

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

なぜ形式手法か

品質の高いソフトウェアの効率よい開発へ向け、形式手法の有用性を理解した上で、形式手法の導入に前向きに検討する姿勢の獲得を意図したモジュール。特に正しい仕様が重要であることを理解する。

■ 事前知識・経験

- 旧来のソフトウェア開発に対する問題意識(ソフトウェア開発の実経験がある
とより好ましい)
- 形式手法の知識・スキルは前提としない

■ 学習目標

- 形式手法とその効果について概略レベルの知識を得る
- 旧来の開発の課題に対する形式手法の有用性を理解し、導入の検討を開始できる
- 教材中の例や自身の経験との関連付けにより、仕様の重要性を理解し、まず正しい仕様の記述に焦点を当てた形式手法導入の検討を開始できる

■ 主な学習項目

1. 背景 p4
2. 形式手法の定義と期待できる成果 p6
3. ソフトウェア開発の課題と形式手法 p9
4. ソフトウェアの信頼性・安全性への期待の高まりと国際規格・標準での形式手法の推奨 p15
5. 慣習的手法とその限界、およびその解決に有用な形式手法の特性 p23
6. 現場向きの形式手法VDMの概要 p34
7. 自然言語や慣習的な非形式的記述による問題例 p39

1. 背景

- ソフトウェアの信頼性や安全性に対する関心が高まってきた
- これまで以上に効率良く開発することが求められてきている
- 普通の現場で開発に携わっているのは**普通のチームとメンバ**
- 現場にプレッシャーをかけても品質は高くなり、ストレスばかりがかかる
- 効率良く品質の高いソフトウェアを開発するための「**銀の弾丸**」はない
- 形式手法は科学的に裏づけを持つ手法

2. 形式手法の定義と期待できる成果

- 形式手法とは、システム開発に用いられる手法で、**数理論理学**に基づく科学的な裏付けを持つ仕様記述言語を用いて設計対象を表現することにより、**ある側面の仕様を厳密に記述**し、開発の**各工程で利用**する手段の総称である
- 形式記述を行うことで曖昧さが取り除かれ、**機械処理が可能**になり、様々な可能性が開ける
- 形式手法の一分野であるモデル規範型の手法と、形式仕様記述言語、仕様の記述や検証のためのツールの活用により、厳密な仕様の記述や検証が可能となる上、形式手法の導入は、システム開発におけるステークホルダ間の**コミュニケーションの活性化**に寄与する

形式手法による仕様作成の成果例

	対象システム	適用工程	仕様規模	欠陥数	生産性	開発期間	備考
(1) SCSK	証券業務	実システム UseCaseレベル 要求仕様	3+3万行 仕様+テスト ケース	0 品質	2.5倍 効率	45%	開発チームに 業務知識無し
(2) フェリカ ネットワークス	おサイフ ケータイ ファーム ウェア	実システム API外部仕様	7.4+6.6万行 仕様+テスト ケース	0	2倍	85%	開発チームに 形式手法知識 無し
(3) 産業技術 総合研究所, オムロン	鉄道関係 駅務シス テムデー タ	既存システム 厳密仕様 作成実験	0.75+80万 行 仕様+業務	欠陥 発見 数 29			「暗黙知は 仕様の欠陥」

3. ソフトウェア開発の課題と形式手法

- ソフトウェアの開発現場では、曖昧な仕様に起因するトラブルが多い
- 下流工程の修正コストは、上流工程よりも大きくなる
- 正しくないプログラムの欠陥修正は非常に難しい、または不可能である
- 自然言語により、曖昧かつ不完全な仕様を記述している
- 仕様記述と称して、日本語プログラミングを行っている
- 対象業務の内容をよく知らないプログラマがプログラミングを行っている
- 結果として、重大な欠陥を含み、保守性・再利用性のないソフトウェアが生産されている

