B-004

VDM++ 分散モデル実行環境を用いたプロトタイピングと非同期 UI への適用

Application to the Prototyping and its Asynchronous UI using Distributed VDM++ Model Execution Environment

多田 圭佑 †　　和崎 克己 ††
Keisuke Tada　Katsumi Wasaki

1　はじめに

システムの仕様は，自然言語や表・図などを用いて記述されることが主流である．しかし，仕様自体の曖昧さや人的エラーの混入，それらに起因する欠陥の発見遅れ，手戻りなど様々な問題がある．形式的な言語を用いてシステムモデルを構築し，仕様や設計の明確化・厳密化を行い，モデルを分析・検証することでシステムの高品質化が期待できる．VDM++ は，厳密なシステムモデルが構築できる形式仕様記述言語の一つである．VDM++ モデルの妥当性は，援用ツールのモデル実行機能を用い，テスト・プロトタイピングを行うことで確認できる．しかし，援用ツールを直接操作する際には専門の知識が必要であり，実際のシステム利用者とって相応しいものではない．

著者らは VDM++ 向けの分散モデル実行環境 [1] を開発している．Java 向けに提供している Web API を用いることで，実行環境上にある VDM++ モデルをインタープリター実行できる．この API を組み込んだクライアントアプリケーションを GUI で構築することで，モデル実行が直感的に実施可能となる．また，モデル実行により得た結果をわかりやすい形で伝えられるため，妥当性確認が容易になる [2]．しかし，問題点としてクライアントアプリケーションを VDM++ モデルと一対一で構築する必要があり，妥当性確認におけるコストの増大が課題であった．

本研究では，UML アクティビティ図を用いた上位設計からスケルトンコードを自動生成する手法を導入し，クライアントアプリケーションの実装を Java Servlet コードとして自動生成し，プロトタイピングにかかるコストの削減を試みた．本報告では，Ajax 向け JavaScript コード，データベース通信コード生成のためにコード生成器を拡張した．最後に，非同期 UI を有する予約システムのプロトタイピングを行い，今回追加した生成機能を評価した．

2　形式仕様記述言語 VDM++

2.1　形式手法と形式仕様記述言語 VDM++

形式手法とは，信頼性の高いシステムを作るために用いる仕様記述・設計開発・検証の技術である．形式手法の一種として形式仕様記述が存在する．形式仕様記述は理論に基づいた形で仕様を厳密，正確に書く．基本設計の段階でモデル実行を行い誤りを排除することで，手戻りによるコスト増大を防止できる（図 1）．形式仕様記述言語 VDM++ は，Vienna Development Method（ウィーン開発手法）の基盤言語 VDM-SL を，オブジェクト指向に基づいたモデル化を扱えるように文法を拡張したものである [3]．VDM++ には，不変条件，事前条件，事後条件を制約条件として記述可能な支援ツールが用意されており，これによって仕様を検査，テストすることが出来る．VDM++ には開発援用ツールが存在する．厳密さでは数学的証明を用いた定理証明系に劣るものの，専門的知識がなくとも仕様実行を用いて妥当性を確認出来る．

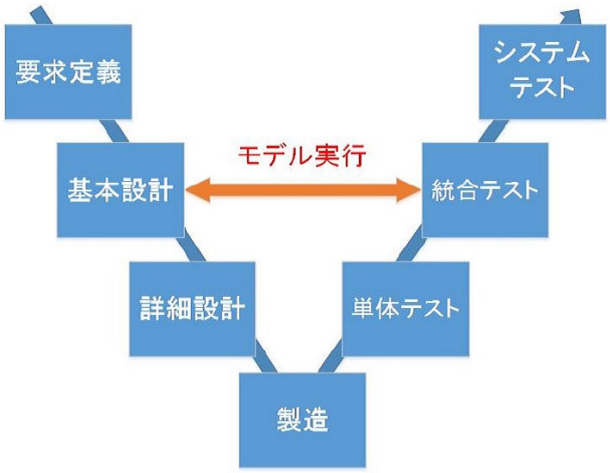

図 1　上流工程におけるモデル実行と統合テストの位置

2.2　VDMJ

VDMJ は Java で記述された仕様記述言語 VDMSL，VDM++，VDM-RT（VDM++ の非同期リアルタイム向けシミュレーションモデル用拡張）のための VDM 開発援用ツールである [4]．構文解析器，型チェッカー，インタプリタ，デバッガ，および証明課題生成機能を持つ．Overture という VDM 向け Eclipse プラグインに用いられている．オープンソースであり，自由な改変が可能である．

3　モデル実行環境を用いた妥当性確認

3.1　上流工程における妥当性確認の全体フロー

本研究で対象とする上流工程における妥当性確認の全体フローを図 2 に示す．モデル実行の際，クライアントアプリケーションを GUI で構築することで，妥当性確認は容易になるが，妥当性確認のためのコード記述コストが増大するという欠点が存在する．そこでモデルを構築する際に作成するクラス図やアクティビティ図から，

† 信州大学大学院理工学系研究科，Graduate School of Science and Technology，Shinshu University.
†† 信州大学工学部，Faculty of Engineering，Shinshu University.

