

実務家のための形式手法

厳密な仕様記述を志すための形式手法入門 第二版

# 事例: 実証実験

## 事例: 実証実験(このモジュールについて)



形式手法導入に関する実証実験から、自らの形式手法導入の検討や計画立案の際に参考となる知見を得る。

### ■ 事前知識・経験

- 形式手法の導入を検討、計画している
- 形式手法の有用性についての基礎知識
- 形式手法導入のガイダンス

### ■ 学習目標

- 実証実験から、自らの形式手法導入の検討や計画立案の際に参考となる知見を得る

### ■ 主な学習項目

- 実証実験の概要と結論
- 形式手法で欠陥が見つかるパターン例

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

2

講師の方へ:

このモジュールでは、IPA/SEC「形式手法適用実証WG」の活動成果である、「情報系の実稼働システムを対象とした形式手法適用実験」の事例を取り上げます。

1. 複数の手法の比較
2. 期待できる効果の分類

といった点を意識するようにお話しください。

ここでのポイントは、つぎのような点です。

1. 用いた形式手法の種類により結果が異なることに注意してください
2. 発見される欠陥はいくつかに分類できます
3. 発見される欠陥にはいくつかのパターンがあります

詳細は、「情報系の実稼働システムを対象とした形式手法適用実験報告書」(<http://sec.ipa.go.jp/reports/20120420.html>)を参照してください。

## 具体的事例: 形式手法適用実証WG実証実験事例



- IPA/SEC「形式手法適用実証WG」の活動として実施
- 実験期間: 2011年8月 ~ 2012年3月
- 参加メンバ(WG委員):

立場	メンバ	実験での役割	DSFのメンバ が中心
ベンダ	株式会社NTTデータ	形式手法の適用 (実験者)	
	富士通株式会社		
	日本電気株式会社		
	株式会社日立製作所		
	株式会社東芝		
	SCSK株式会社		
ユーザ	株式会社東京証券取引所(設計書提供者)	設計書の提供 指摘事項の評価 実験結果の評価	
	住友電気工業株式会社		
学識経 験者	九州大学	実験結果の評価	
	国立情報学研究所		
	名古屋大学		

※DSF(Dependable Software Forum):

障害を起こさないソフトウェアの設計技術を確立し、開発現場に普及させることを目的とした任意団体

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

3

IPA/SEC「形式手法適用実証WG」は、以下の委員で構成した。

- ・形式手法をソフトウェアの設計にどのように使用できるかを評価してきた任意団体DSFのメンバを中心としたベンダ企業の委員
- ・商用システム開発で作成された設計書の提供、及び指摘事項や実験結果を評価するユーザ企業の委員
- ・専門的な観点で実験結果を評価する学識経験者

WGの活動期間は2011年8月～2012年3月までの8ヶ月であるが、報告書作成に2ヶ月を要したので、実際の実験期間は6ヶ月(形式手法適用、指摘事項の評価、実験結果の評価を含む)

※DSF(Dependable Software Forum)

株式会社NTTデータ、富士通株式会社、日本電気株式会社、株式会社日立製作所、株式会社東芝、SCSK株式会社、大学共同利用機関法人 情報・システム研究機構 国立情報学研究所の6社1機関から構成。「形式手法活用ガイド【改訂版】」の完成を受けて2012年6月30日をもって活動を終了。

## 実証実験(続き)



### ■ 使用した形式手法

- Event-B、SPIN、VDM++ の3種類。
  - 知名度が高く利用実績も多い
  - 解説書、支援ツールが手に入りやすい

### ■ 実験チーム

- 形式手法の種別(適用法)に対応して、5つの実験チームを編成。
- 各実験チームが、独立に実験を実施。

チーム	実験対象設計規模(ページ数)		実施体制 (人数)
	形式記述作成対象ページ数	参照を含む総ページ数	
Event-B (1)	110	707	1
Event-B (2)	106	287	3
Event-B (3)	49	381	1
SPIN	109	429	2
VDM++	300	700	5