- 口頭
- ホワイトボード
- 「パワーポイント」独自スタイル
- 表
- UML
- 自然言語
- 「VDM」、形式仕様記述言語

- 「これから配布する文書の、ある文字の数をかぞえる【7 分間】

■ 日本語で正しい仕様を書くのは「不可能」である

- 曖昧さの排除が困難
- ツールによるチェックが、ほとんど不可能
- 仕様の修正に弱い
- その場で「文法」を考えられない

□ 日本語で仕様を書ききれなくなり、if-then-else など擬似コード的な仕様を書くことが多く、**擬似コードの文法を考えている時間が馬鹿にならない**

■ 形式手法の方が**技術移転**や**蓄積**が容易

- 日本語による意思疎通は難しい
- 仕様記述言語を読むのは容易
- **フレームワークやライブラリの構築**が容易
- 業務知識の蓄積が容易
- 仕様を「**動かしてみる**」ことができるので、理解しやすい

■ プログラムは数学系だから

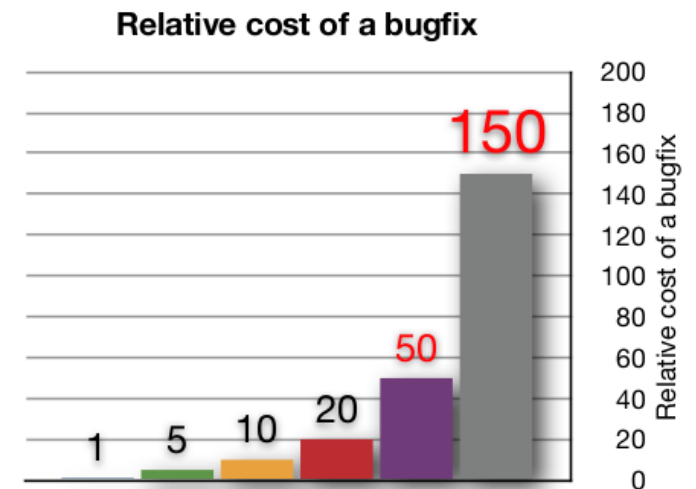
- しかも, CやJavaのプログラムは「**検証が困難な数学系**」
- 要求仕様記述工程からプログラミング工程までのどこかで、数学系に変更しなければならない
- **検証しやすい数学系で記述した方が、品質を上げやすい**

■ 形式手法では、上流工程で仕様を検証

- **欠陥修正コストが10分の1から200分の1で済む**([Boehm:07],他)

■ 旧来の手法では、検証が困難

- **レビューだけでは限界**がある



Source: Barry Boehm: "EQUITY Keynote Address", March 19th, 2007

■ Req ■ Design ■ Code ■ DevT ■ AccT ■ Ops

4. ソフトウェアの信頼性・安全性への 期待の高まりと国際規格・標準での 形式手法の推奨

■ Safety Critical System／Mission Critical System

- 原子力発電所制御システム
- 航空管制システム
- 鉄道信号制御システム
- 放射線治療システム

■ 日常生活

- 銀行オンラインシステム

■ ディペンダビリティ (dependability)

- 信頼性、安全性、頑健性、説明可能性、等々

- 鉄道改札機
- 東京証券取引所
- みずほ銀行
- 航空管制システム
- 搭乗手続きカウンタ
- 電話交換機
- 携帯電話リコール

etc.

「情報システムの障害データ. 情報システムの障害状況」

SECジャーナル, No.27, pp.150-152, 2012

SECジャーナル, No.28, pp.6-8, 2012

SECジャーナル, No.30, pp.139-141, 2012.

