運賃計算システム VDM++ 仕様

佐原　伸

法政大学
情報科学研究科

概要

レコードの集合と関数を使った運賃計算の要求仕様記述例である。
型やインスタンス変数の不変条件、事後条件、事前条件、簡易回帰テストの例も含んでいる。
組み合わせテスト機能は、まだ、プロトタイプのため表示がおかしいし、説明はしないが、本モデルのテストには使用した。

目次

1　運賃クラス

要求仕様レベルの運賃を表す。

```
class
運賃
types
 public 駅 = seq of char
 inv w 駅 == w 駅 <> "";
public
 運賃レコード::f 駅 1 : 駅
              f 駅 2 : 駅
              f 運賃 : nat
 inv w 運賃レコード == w 運賃レコード.f 駅 1 <> w 運賃レコード.f 駅 2
```

1.1　運賃を得る

運賃レコードの集合から、2 駅間の運賃を計算する。

```
functions
public static
 運賃を得る : set of 運賃レコード * 駅 * 駅 -> nat
 運賃を得る (a 運賃集合, a 駅 1, a 駅 2) ==
   let  w 運賃レコード in set a 運賃集合 be st
            {a 駅 1, a 駅 2} = {w 運賃レコード.f 駅 1, w 運賃レコード.f 駅 2} in
   w 運賃レコード.f 運賃
 pre  {a 駅 1, a 駅 2} in set {{e.f 駅 1, e.f 駅 2} | e in set a 運賃集合 }
 post  exists1 w 運賃レコード in set a 運賃集合 &
          {a 駅 1, a 駅 2} = {w 運賃レコード.f 駅 1, w 運賃レコード.f 駅 2} and
          RESULT = w 運賃レコード.f 運賃
end
運賃
```

Test Suite :　vdm.tc

Class :　運賃

Name	#Calls	Coverage
運賃 ‘運賃を得る	0	0%

Name	#Calls	Coverage
Total Coverage		**0%**

2 Test クラス

運賃の回帰テストケースである。
エラーケースは、まだ考慮していない。

```
class
Test is subclass of 運賃
values

  w 運賃集合 = {
               mk_運賃レコード("東京","品川",220),
               mk_運賃レコード("東京","新宿",180),
               mk_運賃レコード("新宿","品川",190)}
functions
public
  run : () -> bool * seq of bool
  run() ==
    let testcases = [t1(),t2(),t3()] in
    mk_(forall i in set inds testcases & testcases(i),testcases);
public
  t1 : () -> bool
  t1() ==
    運賃得る(w 運賃集合,"東京","品川") = 220;
public
  t2 : () -> bool
  t2() ==
    運賃を得る(w 運賃集合,"東京","新宿") = 180;
public
  t3 : () -> bool
  t3() ==
    運賃を得る(w 運賃集合,"新宿","品川") = 200
end
Test
```

Test Suite : vdm.tc
Class : Test

Name	#Calls	Coverage
Test`t1	1	42%

Name	#Calls	Coverage
Test‘t2	0	0%
Test‘t3	0	0%
Test‘run	1	11%
Total Coverage		**13%**

3　組み合わせテストケース UseFare0

運賃の組み合わせテストケースである。
インスタンス変数の不変条件の例がある。
組み合わせテスト自体は、まだプロトタイプであるため説明しない。

```
class
UseFare0 is subclass of 運賃
instance variables
  s 運賃集合 : set of 運賃レコード := {
                  mk_運賃レコード("東京","品川",220),
                  mk_運賃レコード("東京","新宿",180),
                  mk_運賃レコード("新宿","品川",190),
                  mk_運賃レコード("新宿","品川",170)};
  inv  forall w 運賃レコード 1,w 運賃レコード 2 in set s 運賃集合 &
          w 運賃レコード 1.f 駅 1 = w 運賃レコード 2.f 駅 1 and
          w 運賃レコード 1.f 駅 2 = w 運賃レコード 2.f 駅 2 =>
          w 運賃レコード 1.f 運賃 = w 運賃レコード 2.f 運賃
  sTest : Test := new Test();

traces
T0 :
  sTest.run()
  ;
; T1 :
  let s1 in set {"東京","品川","新宿"} in
  let s2 in set {"東京","品川","新宿","武蔵境",""} in
  運賃を得る(s 運賃集合,s1,s2)
  ;
end
UseFare0
```

4　参考文献、索引

VDM++[2] は、1970 年代中頃に IBM ウィーン研究所で開発された VDM-SL[1] を拡張し、さらにオブジェクト指向拡張した形式仕様記述言語である。

参考文献

[1] Kyushu University. VDM-SL 言語マニュアル. Kyushu University, 第 2.0 版, 2016. Revised for VDMTools V9.0.2.

[2] Kyushu University. VDMTools VDM++ 言語マニュアル. Kyushu University, 第 2.0 版, 2016. Revised for VDMTools V9.0.2.

索引