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

4

使用した形式手法はEvent-B, SPIN, VDM++の3種類。

各自得意な形式手法を利用することとし、5つの実験チームを構成。

各実験チームが独立に実験を実施したため、実際に記述対象とした設計書の範囲は各チームごとにばらつきが出ている。また、重複している範囲もあり。

## 実験対象と実験方法

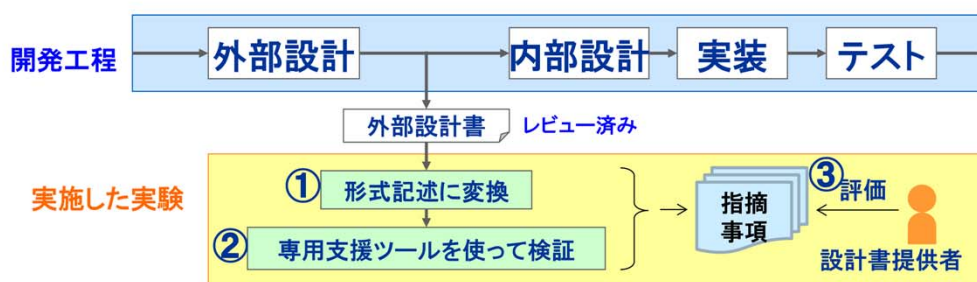


### ■ 実験対象とした設計書

- 東京証券取引所で開発され、運用されている情報システムの設計書を対象
- 外部設計終了(レビュー完了)時点のもの

### ■ 実験の方法

- ① 設計書を形式仕様言語による記述(形式記述)に変換
- ② 専用の検査支援ツールを使って検証
- ③ この作業で見つかった指摘事項をリストアップし、設計書提供者が指摘の妥当性を評価



SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

5

実験対象とした設計書は、実際に東京証券取引所で開発され、運用されている情報システムの設計書である。また、設計書は、通常のレビューは実施済みで外部設計が終了した時点のものである。

指摘事項は、①形式記述に変換し、②専用支援ツールを使って検証する過程において、実験者が不具合の可能性があるものとして抽出した事項である。

指摘事項の妥当性については、設計書提供者が③評価時に実施している。

実験の方法(補足)：

実験者は、今回の東京証券取引所の情報システムの業務内容に精通しているわけではないため、①実施前に、設計書を読解する作業あり。読解時の不明点はチーム間で共有、設計書提供者とのQ&Aを複数回実施して不明点をつぶしていった。

## 形式手法による設計書の検査

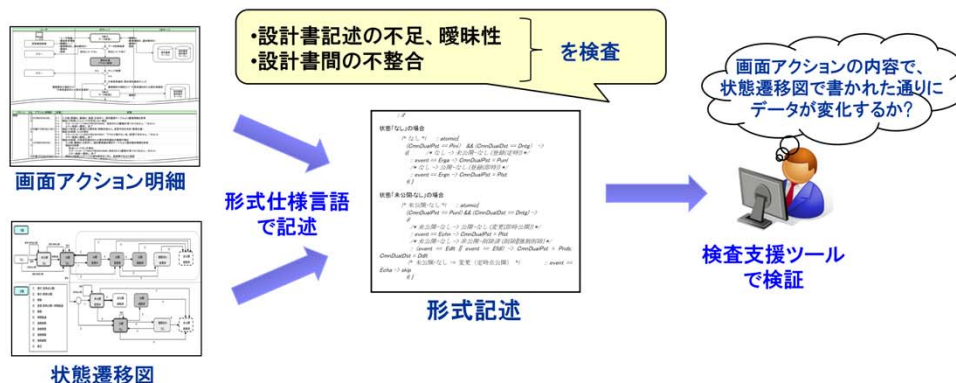


### ■ 形式手法を使う手順

- 『形式手法活用ガイド』（DSF作成）に示される手順で実験を実施

複数の設計書に書かれた仕様が形式仕様言語で記述し、整合性などを検査

例



SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

6

形式手法を適用する手順は、DSFが作成した「形式手法活用ガイド」で示されている手順を利用。

「形式手法活用ガイド」は、2012年6月にDSFからIPA/SECに譲渡された。

下記IPA/SECサイトからダウンロード可能。

<http://sec.ipa.go.jp/reports/20120928.html>

## 指摘事項に対する設計書提供者の評価



### 実験で検出された指摘事項55件に対する設計書提供者の評価

設計書提供者による評価	件数
<b>設計書の修正が必要</b> <b>(実装に影響する可能性がある)</b>	<b>22件</b>
<b>設計書の修正が望ましい</b> <b>(実装に影響する可能性は低い、修正した方が設計書を理解しやすい)</b>	<b>13件</b>
<b>設計書の修正は不要(※)</b>	<b>20件</b>
<b>合計</b>	<b>55件</b>

