できる。

表 ２２ 各プロジェクトにおける工数削減率

プロジェクト	削減率 （対開発工数）	削減率 （対不具合対応工数）
プロジェクト１	2.3%	53%
プロジェクト２	1.5%	22%
プロジェクト３	4.1%	44%
プロジェクト４	0.08%	10%

一方、プロジェクト４は、L2、L3での工数差が他のプロジェクトに比べて小さく、L4では工程差が逆転している。これは、プロジェクト ４ においては、トレーサビリティ確保に対する工数削減効果が少ないことを示しており、トレーサビリティに要する工数によっては、コスト増になることを示している。

次に、プロジェクト ４ においてコスト削減効果が少ない／無い要因を、４ つのプロジェクトの不具合情報から考察する。

図 47はプロジェクトごとの不具合対応工数の分布を表したものである。

不具合対応工数が 5 時間未満の不具合件数の割合が高いことは、4 つのプロジェクトが同じ傾向を示している。しかし、図中の赤矢印の部分（対策時間 5 時間以上）の件数は、プロジェクト 1～3は 10 数件あるのに対し、プロジェクト 4 では 1 件のみであった。

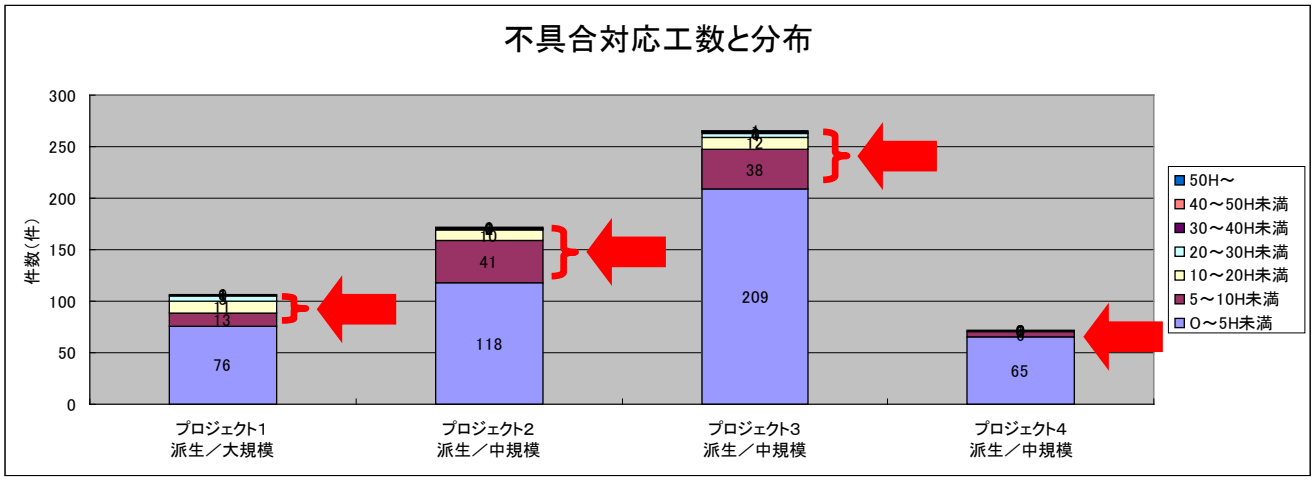


図 47 不具合対応工数と分布

一方、今回の調査で算出したトレーサビリティ対策工数は、「3.3 トレーサビリティ対策工数の試算」で定義した算出式を使用し、PRISMY 情報の処置ステップ数から算出したものであり、この工数が 10 時間以上のものは全体で 14 件だけであった。これは設計要因の全不具合件数のわずか 2%であり、ほとんどのトレーサビリティ対策工数が 10 時間未満であることを示している。

したがって、プロジェクト4では、個々の不具合対応工数とトレーサビリティ対策工数の差が少ない状態であり、しかも不具合対応工数が 5 時間以上の不具合件数も少なかったことにより、トレーサビリティ対策工数との差が少なくなったといえる。

