Vol.2014-SE-186 No.14
2014/11/14

通信キャリア向けサービス制御ノードの開発における形式仕様策定

馬 庚元[1]　林 信宏[2]　大森 洋一[2]　日下部 茂[2]　荒木 啓二郎[2]　吉武 浩[3]　古賀 昭男[3]　原田 英治[3]
杜 翔[3]　井上 康生[3]

概要：本稿では、情報処理推進機構 (IPA) がリリースした形式手法適用手順 (VDM++編) に基づいて通信キャリア向けサービス制御ノードの開発における要求から形式仕様 (VDM++モデル) の策定まで行うべき手順や中間生産物を検討し，形式仕様策定法を提案する．この策定法を適用する対象として，簡単化したSBC(Session Border Controller) の要求を使って形式仕様策定の試行を行い，結果を報告する．

1. はじめに

インターネットはいくつかの発展段階を経て，豊かで高度なインタフェースと機能を活用しながら，効率的で快適なコミュニケーション環境を提供し，社会活動の高度化と効率化へ貢献する段階へと進化している．現在のインターネットの重要な機能の 1 つは，IP 電話である．IP 電話では，SIP(Session Initiation Protocol:セッション開始プロトコル)[1], [2] を使ってクライアント同士の直接通信を行い，IP ネットワークでさまざまなデータや音声・映像などのメディアの通信を行うことができる．インターネットの利用が社会全般に広がり，SIP プロトコルの運用には大量のデータ・トラフィックを処理しながら多様な通信網へ柔軟に対応できるインタネット機器が必要になっている．これらのインターネット機器上のソフトウェアの開発では，新規のサービスが日々開発されており，短期間で複雑な仕様決定が迫られ，開発の早期段階，つまり上流工程において厳密かつ網羅的な仕様策定が必要となる．

厳密かつ網羅的な仕様策定を行うために，形式モデルの導入が一般的である．形式モデルとは，形式仕様記述言語を用いて仕様を符号論理や集合論の数学的記述で作成したものとなる．形式モデル導入の効果として，対象システムの要求に対して，構成要素，操作内容などに対して矛盾，曖昧さ及び未定義部分の発見には効果的である．

本稿では，SIP 関連機器のソフトウェア開発において，上流工程での仕様策定における形式手法の導入方法を提案する．具体的には，形式仕様記述言語 VDM++を用いた形式仕様策定において，厳密かつ網羅的に行うことを支援するために，要求から形式モデル作成までの手順を提案する．形式モデルの作成手順について，一般的には抽象的な手順しか提示できないが，対象範囲によって制限を加えれば定式化・自動化が可能になると考えられる．本稿では，情報処理推進機構 (IPA) が公開した形式手法適用手順 (VDM++編)[3] 及び SIP 関連機器のソフトウェア開発慣習・状況を考案し、SIP 関連のソフトウェア開発を考慮してカスタマイズした形式仕様策定手法を提案する．提案手法を評価するには，事例として簡単化した SBC(Session Border Controller) の要求を VDM++モデル化し，考察を行う．

本稿の構成は以下の通りである．第 2 章では，基礎知識の SIP と SBC，および形式仕様記述言語 VDM++を紹介する．第 3 章では，IPA が公開した形式手法適用手順を説明する．第 4 章では，前提などの条件を説明したうえで本稿の提案手法を述べる．第 5 章では，簡単化した SBC の要求を用いて第 4 章で述べた提案手法を適用し，考察を行う．第 6 章では，関連研究として，形式モデリング方法論と SIP に関するモデリングについて議論し，提案手法の位置付けを述べる．最後に，第 7 章では，結論及び今後の発展について記す．

2. 基礎知識

本章は，SIP と SBC を簡単に紹介し，続いて形式仕様記

[1] 九州大学大学院システム情報科学府
Graduate Schoold of Information Science and Electrical Engineering, Kyushu University
[2] 九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University
[3] 富士通九州ネットワークテクノロジーズ株式会社
FUJITSU Kyushu Network Technologies Limited