著者らが開発している Java Servlet スケルトンコード生成器 [5] を用い，Java Servlet スケルトンコードを自動生成し記述コストを削減する．追記が行われたコードをサーバに配置し，Web API を用いて VDMJWeb サービス（後述）と通信を行う．

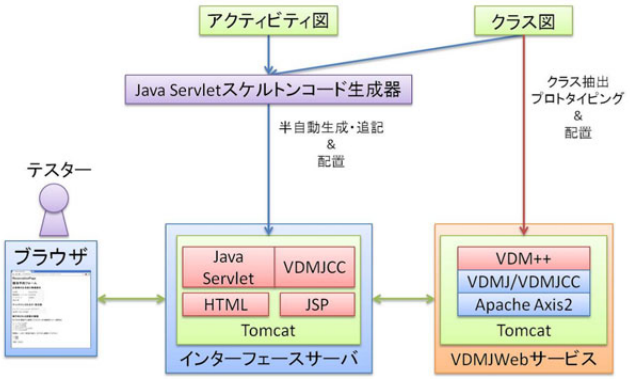

図 2　上流工程における妥当性確認の全体フロー

3.2　VDMJWeb サービス

VDMJWeb サービス [1] は，実際のインタープリタ機能を有する VDMJ，VDMJ のクライアントとして外部と通信を行うための機能が備わっている VDMJC から構成されている（図 3）．VDMJ のクライアントである VDMJC は Apache Axis2 を用いて Web サービスとして公開される．Web コンテナとして Apache Tomcat を用いている．Web サービスの機能を利用するため提供されている Web API（VDMJCC）を用いて，クライアントアプリケーションは VDMJWeb サービスと SOAP 通信を行うことが出来る．

Web ブラウザから送られてくるリクエストは Tomcat 上の Java Servlet で受け，Web API を介して VDMJ サービスへ送られ処理される．VDMJ のインタープリタによる処理結果は，Web API(VDMJCC) を用いた Java Servlet により Web ブラウザに表示される．その結果を見てユーザは妥当性確認を行う．ユーザが行った妥当性確認の結果を踏まえ，VDM++ を記述する開発者は再度 VDM++ を記述する．

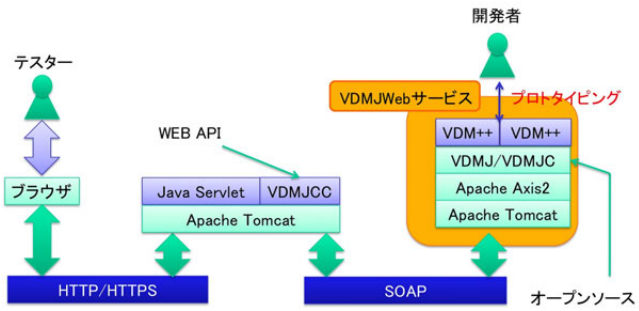

図 3　VDMJWeb サービスの構成

4　上位設計からの自動生成

4.1　UML アクティビティ図

上位設計には UML アクティビティ図を利用し，システムの画面遷移及び処理フローを記述する．UML アクティビティ図は，UML2.0 で独立して定義された振る舞い図の一つで，多様な処理の流れや条件分岐などを表現できる．アクティビティ図では，モデリング対象が行う個々の活動をアクションノードとして記述し，アクション間の順序関係を制御フローで連結する．条件に応じてフローが分岐する場合は，ディシジョンノードを使う．ディシジョンノードの各分岐には，ガードと呼ばれる要素に分岐条件を記述する．その他にも外部イベントの発生を待機するイベント受信アクション，メッセージやシグナル送信を表現するシグナル送信アクションなどがある．

4.2　アクティビティ図を用いた上位設計

アクティビティ図からコード自動生成を行うための形式記述について提案した．まず，UI（ユーザインターフェース）・Java Servlet・VDM++ の縦パーティションをそれぞれ配置する．次に，入力フォームのイベント毎に 1 つの Java Servlet が対応するよう横パーティションを配置する．パーティションを配置後，各パーティションに対応したアクション，処理フローを記述する．