このようなプロジェクトでは、トレーサビリティ対策工数が増加すると、トレーサビリティを取っていない場合との工数差が逆転することも考えられる。

７.３適用レベルとトレーサビリティ範囲における評価結果

「4. 適用レベルとトレーサビリティの範囲について」で記載した、適用レベルと、トレーサビリティ範囲についての、今回の調査結果を表 23に示す。

なお、トレーサビリティなしのL0については現状データのままなので、全４プロジェクトの評価は0になる。また、適用レベルL1の項目については、「4.適用レベルとトレーサビリティの範囲について」で述べたようにデータがないため、今回の調査の対象から除外する。

表 23 評価結果

適用レベル	プロジェクト区分	評価（人月）
L0 トレーサビリティ：全くなし	プロジェクト 1	0
	プロジェクト 2	0
	プロジェクト 3	0
	プロジェクト 4	0
L1 要求仕様書⇒システム設計書	プロジェクト 1	対象外
	プロジェクト 2	対象外
	プロジェクト 3	対象外
	プロジェクト 4	対象外
L2 要求仕様書⇒システム設計書⇒基本設計書	プロジェクト 1	0.17
	プロジェクト 2	0.33
	プロジェクト 3	0.47
	プロジェクト 4	0.10
L3 要求仕様書⇒システム設計書⇒基本設計書 ⇒詳細設計書	プロジェクト 1	0.96
	プロジェクト 2	0.76
	プロジェクト 3	2.25
	プロジェクト 4	0.11
L4 要求仕様書⇒システム設計書⇒基本設計書 ⇒詳細設計書⇒ソースコード	プロジェクト 1	1.84
	プロジェクト 2	1.02
	プロジェクト 3	2.54
	プロジェクト 4	0.09

今回、選定した 4 つのプロジェクトのうち、プロジェクト 4 のみは、適用レベルを L2 から L3 に上げた場合のコスト削減効果が少なく、さらに L3 から L4 に上げるとコスト削減効果が減少することを示している。

対応に 10 時間以上かかった不具合を多く含むプロジェクトは、プログラムの複雑度が高いと想定される。その結果としてトレーサビリティの複雑度が高くなることが想定される。トレーサビリティの観点から考えれば、トレーサビリティの複雑度を小さくするように成果物間の依存関係が成り立つようなプロジェクト管理を行うことで、プログラムの複雑度が低くなり、シンプルなプログラムを作成することが可能になることも想定できる。

今回の調査では、不具合対応に 10 時間以上かかった不具合件数が多いプロジェクトの複雑度が高いと仮定し、工数削減効果との関係を表 24に示す。

表 24 複雑度と工数削減効果

プロジェクト	不具合対応工数が 10 時間以上の不具合件数	工数削減効果
プロジェクト 1	19	53%
プロジェクト 2	13	22%
プロジェクト 3	18	44%
プロジェクト 4	1	10%

今回の調査結果から、表 25のようなことが推測できる。

表 25 トレーサビリティの効果とプロジェクト属性

規模	効果
大規模	◎
中規模	○
小規模	△

複雑度	効果
大	◎
中	○
小	△

今回の調査ではサンプル数が少なかったため、規模と複雑度の組合せについては推測することはできなかった。また、今回の調査対象データは通信ソフトウェアのみだったため、選択肢は限定的であり、様々なパターンのプロジェクトのデータを網羅することはできなかった。

分野に応じ、保証されるべき信頼性・安全性の範囲とコストとの関係を明らかにするには、対象とするソフトウェアの分野を拡げて、さらに調査を行う必要があると考える。

また、トレーサビリティの観点からトレーサビリティの複雑度の低減を図ることによりプログラムがよりシンプルになることも想定されるため、今回の実験とは異なる観点での調査も行う必要があると考える。

８．トレーサビリティに関する管理レベルについて

今回の調査では、設計工程から製造工程までのトレーサビリティのみに注目したが、ソフトウェアの品質向上策としての有効活用を考えれば、トレーサビリティに関する管理レベル（管理上の観点）について、考慮すべき点を挙げる。