■ 経緯

- 欧米各国独自の情報技術セキュリティ評価基準では、相互認証は困難であったため、1999 年に国際標準として採択
- 日本では:2000年に、JISX5070 として採択

■ 範囲

- セキュリティ製品（ハードウェア・ソフトウェア）およびシステムの開発や製造、運用
- 製品やシステムが備えるべきセキュリティ機能に関する機能要件と、設計から製品化に至る過程でセキュリティ機能が実現されていることを確認する保証要件を網羅した要件集
- セキュリティについての基本概念の説明とともに、評価対象となる製品やシステムのセキュリティ基本仕様を記述するセキュリティターゲット (ST) や、ST のベースとなる文書であるプロテクションプロファイル (PP) についても規定

■ 以下のパートから構成

- Part 1. 総則および一般モデル
- Part 2. セキュリティ機能要件
- Part 3. セキュリティ保証要件

■ Part 3 では評価保証レベル Evaluation Assurance Level (EAL) として、7 段階のレベルで保証要件が規定されている

- EAL 1: 機能テストの保証
- EAL 2: 構造テストの保証
- EAL 3: 系統的テストおよび確認の保証
- EAL 4: 系統的計画およびテスト、レビューの保証
- EAL 5: 準形式的設計とテストの保証
- EAL 6: 準形式的な設計の検証とテストの保証
- EAL 7: 形式的な設計の検証およびテストの保証

■ EAL 1～3 が一般向けで、EAL 4 が政府機関向け、EAL 5 ～ 7 が 軍用・政府最高機密機関レベル

- リスクの事前評価=「絶対安全は存在しない」
- リスクの軽減=許容範囲内におさめる
- IEC 61508:2000
 - Functional safety of electrical/electronic/programmable electronic safety-related systems
- JISCO508:2000
 - 「電気・電子・プログラマブル電子安全関連系の機能安全」
- 安全尺度 **SIL** (Safety Integrity Level)

SIL	信頼性	連続稼動設備における1時間あたりの故障発生率
1	90%以上	10^{-6} 以上 10^{-5} 以下
2	99%以上	10^{-7} 以上 10^{-6} 以下
3	99.9%以上	10^{-8} 以上 10^{-7} 以下
4	99.99%以上	10^{-9} 以上 10^{-8} 以下

規格種別	規格名称	対象
EN規格	EN 61548	一般(計装)
	EN 50126	鉄道
IEC規格	IEC 6151	プロセス産業
	IEC 61513	原子力
	IEC 62061	産業機械
	IEC 62304	医療
	IEC 62800	モータ
ISO規格	ISO 26262	自動車
指針類	EEMUA	アラーム指針
	DFT STAN	防衛(英国)
	海軍規格(英国D	海軍(英国)
	MISRA-SA	自動車(英国)
	計量ソフトウェア指針	モータ
	EMCガイドライン	IEE(英国)

表 3 IEC 61508 における形式手法の推奨例 (R:Recommended, HR:Highly Recommended)

Software safety requirements specification				
Technique/Measure	SIL1	SIL2	SIL3	SIL4
1 Computer-aided specification tools	R	R	HR	HR
2a Semi-formal methods	R	R	HR	HR
2b Formal methods including for example, CCS, CSP, HOL, LOTOS, OBJ, temporal logic, VDM and Z	—	R	R	HR
Software design and development: detailed design				
Technique/Measure	SIL1	SIL2	SIL3	SIL4
1a Structured methods including for example, JSD, MASCOT, SADT and Yourdon	HR	HR	HR	HR
1b Semi-formal methods	R	HR	HR	HR
1c Formal methods including for example, CCS, CSP, HOL, LOTOS, OBJ, temporal logic, VDM and Z	—	R	R	HR
2 Computer-aided design tools	R	R	HR	HR
3 Defensive programming	—	R	HR	HR
4 Modular approach	HR	HR	HR	HR
5 Design and coding standards	R	HR	HR	HR
6 Structured programming	HR	HR	HR	HR
7 Use of trusted/verified software modules and components(if available)	R	HR	HR	HR

(出典:IEC 61508 Part1 Annex A より作成)