© 2014 Information Processing Society of Japan

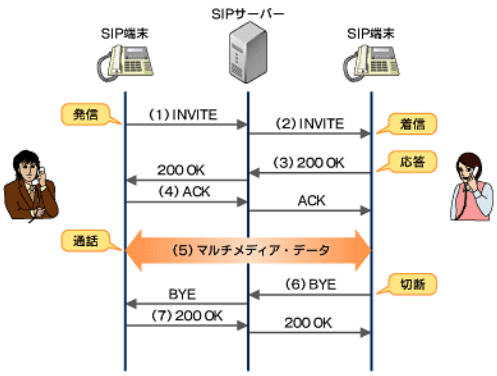

図 1 SIP の通信例

述手法の VDM++について説明する．

2.1 SIP と SBC

SIP(Session Initiation Protocol:セッション開始プロトコル) は，IP ネットワーク上の端末の間でデータのやりとりを行うセッションを管理するために，交信すべき標準手続きを規定している．現在使われている SIP は基本的に 2002 年に改版された RFC3261 である．SIP による通信は，HTTP をベースとするテキスト形式のメッセージで行うため，処理しやいという利点があり，よく使われている．SIP の手続きは図 1 の例で示されるように，２つの端末の一方（左側）から INVITE のリクエストを送信し、相手の端末（右側）が SIP の規定に従って応答をしながら通話フェーズに入って音声・映像などのマルチメディア通信を行う．通話フェーズが終わってからセッション切断までも SIP に従って応答を行う．この通信の開始から終了までの交信すべきリクエスト・レスポンスとその順番が SIP に規定されている．もう少し詳しく説明すると，図 1 の場合では，左の端末がクライアント側（User Agent Client, UAC）で INVITE のリクエストを出す；右の端末がサーバ側（User Agent Server, UAS）で INVITE リクエストを受ける；SIP サーバは２つの端末をそれぞれ見つけて通信の受け渡しを行い，SIP のセッションの制御・管理などはしない．

SIP の規定では，１つのセッションには複数の処理で構成されている．これらの処理をトランザクションという．図 1 の例では，INVITE メッセージによってセッションが開始され，BYE メッセージによってセッションが終了する．具体的には，INVITE リクエストと 200OK レスポンスの応答がセッション開始のトランザクションとなり，続いての BYE リクエストと 200OK レスポンスのトランザクションでセッションが終了となる．[*1]

実際の製品開発においては，図 1 のような２つの端末だけのシナリオではなく，同時に複数の端末の通信を協調しながらサービスを提供することが一般的である．このような場合，図 1 の SIP サーバが単なる転送を行うだけではなく，複数の端末を 1 対 1 や多対多で繋ぐなどのサービスを提供することになり，サーバのソフトウェアに複雑な処理が要求されている．それ故，実際の製品には SIP をベースにして自らのサービス及びプロトコルまでを作ることになり，SBC(Session Border Controller) はその代表例の 1 つである．SBC とは，SIP を利用する IP 電話ネットワークで、別のネットワークとの境界に設置されるゲートウェイ装置である．図 2 に示すように，SBC は端末間の通信，及びサーバ間の通信を協調する役割がある．IP 電話事業者間や、企業の自社 IP 電話網と通信事業者の IP 電話網の接続点などに設置され，IP アドレスの相互変換やポートの自動開閉によるセキュリティ確保などの機能を提供する．

[*1] マルチメディア・データのやりとりには，SIP で相互にやりとりされたデータの特性を適用した U-PLANE パケット（例：RTP など）によりセッションが作られるが，本稿では，C-PLANE（SIP）に着目するため，U-PLANE を対象外にする．

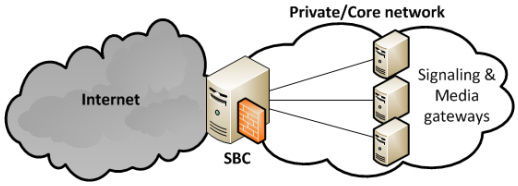

図 2 SBC の役割

2.2 VDM++形式仕様記述

VDM(Vienna Development Method)[4], [5] とは，IBM のウィーン研究所で 1960 年代から 70 年代にかけて開発された形式手法で，形式仕様記述手法の 1 つである．ソフトウェア開発の上流工程において対象システムの任意の抽象レベルにおける仕様を VDM モデルとして記述し，対象システムの機能が要求通りであるか正確性と完全性の観点での検証を行う．さらに，作成した VDM モデルは開発下流工程の活動ガイドとして利用できる．VDM モデリングを通して，開発早期に対象とするシステムに対する理解を明確にし，欠陥を発見して取り除くことでソフトウェアの品質を上げることが実証されている [6].

VDM で用いる仕様記述言語は本来プログラミングの仕様記述のために VDM-SL が提案され，さらにオブジェクト指向の概念などを拡張して VDM++ができた．VDM を使うには，The Overture Tool や VDMTools といったツールが提供されている．これらのツールを利用して，VDM モデルの編集（モデリング），検証（型検査，証明課題生成），アニメーション（インタプリタによる実行）ができる．VDM は実行可能な陽仕様として記述できるので，その特性を活かしてテストを用いて上流から下流まで開発工程の全活動を繋げることが可能である [7].