８．１トレーサビリティ自体の品質

今回の調査では、トレーサビリティの質については考慮していない。

しかし、トレーサビリティを確保していても、その精度／粒度などの品質が設計品質に影響することは明らかなので、トレーサビリティの品質にも考慮が必要である。

８．２トレーサビリティの保守

継続的に開発されるソフトウェアの場合、トレーサビリティ情報の保守も重要になる。

開発規模やトレーサビリティ管理方法によっては、専任の管理者が必要なケースなどが考えられる。その場合のコストもあらかじめ考慮する必要がある。

８．３トレーサビリティの管理方法

トレーサビリティの有無やトレーサビリティの範囲以外に、トレーサビリティの管理方法にも考慮が必要だと思われる。「8.1 トレーサビリティ自体の品質」や「8.2 トレーサビリティの保守」にも関連するが、専用ツールの利用の有無やその機能等が品質に影響すると考える。

８．４検査工程とのトレーサビリティ

今回は対象外としたが、一般に、設計書と検査仕様書間のトレーサビリティも、ソフトウェア品質の評価では重要であり、今後、データ収集の対象とすることが望まれる。

９．参考文献

[1]SEC BOOKS 「高信頼化ソフトウェアのための開発手法ガイドブック」
独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター編
[2]SEC BOOKS 「組込みソフトウェア開発向け品質作り込みガイド」
独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター編
[3]SEC BOOKS 「ソフトウェア開発データ白書 2010-2011」
独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター編
[4]「ソフトウェア品質説明力強化のための制度フレームワークに関する提案」
（中間報告）(2011 年 9 月 30 日公開)
独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター編

１０．Appendix

１０.１ トレーサビリティの方法【例】

設計書間のトレーサビリティを確保する方法は、要件管理ツールを導入することも 1 つの選択肢であり、早期効果が期待できる。しかし、コストや習熟時間の問題、または対応可能なファイル形式に制限がある場合もある。

ここでは、Excel や MS Office の機能を使用してトレーサビリティを確保する例を示す。

要求仕様	システム設計書	基本設計書	詳細設計書	ソースコード	単体検査仕様書	結合検査仕様書	システム検査仕様書
高齢者でも使い易い	簡易で判りやすい表示	明るい色を使用する			明度：〇〇以上		
					輝度：□□以上		
		アイコンを使用する			アイコンサイズ：△△以上		
	老眼でも見やすい表示	明るい色を使用する			明度：〇〇以上		
					輝度：□□以上		
		大きな文字を使用する			フォント：△△以上		
		：					

図 48 トレーサビリティマトリクスの例

図 48のトレーサビリティマトリクスは、左端に基となる要件を記述し、その右側に各工程において要件に対応する内容を記述する。

例えば、基本設計書の作成の際は、基本設計書の作成の進行に伴って参照したシステム設計書の項目の隣に対応する内容を埋めていく。これにより、基本設計書が完成した際には、要求仕様から基本設計書までのトレーサビリティが確保されていることになる。

この作業を各工程で繰り返すことにより、要求仕様からシステム検査仕様書までのトレーサビリティが確保できる。

なお、この表の作成は各設計書の作成と同時並行で行うことに意味がある。設計書完成後に改めてこの表を作成しようとすると、作成した設計書をもう一度見直すことになり、設計書作成と同程度の工数が必要になることが考えられる。

表 26 トレーサビリティマトリクスのメリット／デメリット

メリット	① 各設計書の記述内容がひと目で比較／確認ができるので、不適切な記述や矛盾を容易に発見することができる。 ② 表中の空欄は仕様や検査項目の漏れを表しているので、仕様のカバレッジの確認を簡単に行うことができる。
デメリット	① 各工程で作成する設計書の内容をコピーして作成する表なので、元の設計書に変更が発生した場合、この表も必ず修正する必要がある。 ② 設計書の関係が 1 対 1 のときはシンプルで分かりやすい表であるが、1 対多になった場合は関係が分かり難くなることが考えられる。

