ISS-P-94

# 関数型言語による代数仕様からのプログラム実装法に関する研究

寺嶋　誠† 　　中村　正樹†
†富山県立大学工学部情報システム工学科

## 1. はじめに

形式手法とは、数学を基盤として、システムやソフトウェアのモデル化、正確な仕様記述、仕様の検証などを行う手法である[1]。形式手法によって検証された仕様をもとにプログラムを作成することで、信頼性の向上に期待できる。

本研究では、代数仕様言語 CafeOBJ と関数型言語 Haskell を用いて、代数仕様から関数型言語のプログラムを実装する方法について検討する。

## 2. CafeOBJ による振舞仕様の記述

本研究では、CafeOBJ 仕様を記述する方法として振舞仕様を扱う。振舞仕様では、システムをブラックボックスと見なし、外部から観測可能な値がどのように変化するかに着目することでシステムの状態を識別する。このとき、システムの状態を変化させる演算を遷移演算、システムの状態を観測する演算を観測演算という。

CafeOBJ 仕様は、複数のモジュールを組み合わせてできている。本研究では、モジュールを分類したうえでそれぞれの実装法を検討し、それらを合わせて振舞仕様全体の実装法とする。

まず、モジュールをデータ型モジュールとシステムモジュールの2つに分ける。データ型モジュールはシステムが用いるデータを定義するモジュールであり、システムモジュールはシステムの振る舞いを記述するモジュールである。次に、データ型モジュールを、CafeOBJ に内蔵されている組み込みのモジュールと、ユーザが作成するユーザ定義のモジュールの2種類に分類する。さらに、ユーザ定義のモジュールを抽象度の低い始代数データ型と抽象度の高い抽象データ型に分ける。

## 3. CafeOBJ 仕様の実装法

上記のようにモジュールを分類し、モジュールごとに検討した実装法を表1に示す。

表 1:分類したモジュールごとの実装法

| 分類 | | | 実装方法 |
|---|---|---|---|
| データ型モジュール | 組み込み | | あらかじめ用意したデータ型に置き換え |
| | ユーザ定義 | 始代数 | Haskellの代数データ型を用いて実装 |
| | | 抽象 | 仕様を満たす任意の実装 |
| システムモジュール | | | 観測を引数とする関数を定義する |

データ型モジュールの実装方法を以下に示す。組込モジュールはあらかじめ Haskell 側に用意したデータ型に置き換える。始代数データ型は Haskell の代数データ型を用いて実装し、抽象データ型は仕様を満たす任意の実装を選択する。CafeOBJ 仕様の実装法は、まずそれらのデータ型を実装し、その後、実装した型を用いてシステムの実装を記述する。

次に、システムモジュールの実装法を示す。例として、銀行口座システム(ATM)の仕様を考える。ATM は、遷移演算として deposit(入金)と withdraw(出金)、観測演算として balance(残高確認)を宣言した仕様とする。以下に、ATM の仕様の一部である、入金を表す遷移演算 deposit を定義する等式を示す。

```
ceq balance(deposit(I,A)) = I + balance(A) if  I >= 0 .
```

ceq は条件付き等式を表し、if 以下の条件が成り立つとき、等式が成り立つ。事前状態 A に対して deposit(I,A) は金額 I を入金した後の状態である。また、balance(t)は状態 t の残高を表す。

関数型言語では、関数を組み合わせてプログラムを記述する。また、手続き型言語のようにシステムの状態を保持する変数を定義できない。関数型言語で繰り返しを表現する場合は関数の再起呼び出しを用いる。

システムモジュールの実装法は、観測演算による観測値を引数とし、再帰呼び出しによって変化していく引数を状態遷移として表現する( f n ⇒ f n' ⇒ … )。

この実装法に従って ATM 仕様を Haskell で実装した例を以下に示す。

```
atm n = … action a
              |a == "deposit" = deposit n m
              |a == "withdraw" = withdraw n m
deposit n m = if m >= 0 then atm (n + m) else atm n
```

関数 atm の引数 n は観測値(残高)に対応する。関数 atm n は、入力された文字列 a に応じて関数 deposit または withdraw で定義されている。関数 deposit は、遷移演算 deposit に対応する関数である。関数 deposit は、入力された預金額 m の値に応じて 関数 atm で定義されている。この際、atm の引数の変化で状態変化を実現している。

## 4. おわりに

本研究では、CafeOBJ 仕様をモジュール単位で分類し、それぞれ Haskell での実装法について検討した。今後は、本研究で取り扱わなかった記述方法に対する実装法や、本研究で検討した実装法に則った変換ツールの作成などが課題となる。

## 参考文献

[1]二木厚吉, 緒方和博, 中村正樹, CafeOBJ 入門(1) 形式手法と CafeOBJ, コンピュータソフトウェア, Vol.25, No.2, pp.1-13, 2008.


Copyright © 2017 IEICE