3. IPA 適用手順の紹介

IPA は、ソフトウェア開発の高信頼化に向けた有効な手

© 2014 Information Processing Society of Japan

法の一つとして形式手法の普及活動に取り組んできていて，2012 年 9 月に「形式手法活用ガイドならびに参考資料」を公開した．第 1 章で述べたように，本稿では上述公開資料の中の「形式手法適用手順（VDM++編）」を参考にしてカスタマイズした策定手法を提案し，事例の適用を行った．本節はまず「形式手法適用手順（VDM++編）」の手順（以下，IPA 適用手順）を説明する．

IPA 適用手順ではあるシステム業務説明書の分析から，VDM++モデル作成までの過程をいくつかの段階で分けて，各段階での作業と中間生産物を提示する．IPA 適用手順では，以下のステップがある：

(1) **VDM 要素抽出**：要求ドキュメント（システム化業務説明書など）から『VDM 要素一覧』を作成する．

(2) **仕様構造定義**：UML のクラス図記法を使って，仕様の階層構造を書く．仕様の構造とは，

- ユースケース記述階層
- ドメイン・オブジェクト記述階層
- 要求辞書記述階層
- ユーティリティ階層
- テスト階層

が含まれている．ここで注意してほしいのは，UML クラス図の記法を使うが，クラスやオブジェクトだけではなく，形式仕様の構造を全面的に表現することが目的である．

(3) **状態遷移定義**：状態を持つオブジェクトの状態遷移を記述する．

(4) **制約条件抽出**：システムに存在する制約条件を明らかにし，『VDM 要素一覧』中の用語と抽出した制約条件を結びつける．

(5) **形式記述の作成（陰仕様の定義）**：属性定義、クラスに型、インスタンス変数、定数、不変条件の追加する．

(6) **形式記述の作成（陽仕様の定義）**：前ステップ（陰仕様の定義）で記述した関数や操作の内部処理を記述する．

(7) **テストケース作成**：仕様を確認するためのテストを検討し，単体テスト支援ライブラリ VDMUnit を用いてテストするためのテストケースを作成する．

(8) **テスト実行**：仕様の確認・検証を行う．

上述 IPA 適用手順のステップは，実際に適用するときには必ず順番を守る必要はなく，前のステップに戻して関連の中間生産物を修正・見直しながら VDM++モデルを作成していくこともある．また，中間生産物の形や内容は対象システムに応じて調整・変更が可能である．

4. 提案手法

本研究の提案手法は，第 3 節で紹介した IPA 適用手順をベースにして，SIP 関連サービスの開発におけるドメイン知識や条件を考慮した手法である．IPA 適用手順でも，ドメインに応じたテーラリングの重要性が述べられている．まず，SIP 関連サービス，あるいは通信キャリア向けのサービスについて，以下の前提が考えられる：

- 要求，あるいは実現したいサービスの仕様説明は，ノード間のメッセージ通信**シーケンス図**で表現すること．
- サービスを提供するといっても SIP の規定に従うことが必須である．また，UAS と UAC に相当するクラス・オブジェクトが必要である．その上，UAS と UAC の動作が SIP の規定より決められる．
- 基本的にメッセージ通信のフォーマット（ヘッダなど）が SIP に従う：リクエスト・メッセージは INVITE, ACK，BYE などがある；レスポンス・メッセージは暫定応答（例 : 100 Trying），成功応答（例 : 200 OK），エラー応答（例 : 400 Bad Request）などがある．

上述前提を考慮して，IPA 適用手順をカスタマイズして提案する手法には下記の手順が含まれている：

(0) **要求説明の分析**：このステップでは要求の説明を読み取って開発側の解釈（不明な点などの確認を含めて）を作成する．解釈の作成過程において要求を明確する役割がある．また，メッセージ通信シーケンス図がない場合はシーケンス図の作成も含む．

(1) **VDM 要素抽出**：このステップでは，VDM++モデルに必要なクラス・オブジェクト，及びメッセージ通信に使うリクエストとレスポンス・メッセージを洗い出し，中間生産物の『VDM 要素一覧』を作成する．

(2) **仕様構造定義**：ここでの仕様構造にはユースケース階層，ドメイン・オブジェクト記述階層，およびテスト階層がある．

- **ユースケース階層**：ここでは，１つのユースケースを１つのトランザクションとして表現する．トランザクションとは，リクエスト・メッセージが受信され，処理などを経てレスポンス・メッセージが送信されるまでにする．
- **ドメイン・オブジェクト記述階層**：基本的なオブジェクトは開発対象システムのオブジェクトである．また，SIP 関連なので，SIP に規定されている UAS, UAC，及びリクエスト・メッセージ，レスポンス・メッセージといったオブジェクトも必要である．
- **テスト階層**：テスト階層では，テストケースのオブ

© 2014 Information Processing Society of Japan

ジェクトを記入する．テストケースの内容として，ユースケース階層の各トランザクションにおいて確認に必要なテストケースを列挙する．基本的には，ノードが 1 つのリクエストやレスポンス・メッセージを受信し，処理する結果（返すメッセージ）の確認を行うことが 1 つのテストケースである．

(3) **状態遷移定義**：このステップでは，IPA 適用手順と同じく状態を持つオブジェクトの状態遷移を記述する．基本的に，状態遷移の遷移とするイベントはメッセージの処理である．メッセージを処理しながらオブジェクトが状態変更していくことにする．

(4) **制約条件抽出**：このステップで行うことは，各メッセージの処理に関する事前事後条件を書くことである．また，状態遷移を見て各メッセージの受付できる状態も制約条件にする．