一方、図 48のようなトレーサビリティマトリクスの作成／管理を、各設計書や仕様書の作成と並行して行うのは、限られたスケジュールの中では困難な場合も想定される。

改めてトレーサビリティマトリクスを作成せずに、Word または Excel で作成した設計書に工夫を加えて、トレーサビリティを確保する方法の一例を挙げる。

Word の場合、図 49 の通りコメント機能を使用して関連付け情報を設計書に埋め込む方法がある。

コメント機能であれば、コメントを非表示にすることで成果物としての設計書の体裁に影響することもない。作成する設計書において、基本的に各行にコメント欄を挿入し、ここに関連している上位設計書のコメント番号を記入する。このコメント番号を辿ることでトレーサビリティが確認できるようになる。

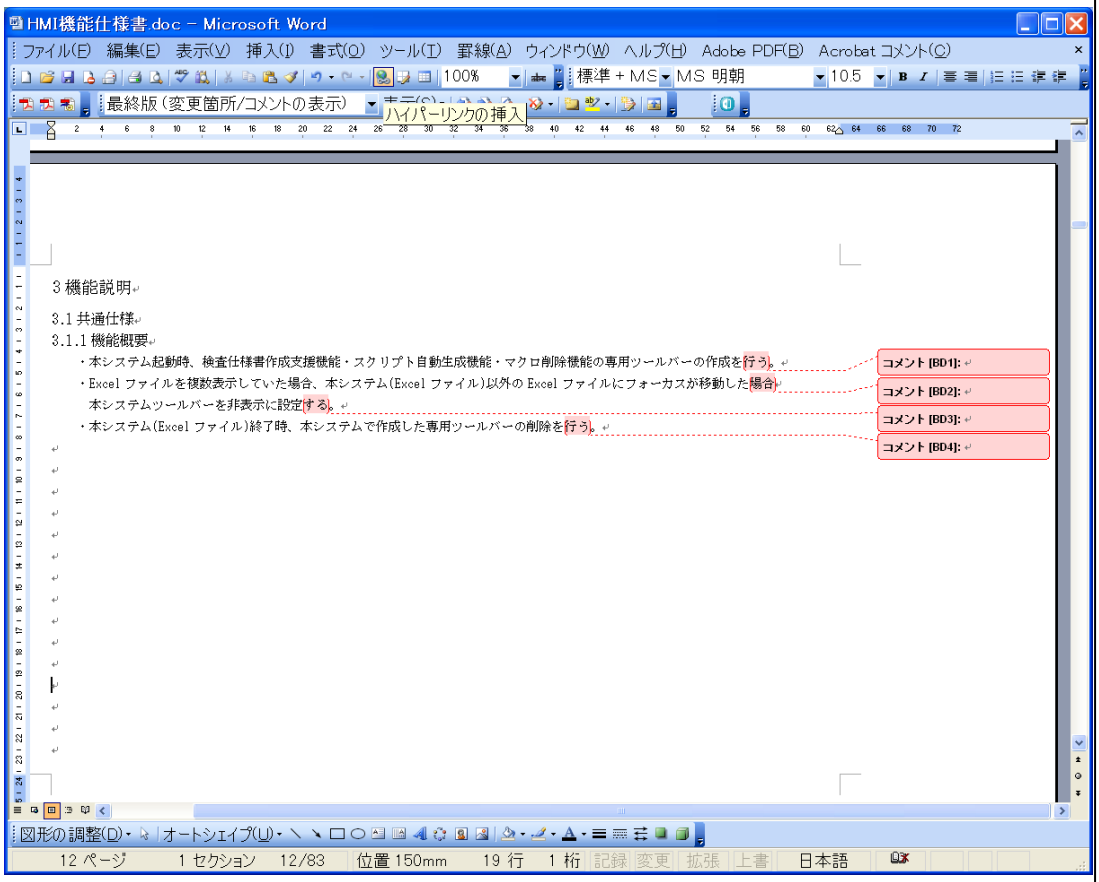


図 49 Word でのトレーサビリティ情報の付加例

Excel の場合は、関連付け情報の列（図 50の赤枠）を追加する。

ここに関連している上位設計書の行番号を記入する。また、単純に行番号だけでは、混乱することも考えられるので、要求仕様書であれば、例えば「UR」を接頭語として、「UR-1」（要求仕様書の1行目）などとするとより明確になる。