5. 慣習的手法とその限界、および その解決に有用な形式手法の特性

- ソフトウェアのレビュー
- テスト
- 検証、証明

■ 開発の各工程ごとに成果物を検査・評価すること

- ウォークスルー(walk-through)
 - 開発者が進行役となって実施
 - エラーの早期発見が目的
 - 修正の検討と確認はしない(開発者の責任)
 - 資料を事前配付
- インспекション(inspection)
 - モデレータが進行役となって実施
 - エラーの早期発見と修正の検討が目的
 - 修正結果まで追跡確認(モデレータの責任)
 - ウォークスルーよりは形式的
 - 資料を事前配付
- ラウンドロビン(round-robin)
 - レビューメンバが持ち回りでレビュー責任者を務める

- 「プログラムのテストは、**バグの存在を示すには極めて有効**であるが、バグが**存在しないことを示すには絶望的に無力**である」

[E. Dijkstra]

- 不具合の検出
- 信頼性に対する確信度
- 品質指標を用いたテスト項目の作成
 - 要求品質
 - 顧客に依存
 - 品質指標で危険の掘り下げ
 - 品質特性の分類
 - テスト項目
 - 漏れなく、顧客が認める評価項目

- 「納得できる正当性の証明が、必要な確信のレベルに到達する唯一の方法であるように思われる」[E.Dijkstra]
 - 公理系に基づく証明
 - 証明の記述、チェック、理解、評価
 - 「プログラムが大規模・複雑になればなるほど、検証が有効・重要になる(何故なら、テストではいよいよ解決できない)」
- 形式手法の基礎
- 理論体系の構築
 - 対象=システム、性質
 - 理論と現実
- 証明することの副産物
 - 概念の明確化
 - 曖昧さのない議論と記述

- プログラムの正しさ(検証)=1960年代～
- 検証理論に関する研究の発展=1970年代～
- 形式仕様記述言語/方法論/ツールの開発
 - VDL(Vienna Description Language),VDM(Vienna Development Method)
 - Z Notation
 - Bファミリー(B Method, Event- B)
 - Larch
 - OBJ
 - HOL
 - モデル検査etc.
- 適用事例蓄積と実用化