(5) **形式記述の作成（陰仕様の定義）**：このステップでは，前ステップまでの中間生産物をもって VDM++モデルの陰仕様を作成する．基本的には，クラスと関連する型定義，変数定義，操作の定義などがある．陰仕様は，状態変化させる操作に対して事前条件と事後条件のみを記述した仕様で，陽仕様と異なり，実行はできない．

(6) **テストケース作成**：このステップでは，仕様構造のテスト階層にあるテストケースの内容を作り，VDMunit で実行できる形で VDM++のモデルにする．テストケースには，正常の場合のケースと異常の場合のケースを含めて適宜に作成する．

(7) **形式記述の作成（陽仕様の定義）**：このステップでは，作成した陰仕様をさらに前ステップで作成したテストケース用いてテスト結果が合うような陽仕様を作成する．

(8) **モデル評価**：このステップは，仕様を評価するために，仕様構成などを含めてレビューを行う．また，テスト実行も行って最終確認を行う．このステップで使うテストケースは，前のステップで作成したテストケースだけではなく，評価用のためのテストケースも作成する．

上述の手順では，中間生産物も最終生産物の VDM++モデルも手順を繰り返しながら最終版に修正していくことになる．例えば仕様構造の中のオブジェクトが後のステップで足りないと判断する場合は，仕様構造検討に戻ってオブジェクトの追加・修正を行う．また，陽仕様を作成しながら仕様構造のオブジェクトの内容も合わせて修正しながらすることもあり得る．上述手順のフロー図は，図 3 で示されている．

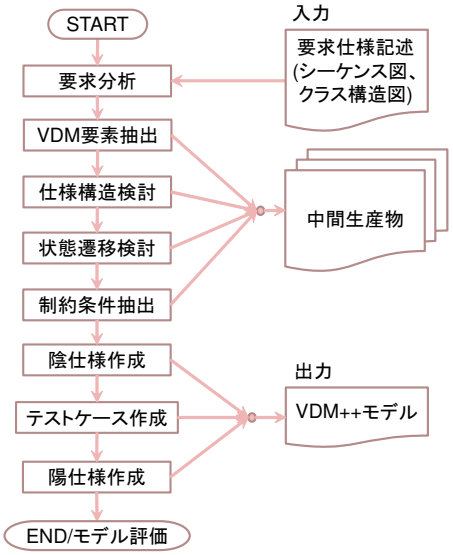

図 3 提案手法のフロー

5. 事例

本章では，第 4 章で述べた提案手法の手順に沿って，簡単な SBC の要求記述から形式仕様の VDM++モデル作成まで行い，評価・考察を行った結果を述べる．

5.1 要求記述

簡単化した SBC(Session Border Controller) の要求記述は以下の通りである：

- ネットワーク構成は図 4 に示されている．ここでは，SBC ノードに対して，Node A と Node B の通信を管理し，サービスを提供することを想定している．
- SBC ノードが行うサービスのシーケンス図は図 5 に示されている．Node A は基本的に SIP のクライアント側で INVITE リクエストを送信する．Node B はその反対のサーバ側で INVITE リクエストを受け取り，レスポンスを送信する．SBC の主な役割は、Ringring までの途中に，暫定応答の 100 と 183 を Node A に返答し，Node A からの PRACK メッセージを受け取り，セッションを先に確保する．その後，SBC が Node B に INVITE リクエストを転送し，Node B との応答を行いながら Node A と Node B の通話を Ringing，及び通話開始・終了段階まで協調する．
- 送受信される SIP メッセージには，メッセージ名，IP アドレス（送信元/送信先），および Cseq が含まれている [*2]．
- シーケンスの振る舞いに逸脱するようなメッセージ送