(a) UI パーティションにはユーザのシステムに対する入力イベントアクション，画面出力アクション，画面の書き換えアクション，Java Servlet との Ajax 通信アクションを記述する．入力イベントアクションにはイベント受信ノードを用いる．画面出力アクションには画面のフォーム情報，フォーム情報を受け取る変数を記述したノートノードを紐づけする．画面の書き換えアクションや Java Servlet との Ajax 通信アクションは，クリックなどのユーザによる入力イベントアクションの後に記述する．非同期動作 (複数サーバとの通信など) を表現したい場合はフォークノードを用いて分岐させる．

(b) Java Servlet パーティションには UI との Ajax 通信アクション，入力されたデータに対する処理やモデル実行による条件分岐など，Java Servlet が行う処理を記述する．データベース通信を表現する場合は必要事項を定義部に記述する．

(c) VDM++ パーティションにはモデル実行アクションを記述する．妥当性確認を行いたいモデルはあらかじめクラス図で記述しておき，モデル実行アクションとリンクさせる．

UI と Java Servlet 間の Ajax 通信を表現する場合はシグナル送信ノード，イベント受信ノードを用い，必要事項を定義部に記述する．UI パーティションのシグナル送信ノードとイベント受信ノード，Java Servlet パーティションのイベント受信ノードとシグナル送信ノードはそれぞれリンクさせる．

4.3　コード自動生成

Java Servlet のスケルトンコードは，生成器に記述したアクティビティ図を通すことで得られる．UI パーティションのアクションからは直接 Java Servlet コード生成はされないが，ノートノードの記述を元にフォームからのデータを格納する変数が生成される．Java Servlet パーティションのアクションはコメントとして，条件分岐等の要素は Java の構文に対応して生成される．UI との Ajax 通信を行うコードは，イベント受信ノード

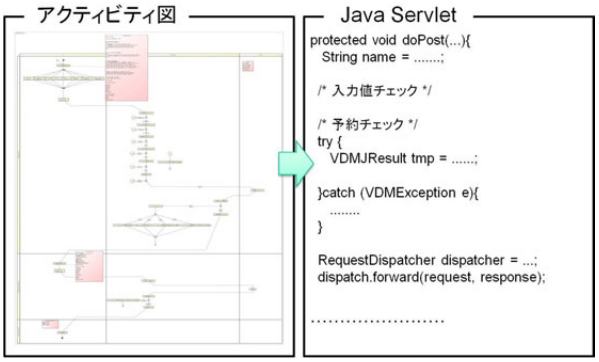

図 4　Java Servlet コード生成例

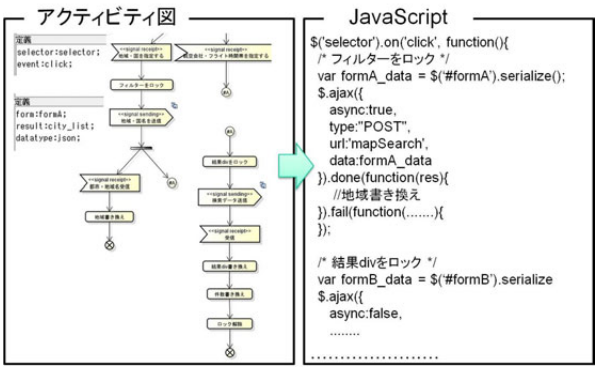

図 5　JavaScript コード生成例

からシグナル送信ノードまでのフローを解析して生成される．UI に返すデータは，JSON オブジェクトを想定している．VDM++ パーティションのアクションからは，モデル実行環境へ処理を委託するコードが生成される．画面遷移のコードは，Java Servlet パーティションから UI パーティションへのフローを元に生成される．データベースとの通信を行うコードは定義部に記述された必要事項を元に生成される．Ajax 向け JavaScript コードは，UI パーティションの入力アクション後の処理フローを解析して生成される．シグナル送信ノード，イベント受信ノード，定義部に記述された必要事項を元に Java Servlet との Ajax 通信コードが生成される．フォークノードを用いて非同期化している場合は非同期通信コード，そうでない場合は同期通信コードが生成される．画面書き換えアクションやその他のアクションはコメントとして生成される．Java Servlet との Ajax 通信には JQuery を用いて簡素化した．