その列を非表示にすることで、成果物としての体裁に影響が出ることはない。

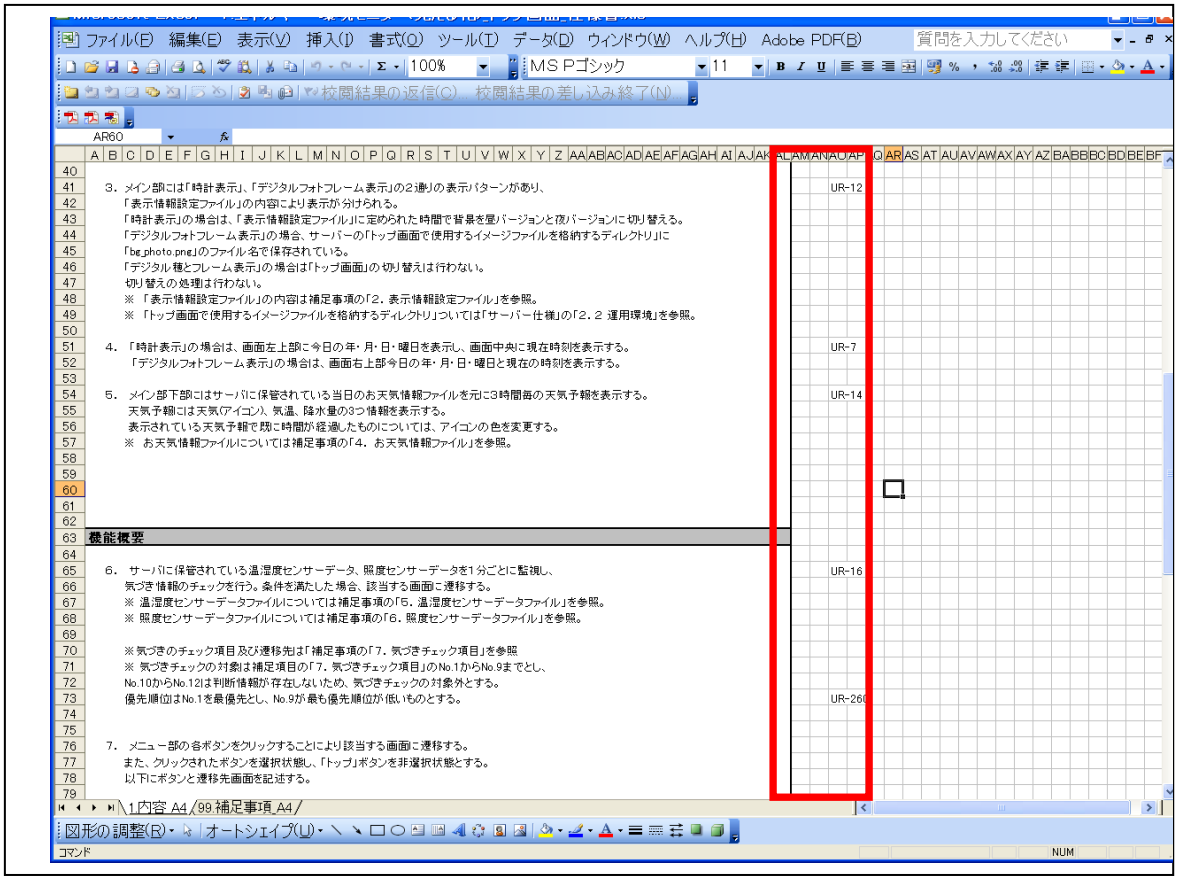


図 50 Excel でのトレーサビリティ情報の付加例

表 27 Word、Excel 使用のメリット／デメリット

メリット	① レーサビリティ情報のために別ファイルを用意する必要がない。 ② 本文の変更に合わせて、別のファイルのトレーサビリティ情報をメンテナンスする必要がない。
デメリット	① トレーサビリティ全体を見渡すのが難しい。 ② 関連先の内容を確認する際は、関連先のファイルを別途開く必要がある。

以上