[*2] Cseq とは SIP メッセージの主要なヘッダの 1 つである．本稿では，ヘッダ情報に関する細かい設定の説明は省略する

© 2014 Information Processing Society of Japan

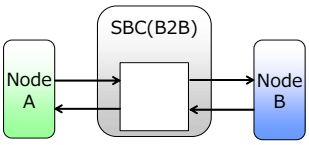

図 4　SBC のネットワーク構成

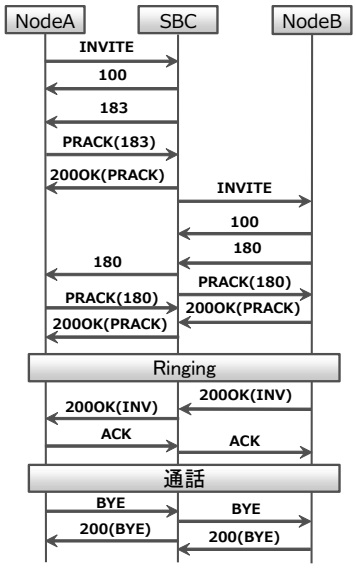

図 5　SBC のシーケンス図

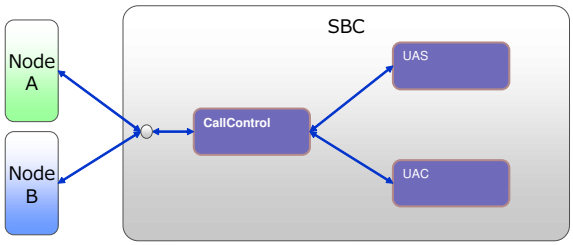

図 6　SBC のクラス構成

受信の場合はエラー検出になる．

- SBC ノードのクラス図は図 6 に示されている．基本構造は SIP に従うために，SBC は UAS，UAC が必須である．Node A と Node B とのやりとりを行うための窓口は Call Control である．

要求記述を分析して要求記述を明確にする過程はここでは省略するが，要求記述の文書のほか，図 4 のネットワーク構成，図 5 のメッセージ送受信シーケンス，及び図 6 のクラス構成を確認して必要情報の記述漏れかあるかどうかを最初的に確認する必要である．特に図 5 のメッセージ送受信シーケンスには，メッセージの順番や内容に漏れがある場合は下流工程でバグや不具合の原因となり，手戻りや追加コストの原因になる．

5.2　VDM 要素抽出

次は，第 5.1 節の要求記述から，VDM 要素の抽出を行って，以下の要素があげられた：

- アクター：NodeA，NodeB，SBC_CC，SBC_UAS，SBC_UAC．
- データ：IP アドレス（送信元，送信先），リクエスト/レスポンス・メッセージ，Cseq．
- ユースケース：INVITE トランザクション，ACK トランザクション，BYE トランザクション．
- 状態：SBC_CC, SBC_UAS, SBC_UAC の状態．

SBC_CC, SBC_UAS, SBC_UAC とは，図 6 のクラス構造の中にある CallController, UAS, UAC に対応する．SIP 関連のサービスの場合に限定すると，これらの要素は基本的に，アクター，データ，ユースケースと（オブジェクトの）状態に分類できる．また，具体的なリクエスト/レスポンスメッセージ名もシーケンス図の内容を見ながら列挙できた．

5.3　仕様構造定義

仕様構造を検討する段階では，第 4 節で述べたユースケース階層，ドメイン・オブジェクト階層，およびテスト階層の内容を決める．それぞれの階層の内容は以下の通りに決められる．

- ユースケース階層：この階層の内容は，『VDM 要素一覧』のユースケースに該当する．ここでは直接的に INVITE トランザクション，ACK トランザクション，BYE トランザクションとする．
- ドメイン・オブジェクト階層：この階層の内容は，『VDM 要素一覧』のアクターに該当するが，NodeA と NodeB はデータの IP アドレスで表現することがよいと判断し，SBC_CC, SBC_UAS, SBC_UAC の 3 つのオブジェクトとする．
- テスト階層：図 5 のシーケンス図を分解して 10 個のテストケース（正常系）を作った．

テスト階層のテストケースの作成について，VDMunit というユニット・テストのための VDM++の標準的なクラス/モジュールを利用するため，テスト階層のテストケースはユニット・テストの形にする．例えば，図 5 のシーケンス図の最初のメッセージは NodeA からの INVITE である．テストケースとして，「NodeA から INVITE を受信する」イベントを入力として，出力は次の 2 つのイベント：「NodeA に 100 を送信する」，「NodeA に 183 を送信する」となる．このように，入力と出力の対をシーケンス図から抽出してテストケースを作成した．