(※ 実験者の誤解による指摘や、業務への影響がない指摘)

実験で検出された指摘事項は全部で55件である。

指摘事項55件に対して、設計書提供者が評価し、「設計書の修正が必要」「設計書の修正が望ましい」「設計書の修正は不要」の3つに分類した。



## 実際に行われた開発との関係



「修正が必要」と評価された指摘事項22件の内訳

実際の開発における発見時期	件数
後工程(実装・テスト)において発見されていた	13件
実際の開発では指摘されていない (ただし、開発関係者の間では、共通ルールとして徹底されていたため、問題とならなかった)	9件

● 形式手法を使うことにより、従来は実装・テストで発見されていた問題を、設計段階で発見することが可能

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center

8

設計書提供者が「修正が必要」と評価した指摘事項22件について、実際の開発における発見時期という観点で、さらに分析を実施。

22件中、13件が後工程(実装・テスト)において発見されていたことが分かり、実証実験では以下の結論を導いた。

「形式手法を使うことにより、従来は実装・テストで発見されていた問題を、設計段階で発見することが可能」



## 形式手法の作業内容と工数の計測



### ■ 形式手法を適用する作業を、3段階に分類

- ① 設計書の読解と形式化する情報の抽出
  - ② 形式記述の作成
  - ③ 形式記述の検証
- (形式手法によっては、作業をさらに細分化)

### ■ 上記作業毎に、かかった工数と、検出した指摘事項を記録

この実証実験では、形式手法を適用する場合作業工程を細分化し、各作業工程ごとの工数や検出した指摘事項を記録した。

## 作業内容と指摘事項の関係



### 作業内容と指摘件数の関係

設計書の 修正必要性	作業内容			合計
	文書読解と 情報抽出	形式記述の 作成	形式記述の 検証	
修正が必要	1件	19件	2件	22件
修正が望ましい	4件	6件	3件	13件
修正は不要	4件	11件	5件	20件
合計	9件	36件	10件	55件

### ● 形式記述の作成までで、大半の指摘事項を検出

(世の中で知られている知見<sup>(※)</sup>と一致。)

⇒ 厳密な記述が求められるため、設計書を綿密に読み、理解する必要があった。

(※ S. M. Easterbrook, 他, Experiences Using Lightweight Formal Methods for Requirements Modeling. : IEEE Transactions on Software Engineering, Special Issue on Formal Methods in Software Practice, vol. 24, (1), 1988.)

形式記述を作成する場合には、厳密に記述することが求められるため、設計書を綿密に読み、理解する必要があった。

このように、形式記述を作成するとい意識を持って設計書を読むことにより、「文書読解と情報抽出」「形式記述の作成」までで、大半の指摘事項が検出されたという結果が得られたと分析。

## 作業に要した工数



実験チーム	全体工数 (人時)	形式化対象の設 計書ページ数(頁)	作業効率 (人時/頁)	形式記述規 模(行)
Event-B (1)	107.5	110	0.98	1,084
Event-B (2)	64.5	106	0.61	443
Event-B (3)	84.0	49	1.84	425
SPIN	44.5	151	0.29	724
VDM++	254.0	300	0.85	10,866

『ソフトウェア開発データ白書2010-2011』 p. 209

「図表8-3-1 ページあたりの基本設計レビュー実績工数の基本統計量(新規開発)」

N	最小	P25	中央	P75	最大	平均	標準偏差
43	0.018	0.065	0.223	1.166	120.267	12.489	30.260

**75%のプロジェクトにはほぼ収まる。**

● **従来のレビューと大差ない作業効率で、形式手法による設計書の検査ができた**

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center 11

作業工数と形式化対象の設計書ページ数から各チームの作業効率を抽出。

「ソフトウェア開発データ白書2010-2011」の作業効率と比較、75%のプロジェクトでの従来レビューの作業効率にはほぼ収まる結果となっている。

このことから、今回の実証実験では、「従来のレビューと大差ない作業効率で、形式手法による設計書の検査ができた」という結論を導いている。

注意点:

「ソフトウェア開発データ白書2010-2011」の作業効率との比較は、以下の実証実験の前提を考慮していないことは注意すべきである。

「今回の実証実験では、従来レビュー完了済みの設計書を利用している。」

## 形式手法で見つかる欠陥のパターン



なぜ、形式手法で設計書の欠陥が発見できるのでしょうか？

### ■ 主要な3つの欠陥パターンを提示

1. 既存の設計書は「同じであるはずの仕様」が異なる種類の設計書に書かれている
2. 既存の設計書は暗黙知として存在する情報を確認することができない
3. 既存の設計書は自然言語で書かれるため、曖昧な表記で記述することができる