J. P .Bowen, et al. : Ten commandments of formal method,
IEEE Computer 28,No.4,pp.55–63(1995)

■ レベル0: 形式仕様記述

- 数学的な記法を用いて厳密な仕様を記述し、証明や分析までは行わずに、この記述を基にしてプログラムを開発する

■ レベル1: 形式的開発および検証

- プログラムの性質を証明したり、詳細化により仕様からプログラムを作成する

■ レベル2: 機械支援による証明

- 定理証明器や証明支援器を用いてプログラムの性質を証明する

どのレベルでもまず厳密な仕様の記述が必要であり、その記述を行うだけでも(レベル0でも)効果が期待できることが多い

■ 直接的効果

- 各種記述物
 - 記述物(要求、仕様、設計、コード、テスト、バグ、etc.)
 - 証明結果、分析

■ 間接的效果

- 開発対象の認識と理解
- 開発プロセス改善
- コミュニケーション
- 開発者の自信
- 高品質ソフトウェアの効率的開発

■ 「形式手法」とは「書くこと」

- 書くために理解、認識、議論、分析、共有、etc.
- 書いてあること V.S. 書いていないこと
「決めること」

フェリカネットワークスでの形式手法適用事例より

■ 自然言語・形式仕様記述言語の違い

→「理解」と「明確化依頼」以外は似通っている

■ 自然言語

→ まず書いてあることがよくわからない

…「明確化依頼」=「(書いてあることが)わかった」

でとどまってしまう

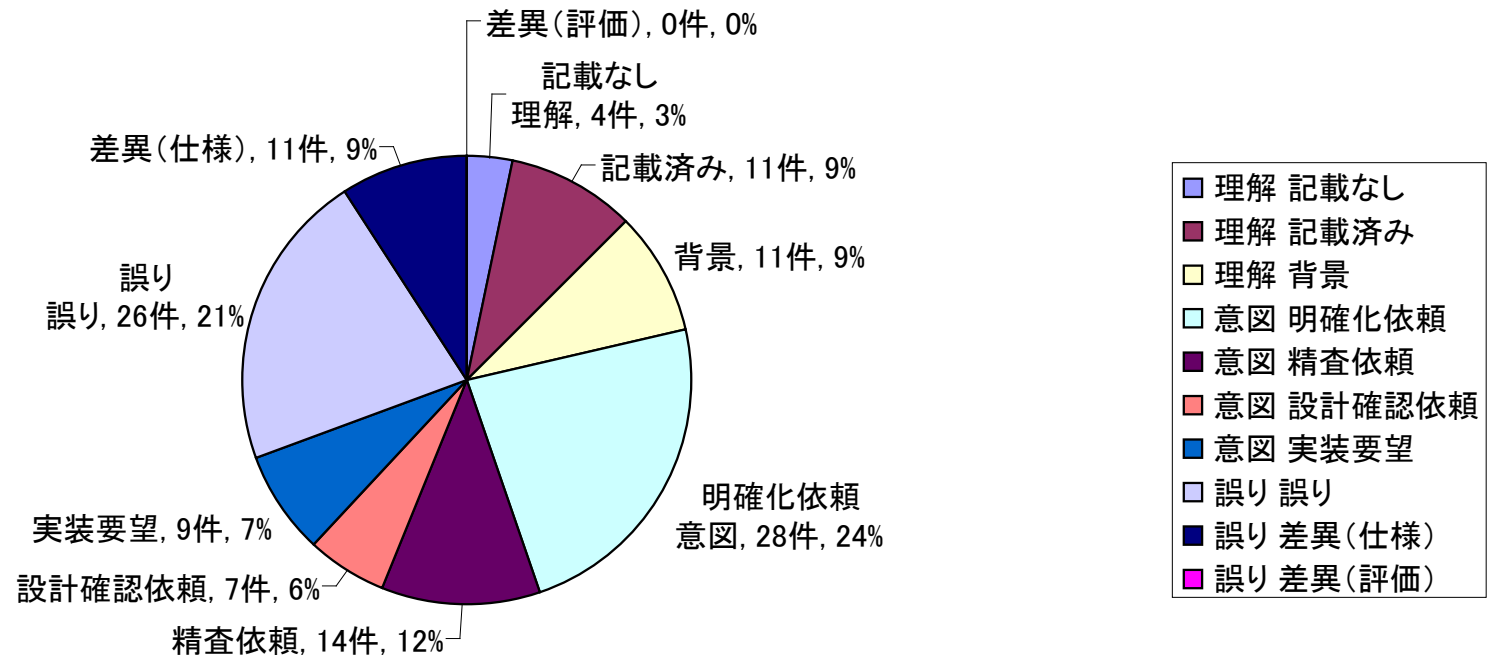
■ 形式仕様記述言語

→ 書いてあることはわかる…中途半端で断片的な「理解」だったと自覚できるので、背景・根拠を含めより深く「理解したい」、「納得したい」につながる

■ 日本語での議論は(精神的にも)つらい？

→ 記述から人格を消して「問題対私たち」とする

■ 自然言語によるマニュアル → 明確化依頼が多い