図 7，図 8，および図 9 はそれぞれ作成した仕様構造の

© 2014 Information Processing Society of Japan

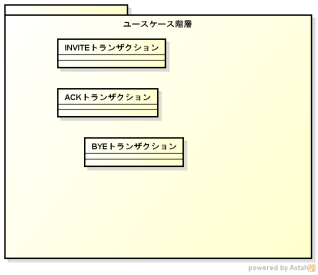

図 7 SBC 仕様構造：ユースケース階層

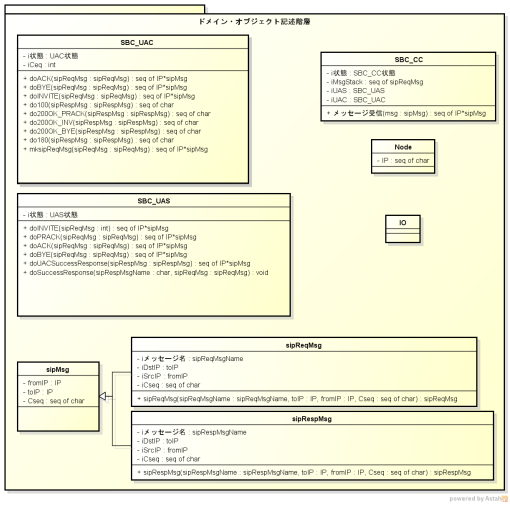

図 8 SBC 仕様構造：ドメイン・オブジェクト階層

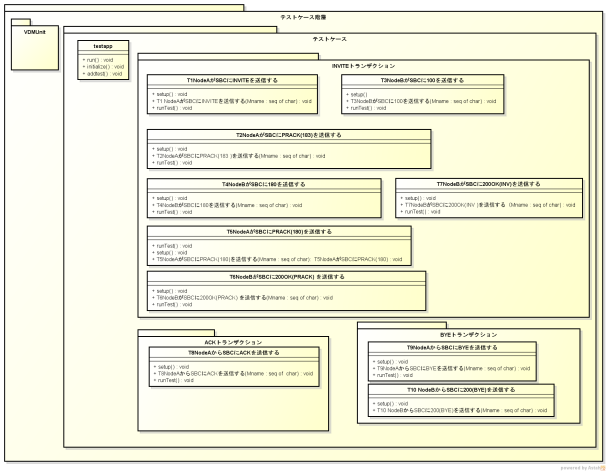

図 9 SBC 仕様構造：テスト階層

ユースケース階層，ドメイン・オブジェクト階層，およびテスト階層を表すクラス図である．

5.4 状態遷移定義

ここでは SBC_CC, SBC_UAS, SBC_UAC の 3 つのオブジェクトの状態遷移を検討することである．基本的には，図 5 のシーケンス図に従って，SBC ノードのメッセージ受信によって，SBC_CC, SBC_UAS, SBC_UAC の状態が処理によって変化していくことである．例えば，最初の

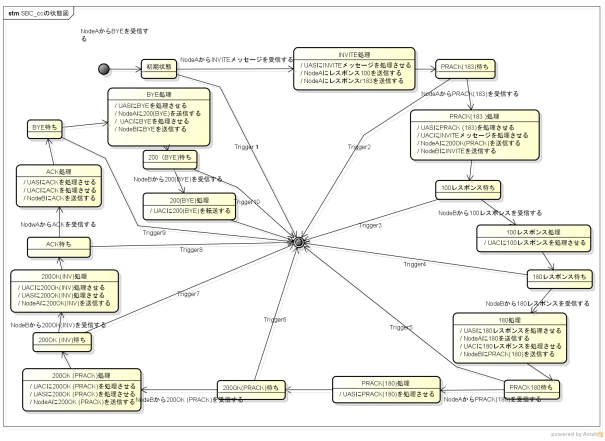

図 10 状態遷移：SBC_CC

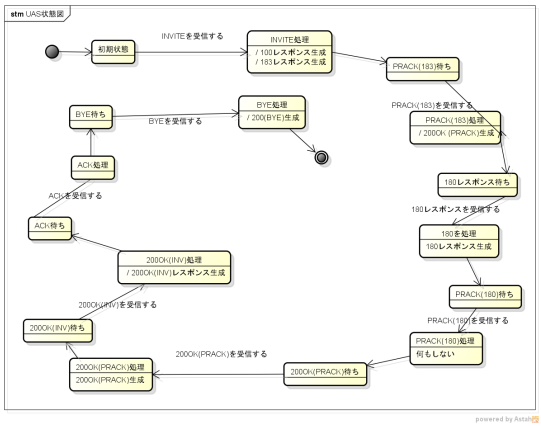

図 11 状態遷移：SBC_UAS

INVITE リクエスト・メッセージの受信するイベントによって，SBC_CC が「初期状態」から「INVITE 処理状態」に入る．「INVITE 処理状態」の処理内容は，SBC_UAS に INVITE を転送し，NodeA に返事するための 100 と 183 レスポンス・メッセージを作らせて返事を送信する．このような「メッセージ受信するイベント」と「状態の中の処理内容」をシーケンス図のメッセージ交信に沿って状態遷移を作成する．

図 10，図 11，および図 12 はそれぞれ作成した SBC_CC, SBC_UAS，及び SBC_UAC の状態遷移図である．

5.5 制約条件抽出

ここで述べる制約条件とは，メッセージ受信するときに，それぞれのメッセージ（INVITE など）を受信して処理するときの制約条件．ベースとなる制約条件は，5.4 で述べた状態遷移の状態である．また，各メッセージの処理内容に当たって，必要な制約条件があればさらに追加することもある．これらは最後に，中間生産物の『制約条件一覧』にまとめる．

