VDM-SL 機能仕様のアニメーション化支援に関する考察

2011SE109 金森悠眞
指導教員：張漢明

1 はじめに

ソフトウェア開発において仕様は, 正当性検証および妥当性確認の基準として重要な役割がある. 正当性検証とは, 仕様と設計されたプログラムが一致しているか否かを検証することである. 妥当性確認とは仕様と, 顧客のニーズが一致しているか否かを確認することである [1]. 開発では, 安全性や信頼性が高い開発を行うために, 形式手法は有効である. 形式手法とは, 数学に基づいたソフトウェア開発技術の総称であり, 形式仕様は, 厳密な記述による曖昧さの排除が可能である. 形式仕様は, 開発者間の開発対象の概念の共有にも有効であり, 開発者間の概念の相違を防ぐことが可能となるが, 仕様に形式手法を用いることで, 顧客が形式仕様を理解することが困難であると考える.

本研究の目的は, 仕様の妥当性確認のための VDM-SL 機能仕様のアニメーション作成手法を提示することである. アニメーション作成を行う際, 関数スタイルで記述された VDM-SL 機能仕様を再利用し, アニメーションができるまでの手順を提示する.

2 背景技術

VDM-SL の概要と関数スタイルの VDM-SL 機能仕様について述べる.

2.1 VDM-SL

1996 年にその構文と形式仕様意味論が ISO で標準化されている. また,VDM-SL は, モデル規範型の形式仕様記述言語で, 記述対象を抽象するための型, 定数, 関数, を定義し, 状態空間の定義と状態を変更する操作を定義することで, システムの入出力や状態変化を記述する.

2.2 関数スタイルの VDM-SL 機能仕様

関数スタイルとは, システムの状態の変化を関数を用いて表現することである. システム状態の変化と入出力を結びつける操作のモデリングを図 1 に示す [2].

図 1 状態の変化をモデリングした図

システム状態の変化と入出力を結びつける操作を関数を用いてた操作定義のひな形を次のように示す.

```
Functions
    操作名：入力型 * 状態出力型 -> 状態出力型
    操作名（入力, 前状態） == 後状態の定義

状態出力型 = 状態型 * 出力型
```

関数を用いて操作を記述するために, 引数で操作前の状態をもらい, 操作後の状態を返すことで定義している.

3 研究のアプローチ

本研究の目的に対するアプローチは, アニメーション作成ツールとして Lively Walk-Through を使用する. アニメーション化に必要な VDM-SL 機能仕様を分析して自動化生成を検討し, アニメーションが作成できるまでの手順の提示を行う.

アニメーション化に必要な VDM-SL 機能仕様の分析とは,Lively Walk-Through に VDM-SL 機能仕様を記述しても, 実行ができないため,VDM-SL 機能仕様に状態の定義が必要となる. この状態の定義を行うために, VDM-SL 機能仕様の分析が必要となる.

4 アニメーション作成手順

本研究の目的となるアニメーションを作成する手順を次のように示す.

- Widgets を作成
- VDM で状態の定義
- LiveTalk でイベントとアクションの対応付け

4.1 Lively Walk-Through の概要

本研究では,VDM-SL 機能仕様のアニメーション化する際に, アニメーションツールである Lively Walk-through を使用する. このツールは,VDM-SL 機能仕様を Lively Walk-Throug で実際に動作させることが可能である. 動作させるまでには, UI の部品となる Widgets を作成する必要がある.

4.2 Widgets の作成

Lively Walk-Through を使用し,UI の部品となる Widgets を選択し, 作成する. UI の部品となる Widgets の選択は,VDM-SL 機能仕様を分析して行う. UI の部品となる Widgets を選択する時には,Control と View がある.

Control は,Field と Button の Widgets に分かれる. Field の Widgets は, 値の入力と出力が可能な機能を持つ. Button の Widgets は,Button を 押下することで操作が可能な機能を持つ.

View は,Image と VariableImage の Widgets に分かれる. Image の Widgets は, 画像ファイルの出力が可能 な機能を持つ. VariableImage の Widgets は, 保持する値に応じて表示する画像をあらかじめ定義できる機能を持つ.