注意: 設計書の欠陥発見はあくまでも形式手法適用効果の1つ

SEC-FM1-05-2

Copyright © 2012,2013 IPA

IPA Software Engineering Center 12

なぜ形式手法で欠陥が発見できるのかというDSFの3つの知見。

1つ目は、既存の設計書は「同じであるはずの仕様」が異なる種類の設計書に異なった仕様として書かれているから

2つ目は、既存の設計書は暗黙知として存在する情報を確認することはできないから。

3つ目は、既存の設計書は自然言語で書かれているため、曖昧であるから。

これらは、レビューで見つかる欠陥といってもよい、”形式手法だから見つかる”というわけではない。

**SEC**  
Software Engineering  
for Mo-No-Zu-Ku-Ri

## パターン 1.

既存の設計書は「同じであるはずの仕様」が異なる種類の設計書に書かれている。

**設計書**

設計の矛盾

離れていると  
気づかない

?

**形式記述**

設計の矛盾

近いと  
わかりやすい!

**発見!**

形式手法は異なる種類の設計書を1つの記述の中に書いて確認するため、異なる種類の設計書間で生じる欠陥を見つけることができる。

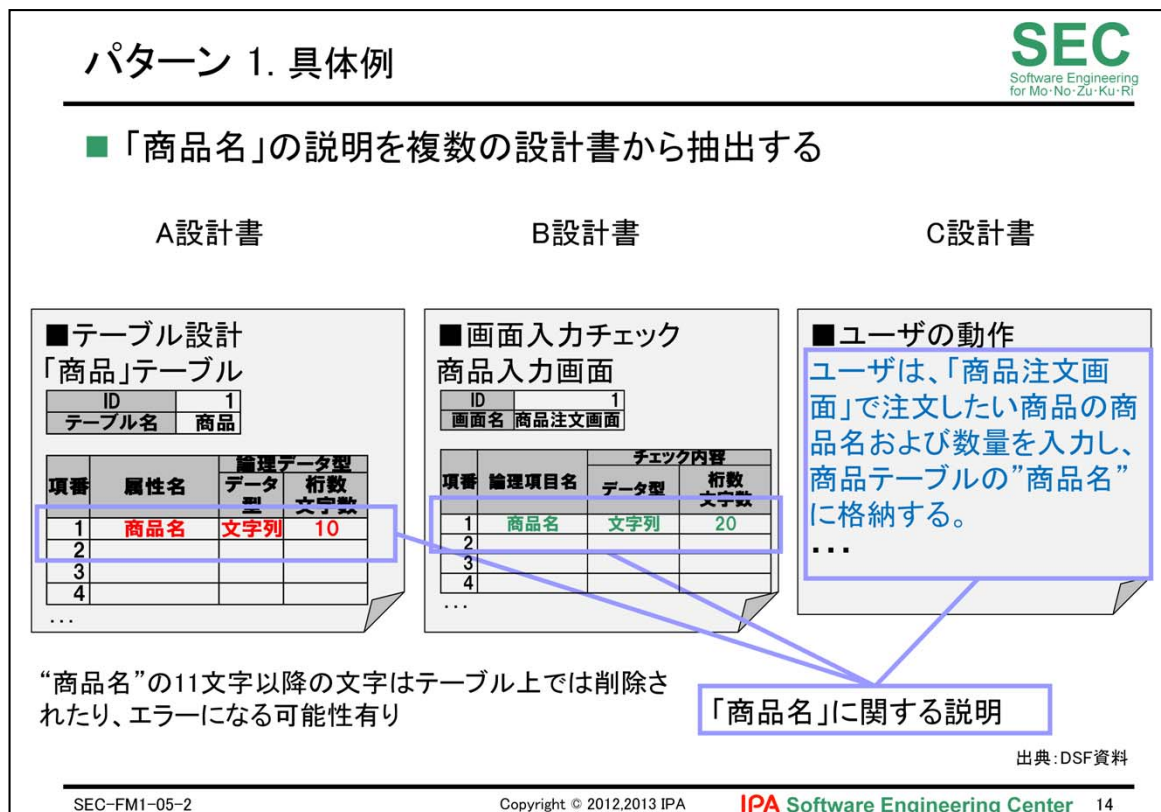
出典: DSF資料

SEC-FM1-05-2      Copyright © 2012,2013 IPA      **IPA Software Engineering Center** 13