© 2014 Information Processing Society of Japan

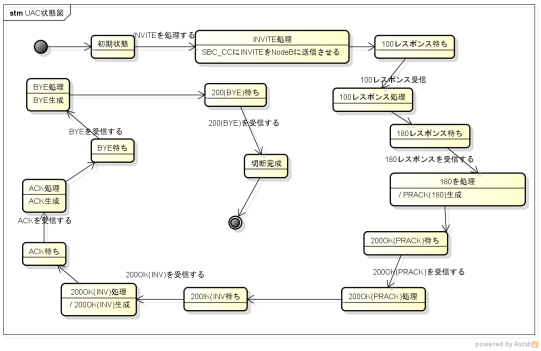

図 12 状態遷移：SBC_UAC

```
>> debug t.execute()
Start test - テストケース名
Start test - Case1 - NodeAがINVITEを送信する
End test  - Case1 - NodeAがINVITEを送信する
Start test - Case2 - NodeAがPRACK_183を送信する
End test  - Case2 - NodeAがPRACK_183を送信する
Start test - Case3 - NodeBが100を送信する
End test  - Case3 - NodeBが100を送信する
Start test - Case4 - NodeBが180を送信する
End test  - Case4 - NodeBが180を送信する
Start test - Case5 - NodeAがPRACK_180を送信する
End test  - Case5 - NodeAがPRACK_180を送信する
Start test - Case6 - NodeBが200OK_PRACKを送信する
End test  - Case6 - NodeBが200OK_PRACKを送信する
Start test - Case7 - NodeBが200OK_INVを送信する
End test  - Case7 - NodeBが200OK_INVを送信する
Start test - Case8 - NodeAがACKを送信する
End test  - Case8 - NodeAがACKを送信する
Start test - Case9 - NodeAがBYEを送信する
End test  - Case9 - NodeAがBYEを送信する
Start test - Case10 - NodeBが200_BYEを送信する
End test  - Case10 - NodeBが200_BYEを送信する
End test  - テストケース名
*** All Tests Passed. ***(no return value)
>>
```

図 13 陽仕様完成時のテストケース実行結果

5.6 形式記述の作成（陰仕様の定義）

陰仕様の作成に必要なのは，ここまでの中間生産物：『VDM 要素』,『仕様構造』,『状態遷移』，および『制約条件一覧』である．具体的には,『仕様構造』のドメイン・オブジェクト階層のオブジェクトごとにクラスを作り，必要な変数や操作のシグネチャを書くことである.『制約条件一覧』の制約条件は，主に VDM++クラスの操作の事前・事後条件となる．

5.7 テストケース作成

『仕様構造』のテスト階層ではテストケースを書いたが，ここでは，テストケースを具体的に VDMunit の形で整えて，実行可能にした．

5.8 形式記述の作成（陽仕様の定義）

このステップでは，前のステップで作った VDM++陰仕様から陽仕様を作成する．陽仕様の作成は，VDMunit テストケースにかけて期待値のメッセージを NodeA または NodeB が受信できるよう陰仕様から拡張する．図 13 では完成時のテストケース実行結果の画面（VDMTools）を示す．

5.9 モデル評価

最後のステップのモデル評価について，簡単化したシステムのため，評価用テストケース作成は行わなかったが，クラス構造，メッセージ・ヘッダなどのチェックリストを用意し，専門家を交えたレビューを行った．評価結果として，要求記述に満たす形式モデルの作成ができたと評価された．

5.10 考察

提案手法の事例として，簡単化した SBC の要求記述から形式仕様の VDM++モデルの作成を行い，作成した形式モデルが要求に満たすことと評価できた．この事例において，まず重要な特徴として，要求記述にはシーケンス図とクラス構造図が提示された．シーケンス図は，IP ネットワーク上の通信関連サービスに共通する振る舞いの表現だと考えられる．クラス構造図は，SIP を利用するサービスには必須な構造が含まれている．これらの図の提示がある前提で，更に要求分析及び形式モデル作成までの手続き及び中間生産物のフォーマットと内容の定式化ができると考えられる．提案手法を適用してみたところ，シーケンス図の分析・分解により，状態遷移，テストケース，及び基本的制約条件が自然に導出できることが分かったので，シーケンス図から自動的に中間生産物を作成する可能性が確認できた．また，シーケンス図で表現した振る舞いは，IP ネットワーク上の通信関連サービスにとって共通的だと考えられ，SIP 関連サービス以外の通信関連サービスの開発にも適用できると予想される．その上に，定式化，または自動化が可能なため，大規模開発への適用も期待できる．

この事例は，簡単化したノードのサービス要求とはいえ，形式モデル作成過程の中に，要求記述，及びとシーケンス図とクラス図の記述漏れ，矛盾などの発見があった．一般的形式手法を適用する効果ではあるが，早期的仕様の漏れや矛盾の発見ができたことが分かった．

テストケースを先に作成してテスト結果が合うまで VDM++陽仕様を完成することが本稿の提案手法の 1 つの特徴である．要求記述を系統的分析し，テストケースを作成することが現場では使われてない作法だが，今回の事例を通して，困難なくテストケースを作成し，テストしながら陽仕様を作成することができたため，現場での適用も考えられる．

1 つ注意すべき点として，今回の事例は簡単化した要求記述のため，単純なテストケースを通すだけで要求を満たすモデルが作れた．複雑な要求に対して，例えば，複数のシーケンス図の場合，単なるテストケースだけではなく，テストケースが表わすシステムの振る舞いまで考慮すべきである．この問題について，提案手法の関連ステップ，特に陽仕様作成の部分には，今後見直す必要がある．