4.3 VDM-SL 機能仕様に状態を定義

Lively Walk-Through を使用して,VDM-SL 機能仕様をアニメーション化を行うには, VDM-SL 機能仕様の状態を定義する必要がある.

VDM-SL 機能仕様に状態を定義するには, VDM-SL 機能仕様は全体の状態を型として定義されているため, この型を状態で定義したもの次のように記述する.

```
State  識別子  of
    変数名 : 型
Init      識別子 == 式 (初期値)
End
```

4.4 イベントとアクションの対応付け

View に状態の表示を行うために, 次のように記述をする.

```
Operations
      状態の変数名 : ( ) ==> 型
      状態の変数名 == return 変数名 . タグ名  ;

      操作名 : 型  ==>( )
      操作名 (引数) ==
      状態識別子 := 関数名 (引数,  状態識別子)
```

また, イベントが発生した時, イベントに対してのアクションの対応付けを定義する必要がある. イベントは,Widgets の Button を押下した時に発生する. アクションは Widgets の Field や VariableImage に出力される. Button を押下時の Field と VariableImage に出力する場合の記述を次のように示す.

- Field に値を出力する記述

```
Button の名前  ' clicked
      操作名 ( )
      状態の変数名 ( )  ->  [Field の名前]
```

- VariableImage に画像を表示する記述

```
Button の名前  ' clicked
      操作名 ( )
      状態変の数名 ( )  ->  [VariableImage の名前]
```

5 考察

本研究では,VDM-SL 機能仕様のアニメーション作成手法の提示と作成手順を作成した. 作成手法を提示することにより,,VDM-SL 機能仕様のアニメーション作成が容易になると考えた. 形式仕様を顧客が理解できず, 妥当性の確認が困難であると考えていたが, 形式仕様を顧客が理解できなくても容易に妥当性の確認が可能になると考えた.

5.1 自動販売機の事例

事例として自動販売機のアニメーションを作成した. 商品を購入する時の硬貨投入をイベントとして考え,Widgets の 10 円 Button を押下する時に必要な手順を示す.

最初に Widgets の作成として,10 円 Button と預かり金を出力する Field を作成する. 次に状態の定義として, VDM-SL 機能仕様に状態を定義するには次のように示す.

```
State   St   of
    vm : 自動販売機型
Init  s == s  = mk_St(自働販売機_0)
End

Operations
    預かり金 : ( )  ==>  int
    預かり金 ( )  ==  return  vm . 預かり金 ;

    貨幣投入操作 : 金額型 ==> ( )
    貨幣投入操作 (投入金額) ==
    vm := 貨幣投入 (投入金額, vm);
```

手順の最後に, イベントとアクションの対応付けを行う. Widgets の 10 円 Button を押下する場合の対応付けと図 2 に自動販売機のアニメーションの UI を次のように示す.

```
10 円 ' clicked
      貨幣投入操作 (10)
      預かり金 ( ) -> [預かり金]
```

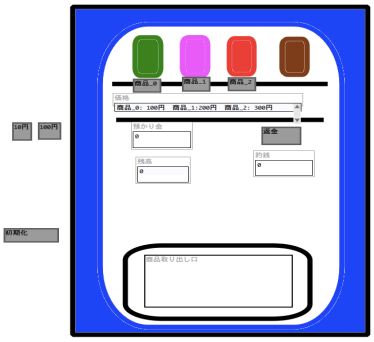

図 2 自動販売機の UI

5.2 自動化生成

アニメーション作成の手順の中で, 状態の定義とイベントとアクションの対応付けの部分が自動化生成できると考えられる. また, 状態の定義は,VDM-SL 機能仕様のどの型を定義するのかを決める必要があるが, 状態定義のひな形により自動化生成が可能であると考えられる.

6 おわりに

自動化生成を実現するには, イベントとアクションの対応付けと状態を対応表にする必要があると考えられる. この対応表により,VDM-SL 機能仕様から必要な情報を取得し, 対応表に当てはめることができれば, この自動生成は実現すると考えられる.

参考文献

[1] 玉井哲雄, ソフトウェア工学の基礎 , 岩波書店,2005.

[2] 荒木啓二郎, 張漢明, プログラム仕様記述論, オーム社,2002.