5　モデル実行ケーススタディ

5.1　海外ツアー予約システム

生成器の評価を行うため，非同期 UI を有するシステムとして海外ツアー予約システムを作成した．一般的な予約システムは，予約，予約変更，予約取り消し等様々な機能があるが，ここでは「予約」のみに注目してプロトタイピングを行う．ツアー検索画面において非同期 UI を適用している．フィルター部分のいずれかの UI コンポーネントをクリックすると，Java Servlet との Ajax 通信と同時に検索結果の書き換え，フィルター部分の書き換えが行われる．

以下にシステムの想定される利用手順，画面遷移を示す．

システムフロー

1. システムはツアー検索画面を表示する
2. ユーザは条件を入力する
3. システムは検索結果を表示する
4. ユーザはシステム検索結果をクリックする
5. システムは予約画面を表示する
6. ユーザは必要事項（人数，パスポート等）を入力し，次へボタンを押す
7. システムは入力事項不備，予約可能性チェック等を行う
8. システムは予約確認画面を表示する
9. ユーザはクレジットカード情報を入力し，確認ボタンを押す
10. システムは予約を作成し，完了画面を表示する

今回作成したツアー予約システムのサブシステムとしてホテル予約システム，フライト予約システムを利用する．

ホテル予約システム

- システムは，入力情報をもとに予約可能かどうかチェックする
- システムは，予約可能である場合ユーザに最終確認を行う
- ユーザは予約内容を確認後，予約を確定する
- 宿泊プランは 5 つある．シングル，ツイン，ダブル，デラックスツイン，そしてスイートである
- ホテルの予約は予約日から 3 日前とする
- 日帰り (0 泊) はできない
- 宿泊人数はシングルは 1 名，ツイン，ダブル，デラックスツインは 2 名，スイートは 4 名まで予約可能である
- 禁煙ルームと喫煙可能ルームが選択できる
- チェックイン時に顧客を判別できるようシステムは予約番号を発行する

フライト予約システム

- 予約が満杯の便は予約不可
- 同じ日の同じフライトに同一人物が重複して予約不可
- 1 度の予約できるランクは 1 種類
- フライトステータスが保留，キャンセルのフライトには予約不可
- 有効期間外のフライトには予約不可
- 便が決まると使うリソースなど（ランク，席数，期間，ステータス，行き先）が決まる

システムの利用手順，画面遷移を元に作成したアクティビティ図の一部を図 6 に示す．この図は，非同期 UI とサーバ側との通信部分である．

5.2　結果

アクティビティ図から生成した Java Servlet コード，JavaScript コードに追記を行い，モデル実行を行った．図 7 にクライアントアプリケーションを用いた妥当性確