6. 関連研究

6.1 形式仕様のモデリング方法論

形式手法の適用に関する研究がたくさんある．作成した

© 2014 Information Processing Society of Japan

形式モデルの良さ悪さで検証および設計・実装まで大きく影響するため，形式モデリングが最初に直面しなければならない重要な課題である．VDM のモデリング方法論について，昔から Fitzgerald と Larsen のモデリング手順がある [4], [5]．これはとても一般的なため，事例ごとに気を付けてモデリングを行うことになり，VDM 仕様記述を導入したい人々にとっては容易でない作業である．この問題の解決を考え，いくつか形式手法適用に関するテキストを IPA がリリースしてきたが，多くは事例ベースのため，事例関連のドメイン知識との関連が排除できない．モデリング手順が対象システムのドメインに依存するなら，対象システム・ドメインの場合分けをし，各々の場合のドメインに向けたモデリング手順を一般的な手順からカスタマイズすることが必要だと考え，本稿の形式モデル作成手法を提案した．

6.2 SIP の形式モデリング

SIP は，第 2.1 節で述べた RFC3261 が主なスペックだが，SIP を使うために，他の関連 RFC 及び SIP の部分修正・拡張した RFC の参照が必要で，RFC3261 だけでは全体的理解は難しい．全体的理解を誤れば，矛盾やバグなどが生じ，開発過程及び製品には大きな問題となる．この課題に対して，形式的に記述し，RFC の補助文書として理解を促進しながら検証目的にも役立つことの試みがあった [8]．本稿では SIP を対象にするモデリングではなく，SIP を使うサービスのモデリングだが，今後，上述研究成果の形式モデルを取り入れることにより，開発対象の通信サービス・ノードに対して SIP との矛盾や違反検出することも考えられる．このような発展ができれば，単なる形式モデルの作成ではなく，形式モデルによる検証にも繋がる．

7. おわりに

本稿では，通信キャリア向けサービス制御ノードに対して，形式仕様の VDM++モデルの作成手順を提案し，提案手法の手順を用いて簡単化した SBC の要求記述から形式モデルの VDM++モデル（陽仕様）を作成し，結果の考察を行った．本稿の提案手順は，SIP 関連の通信サービスのドメインを考慮し，シーケンス図とクラス構造図が提示されている特徴を前提にしている．この前提が満足されるならば，提案手法の定式化または自動化が可能となり，大規模開発への対応が期待できる．事例の仕様策定結果として，要求に満たす形式モデルができたと評価したが，要求が複雑な場合の対応について提案手順の改善および詳細ステップの導入が今後の課題となる．また，簡単化した SBC 要求記述の事例を通して，形式手法の適用において一般的効果ではあるが，要求の記述漏れや矛盾などの誤りも発見できた．

今後の課題として，上述適用手順の見直しと詳細ステップ導入のほか，今回の事例より複雑な事例に適用することを考えている．また，第 5.10 節の考察で述べたように，SIP との矛盾を検証することも考えている．

参考文献

[1] SIP:session initiation protocol, http://www.rfc-base.org/rfc-3261.html.
[2] 千村 保文, 坂口 克彦. **次世代 *SIP* 教科書**. インプレス標準教科書シリーズ. インプレス R&D, 2010.
[3] 情報処理推進機構. 形式手法活用ガイドならびに参考資料, http://www.ipa.go.jp/sec/softwareengineering/reports/20120928.html.
[4] John Fitzgerald, Peter Gorm Larsen, 荒木 啓二郎（訳）, 張 漢明（訳）, 荻野 隆彦（訳）, 佐原 伸（訳）, 染谷 誠（訳）. **ソフトウェア開発のモデル化技法**. 岩波書店, 2003.
[5] John Fitzgerald, Peter Gorm Larsen, Paul Mukherjee, Nico Plat, Marcel Verhoef, 酒匂 寛（訳）. ***VDM++*によるオブジェクト指向システムの高品質設計と検証 : 仕様の品質を飛躍的に高める手法**. 翔泳社, 2010.
[6] 中津川 泰正, 栗田 太郎, 荒木 啓二郎. 実行可能性と可読性を考慮した形式仕様記述スタイル. **コンピュータ　ソフトウェア**, 27(2):pp 130–135, 2010.
[7] 中津川 泰正, 栗田 太郎, 荒木 啓二郎. 形式仕様記述手法を用いた FeliCa カード開発におけるテスト実施効率の考察. **ソフトウェア技術者協会, ソフトウェアシンポジウム *2013* 論文集**.
[8] Pamela Zave. Understanding SIP through model-checking. *Principles, Systems and Applications of IP Telecommunications. Services and Security for Next Generation Networks*, volume 5310 of *Lecture Notes in Computer Science*, pp 256–279. Springer-Verlag Berlin Heidelberg, 2008.

© 2014 Information Processing Society of Japan