1つ目は、既存の設計書は「同じであるはずの仕様」が異なる種類の設計書に書かれており、設計の不整合となっている。

このようにいくつかの設計書に設計が書かれているが、ここ(設計書左枠内)に書いている仕様とここ(設計書右枠内)に書いている仕様が違っているといった時に、2つの設計書であればレビューで見つかる可能性も高いが、5つとか6つに設計書が分かれている中で整合性を見るとなると、一応レビューでも見つけることはできるが非常に大変である。

形式手法で書けばこのように分かれているところも1つの形式記述でまとめて書くので、設計の矛盾しているところが分かる。



例えば、テーブル設計「商品」テーブルでこのような形で表現されている。属性の商品名はデータ型は文字列、桁数は10桁である。

一方、画面入力の方では同じようなテーブルの所の商品名が文字列、桁数20桁となっている。

画面入力なのでここできちんと処理すればいいが、画面で入ったデータをそのままテーブルの中に取り込むという仕様であれば、この商品名の11文字以上の部分はテーブル上で削除されたり、エラーになる可能性もある。

ここをきっちり合わせなければいけない。



## パターン 1. 具体例

SEC-FM1-05-2

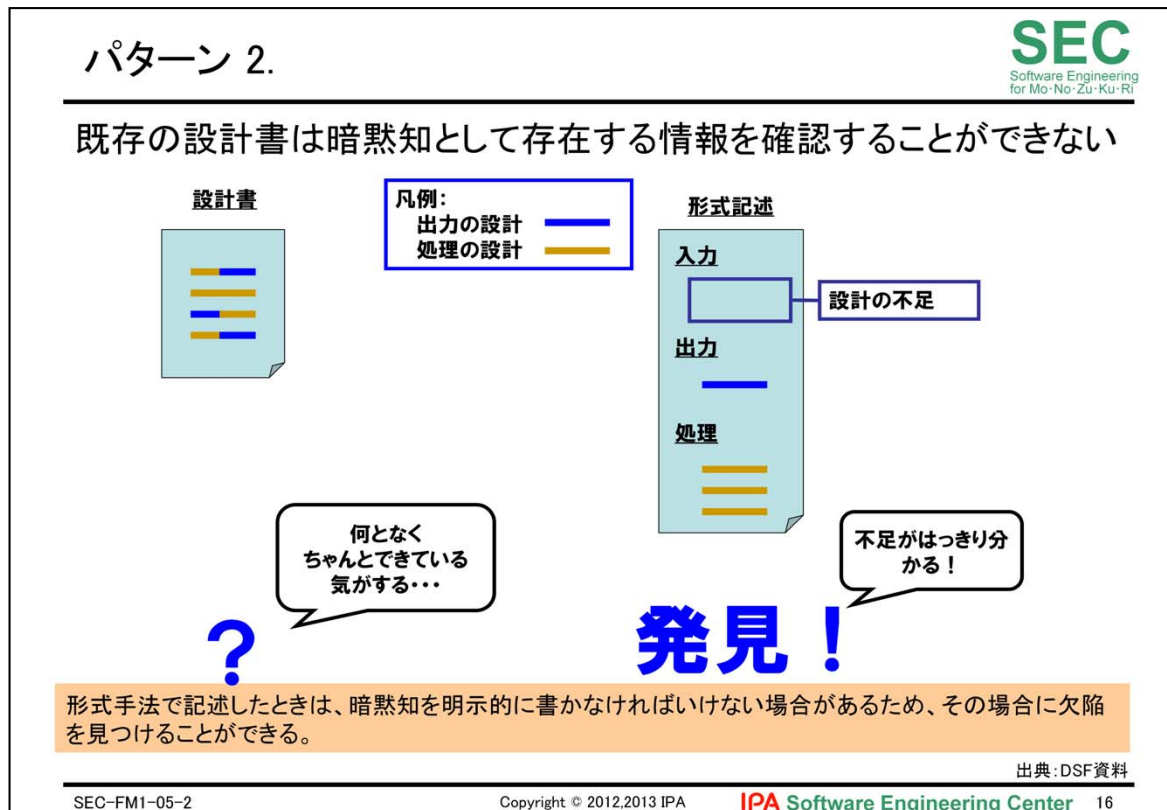
Copyright © 2012,2013 IPA

IPA Software Engineering Center

15

形式手法で書けば、商品名の長さ length が片や 10 以下なのに対して、片や 20 以下となっているので合わないことに気付く。

レビューでは気づきにくい異なる種類の設計書の不整部分が、形式手法で書けば明確に分かる。




2点目、既存の設計書の暗黙知として存在している情報を確認することができない。

設計書は自然言語で書いてあると出力や処理の設計が色々なところに散らばっている。

これだとなんとなく書いてある気がして、レビューでは中々見つけにくい。

形式手法だと構造化して書かなければならないので設計が不足している部分が分かる。

## パターン 2. 具体例



### ■「商品名」の説明を複数の設計書から抽出する

#### ■ユーザおよびシステムの動作

1. ユーザは、「商品検索画面」で商品を検索する。

2. システムは、「商品検索画面」に商品の一覧を表示する。

「商品を検索する」の説明

設計書A

出典: DSF資料

SEC-FM1-05-2 Copyright © 2012,2013 IPA IPA Software Engineering Center 17

その例で、日本語ではなんとなく書いてあるが、実際にはここで何を検索条件とするかが書いていない。

## パターン 2. 具体例

■ 抽出した情報を形式記述で書く

```
商品
{
  ...

  商品を検索する( ??? ) ==> set of 商品
  {
    return ??? に該当するすべての商品
  }
}
```

暗黙知になっている箇所

何を検索条件とする  
のだろう???  
うまく書けないな...

形式記述


?

出典: DSF資料

SEC-FM1-05-2 Copyright © 2012,2013 IPA IPA Software Engineering Center 18