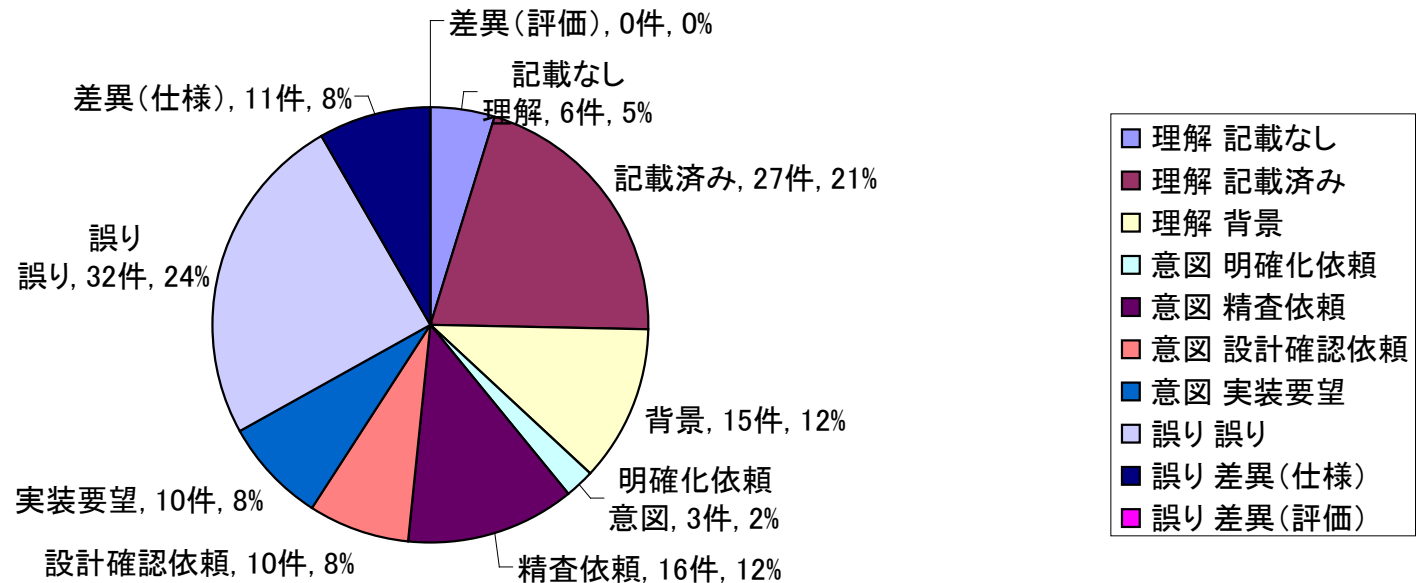


栗田太郎: 携帯電話組み込み用モバイル FeliCa IC チップ開発における形式仕様記述手法の適用,
情報処理情報処理 Vol.49, No.5, pp.506-513, 2008年5月より引用

■ 形式仕様記述言語による外部仕様書

→ 「理解」に関する質問が多い

→ 仕様策定背景・経緯はコメントに記述する



栗田太郎: 携帯電話組み込み用モバイル FeliCa IC チップ開発における形式仕様記述手法の適用,
情報処理情報処理Vol.49, No.5, pp.506-513, 2008年5月より引用

6. 現場向きの形式手法VDMの概要

■ 数学をベースとした仕様記述と検証のための手法

- 1960年代から1970年代にIBMウィーン研究所で開発
- 1996年に、VDM-SLが世界初のISO標準(ISO/IEC 13817)仕様記述言語になった

■ 特徴

- 厳密に定義された仕様記述言語 VDM-SL, VDM++, VDM-RT を持つ
- 制約条件(不変条件、事後条件、事前条件)の記述が可能
- ツールによる証明課題の生成
- 産業界の実用のために拡張された

- 理論的にも経験的にも形式手法が有効
- VDMToolsで使用するVDMは、**現場寄り**の形式手法
 - 証明やモデル検査は、長期的にはやるべきだが、**すぐにはできない**
- VDM導入は、さほど難しくない
 - 3ヶ月程度の教育とコンサルティング
 - 既存の役立つ**ソフトウェア工学ツール**と協調して、より効果が出る
- モデル化は、以下が重要
 - モデル化の**範囲を決め**、仕様を**分割統治**
 - 名詞から型、述語から関数または操作
 - 陰仕様を作成してから、**静的検証**
 - 陽仕様を作成してから、**動的検証**
- VDMは構造化日本語仕様として使うことができる

VDMは構造化日本語仕様として使える。(慣習的日本語仕様は使えない。)