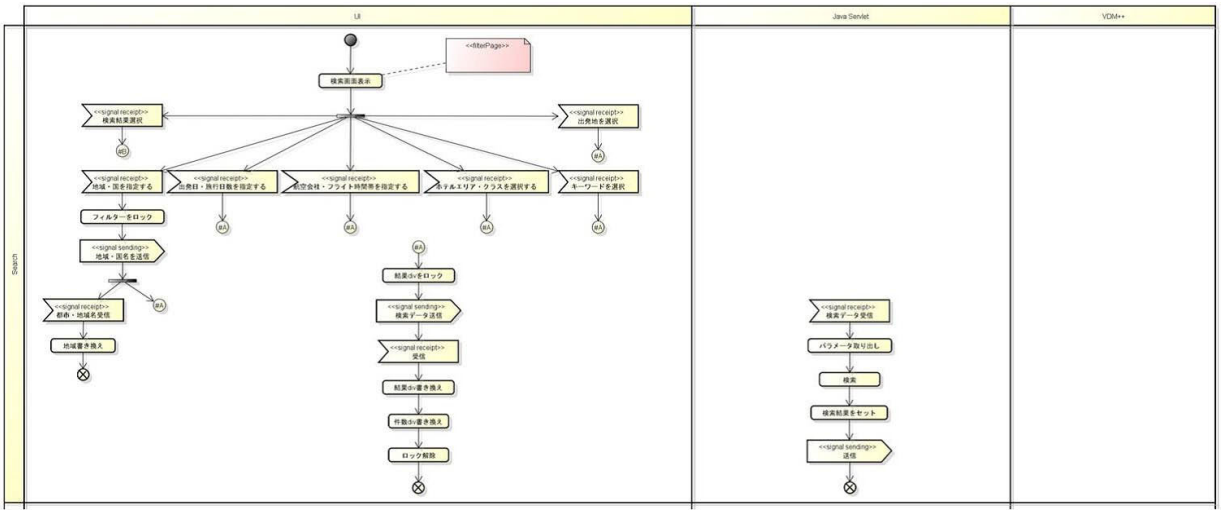

図 6　予約システムのアクティビティ図（抜粋）

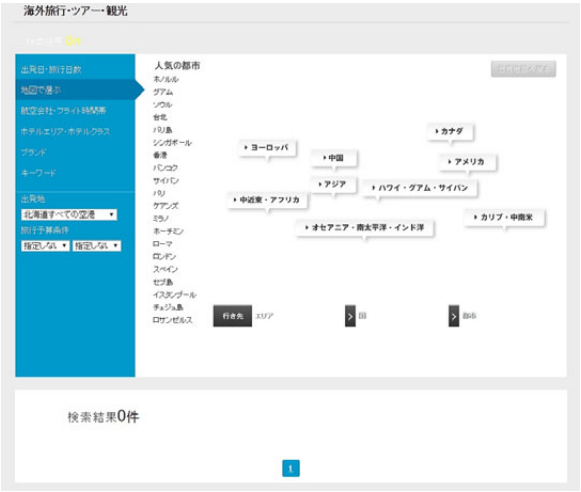

図 7　ツアー検索画面例

認の例を示す．生成された JavaScript コードを用いることで，非同期 UI を有するアプリケーションがプロトタイピングできた．クライアントアプリケーションを用いることで，従来通り VDM++ モデルの妥当性が容易かつ直感的に確認できた．

5.3　VDM++ モデル実行器をクラスタ並列化した構成

非同期 UI からサーバへの通信リクエスト多発により CPU，メモリーのリソース両面でモデル実行環境への負荷が増大する可能性がある．VDMJWeb サービスは SOAP ベースの分散環境でありスケールアウトが容易なため，クラスタ並列化することでマシンにかかる負荷の軽減が期待できる（図 8）．インターネット上に大規模分散させることも可能だが，セキュリティやロードバランシングなどの課題があるためイントラネット内を想定している．

6　まとめと今後の課題

Ajax 向け JavaScript コード，データベース通信コードが生成できるよう生成器を拡張した．ホテル予約システムとフライト予約システムの VDM++ モデル，拡張した生成器から生成したクライアントアプリケーション

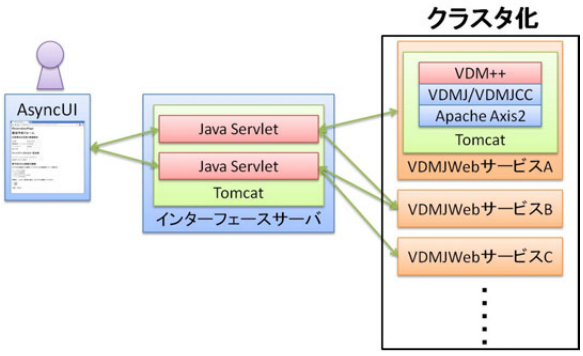

図 8　VDM++ モデル実行器をクラスタ並列化した構成案

を用いて，プロトタイピングを行い今回追加した生成機能を評価した．

現在 1 枚のアクティビティ図につき 1 システムのみの対応となっている．複数のアクティビティ図からコード生成ができるよう生成器を拡張し，ログイン処理など汎用的な部分は分割する等アクティビティ図の再利用性向上に努めたい．また，アクティビティ図の記述に制約をかけてはいるが，自由に記述できる部分も多いためモデルとして正しい振る舞いをするか検証する必要ある．アクティビティ図を Promela コードに変換し，SPIN を利用してアクティビティ図としての振る舞いを検証できるようにしたい．他方，人力によるテストに加えて，テスト自動化のための VDM++ コード，アクティビティ図フローからのメタ情報抽出法についても今後の課題となる．

参考文献

[1] 村林 慧，多田 圭佑，和崎 克己：VDMJ と Apache Axis2 を用いた上流工程におけるモデル実行環境の構築，第 13 回情報科学技術フォーラム，（7B-1），2014
[2] ジョン・フィッツジェラルド，他：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，翔泳社，2010.
[3] 石川冬樹：VDM++ による形式仕様記述，近代科学社，2011.
[4] VDMJ–Free Development software downloads at SourceForge.net，http://sourceforge.net/projects/vdmj/
[5] 多田圭佑，和崎克己：VDM++ を用いたラピッドプロトタイピング向けの Java スケルトンコード生成器；平成 26 年度電気関係学会東海支部連合大会講演集，（P1-1），2014