形式手法を用いると???の部分を書いていないということが分かる。

## パターン 2. 具体例



### ■ 矛盾に気づく

```
商品
{
  ...
  商品を検索する(名前) => set of 商品
  {
    return (商品 | 商品.商品名 = 名前)
  }
}
```

追記

形式知にできた！

**発見！**

形式記述

SEC-FM1-05-2Copyright © 2012,2013 IPAIPA Software Engineering Center19

暗黙知であった検索条件が明示される。

パターン 3.

SEC  
Software Engineering  
for Mo-No-Zu-Ku-Ri

既存の設計書は自然言語で書かれるため、曖昧な表記で記述することができる

設計書	設計の曖昧さ	形式記述
「このボタンは、 チェックボックスA,B,Cがチェック されないとき に出現する。」		<pre>If (   A = false &amp;&amp; B = false &amp;&amp; C = false もしくは   A = false    B = false    C = false )   ボタンが出現する; ...</pre>

すっきりしないけど  
何となく書けちゃった

2通りの解釈ができる。  
設計が曖昧だ！

**？** **発見！**

形式手法は厳密なルールによる数式で記述されるため、曖昧な部分は記述することができず、欠陥として抽出される。

出典: DSF資料

SEC-FM1-05-2 Copyright © 2012,2013 IPA IPA Software Engineering Center 20

3つ目、既存の設計書は自然言語で書かれているため曖昧な記述をすることができる。

ここには

『このボタンは、チェックボックスA、B、Cがチェックされないときに出現する。』

と書かれている。

これでは、『チェックボックスA、B、Cがチェックされないとき』の部分の意味が、

『A、B、C すべてのチェックボックスがチェックされていない』つまり、

A、B、Cすべてがfalseのパターンなのか

『A、B、C のどれかがチェックボックスがチェックされていない』つまり、

A、B、Cのどれかがfalseのパターンなのか

分かりにくい。

要求仕様を確認にして、正しい仕様を表現するようにしないといけない。

この3つが主に形式手法で見つかる欠陥ではないかと考えられる。



## サマリー



形式手法導入に関する実証実験から、自らの形式手法導入の検討や計画立案の際に参考となる知見を得るために、以下を学習

- 実証実験の概要と結論
- 形式手法で欠陥が見つかるパターン例



---

## 実務家のための形式手法

### 厳密な仕様記述を志すための形式手法入門

## 事例:実証実験

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  
統合系システム・ソフトウェア信頼性基盤整備推進委員会  
上流品質技術部会 人材育成WG(編)  
2013年3月 第二版発行

記載されている個々の情報に関しての著作権及び商標はそれぞれの権利者に帰属するものです  
なお、本書の内容は将来予告なしに変更することがあります