- 構文を考える時間が不要である
- 構文・型チェック、証明課題レビューで静的に検証できる
- 証明課題で生成される条件式から、見落としていた不変条件や事前条件が見つかる
- 組合せテストにより、動的に正当性検証ができる
- 回帰テストにより、動的に妥当性検査ができる
- VDMソース自体が、要求辞書ともなる
- 日本語仕様より、記述と検証の工数が少ない
 - 特に、仕様修正に強い
- 擬似コード形式の日本語仕様に近い形で、かなりの部分を記述できる

VDMと慣習的日本語仕様：構造化日本語仕様か？ ※

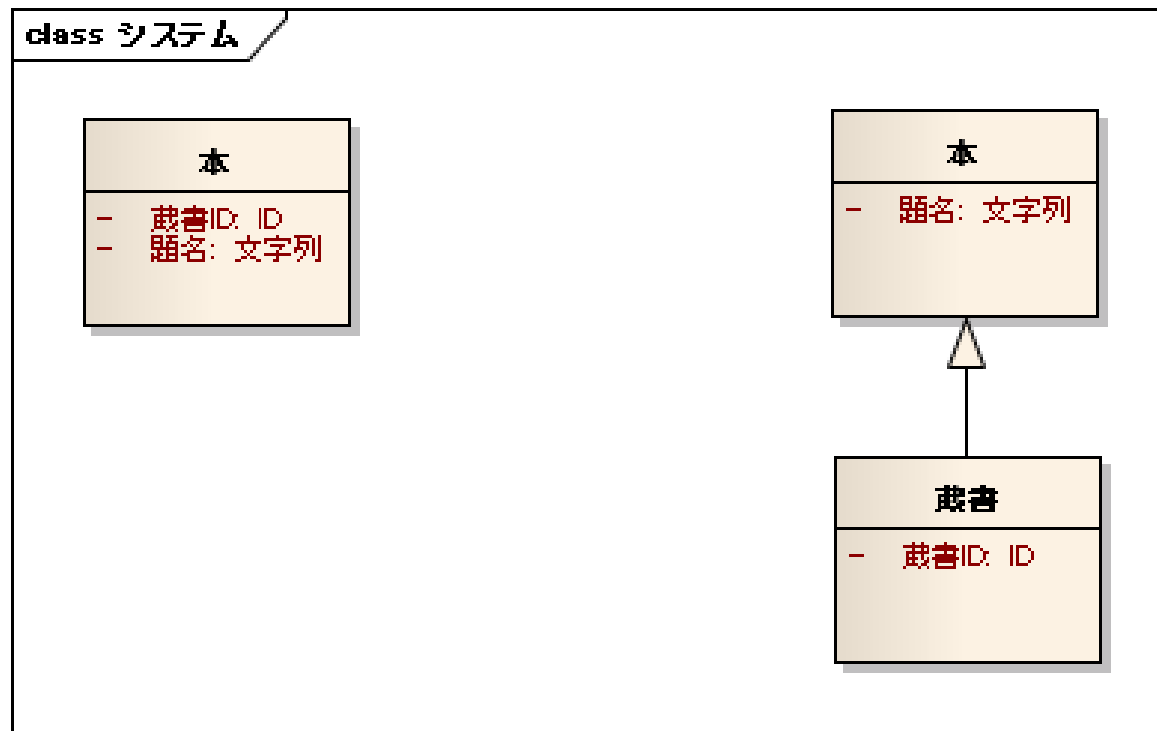
	構文を考える時間	構文チェック	関連チェック	実行テスト	証明問題生成
擬似コードによる仕様	かなりの時間が必要で記法も統一できない	レビューのみ	レビューのみ	具体的データを想定したコードインスペクションに相当するチェックのみ (通常行なわれない)	不可能
VDM仕様	言語マニュアルを参照すれば良いだけ	ツールでチェック	ツールの型チェック	ツールで実行	ツールで生成

7. 自然言語や慣習的な非形式的記述 による問題例

クイズ

■ 以下の2つの「本」は同じか？

- 本を図書館の蔵書として追加する
- 題名で本を検索する



■ 鉄道会社A 特急券予約サポートセンターとの問答

- クレジットカードを変更したら、予約できないんですが?
 - **カード**を変更したら、特急券予約を新規に契約して下さい
- クレジットカード変更前に予約した特急券に引き替えようとしたらできないんですが?
 - **カード**で引き替えできるようになったので、それで引換えて下さい
暗証番号は**カード**のものを使って下さい
- カードの暗証番号って、何ですか?
 - クレジットカードの暗証番号です

■ T駅での問答

- クレジットカードで特急券に引換えできないんですが
 - 会員証でやってみて下さい
- 会員証でもできないんですが
 - 変ですねー、こちら(駅員)でやってみましょう。駄目ですねー
- ひょっとして、古いクレジットカードではどうですか?
 - あ、できました。はい、切符です

■ カードの種類

- 予約会員証（変更前、変更後）、クレジットカード（変更前、変更後）
- その後、今回の件とは無関係と判明したカード
 - ICカード、予約カード

■ 鉄道会社A 特急券予約サポートセンターとの問答

- クレジットカードを変更したら、予約できないんですが?
 - クレジットカードを変更したら、特急券予約を新規に契約して下さい
- クレジットカード変更前に予約した特急券に引き替えようとしたらできないんですが?
 - 予約会員証で引き替えできるようになったので、それで引換えて下さい。
暗証番号はクレジットカードのものを使って下さい

■ T駅での問答

- 変更後のクレジットカードで特急券に引換えできないんですが
 - 予約会員証でやってみて下さい
- 予約会員証でもできないんですが
 - 変ですねー、こちら(駅員)でやってみましょう。駄目ですねー
- ひょっとして、変更前のクレジットカードではどうですか?
 - あ、できました。はい、切符です

■ 要求仕様モデルの考慮不足

- 鉄道会社Bのクレジットカードを変更すると、特急券予約会員として新規に契約せねばならず、予約会員証も新規に作成
 - 変更という概念が無く、ユーザに不便を強いている
- 個々の予約が、クレジットカードにリンク
 - クレジットカード変更に弱く、ユーザにも駅員にも分かりにくい
- 予約会員証からクレジットカードにリンク
 - クレジットカード変更に弱い、ユーザにも駅員にも分かりにくい

■ 信用取引の決済日(期日)を得る

弁済期限とは、信用建玉に対して当社がお客様に信用を供与する期限をいいます。弁済期限は、現在のところ6ヶ月のみを取扱っています

弁済期限が6ヶ月であるということは、信用建玉の建日(信用建玉が約定した日)の6ヶ月目応当日が信用期日となり、この日を超えて建玉を保有することは法律で禁じられています。信用期日が休日の場合には、直近の前営業日が信用期日となります

- 弁済期限と決済日と信用期日との関係は？
- 応当日とは？
- 応当日が月末日を超えたらどうなるのか？
- 直近の前営業日が5ヶ月後になった場合はどうするのか？

サマリー

品質の高いソフトウェアの効率よい開発へ向け、形式手法とその効果について概略レベルの知識の獲得し、旧来の開発の課題に対する形式手法の有用性を理解し、仕様の重要性和正しい仕様の記述に対する形式手法の有用性を理解した上で、導入を検討するために、主に以下を学習

- 形式手法の定義と期待できる成果
- ソフトウェア開発の課題と形式手法
- ソフトウェアの信頼性・安全性への期待の高まりと国際規格・標準での形式手法の推奨
- 慣習的手法とその限界、およびその解決に有用な形式手法の特性
- 現場向きの形式手法VDMの概要
- 自然言語や慣習的な非形式的記述による問題例

- 数理的な道具立て/方法に基づくシステムの開発方法
 - 記述、分析、設計、実現、検証、保守
- 30年以上の歴史
- 多種多様な理論、方法論、ツール例えば、J.Bowenのwebでは、100種類以上
[http://formalmethods.wikia.com/wiki/Formal_Methods_Wiki]

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

なぜ形式手法か

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