電子錠シーケンシャル・モデル

佐原伸
日本フィッツ株式会社
情報技術研究所
TEL：03-3623-4683
shin.sahara@jfits.co.jp

2005 年 3 月 3 日

概要

ソフトウェア技術者協会 (SEA) 主催のデザインワークショップ 2005 の課題である電子錠システムを、VDM++ で並行処理を考えないモデルとして記述したものである。

0.0.1 Store

Store は、電子錠の付いた物置で、かつ、そのテスト用のクラスである。本来は、テスト機能は別クラスに分離すべきであるが、手を抜いた。

new Store().test() とすることで、想定した全てのテストを行う。

class $Store$ is subclass of $KeyCommon$

instance variables

public 錠 : 『錠』 := new 『錠』();
public 取手 : 『取手』 := new 『取手』();
public 施錠灯 : 『施錠灯』 := new 『施錠灯』();
public 表示窓 : 『表示窓』 := new 『表示窓』();
public ボタン : 『ボタン』 := new 『ボタン』();

operations

$Init$: () $\xrightarrow{o}$ ()
$Init$ () $\triangleq$
(取手.$Init$(施錠灯);
錠.$Init$(施錠灯, 取手, 表示窓, ボタン);
施錠灯.点ける();
取手を閉める();
表示窓.消す()
);

public
解錠する : 「鍵」 $\xrightarrow{o}$ ()
解錠する (a 鍵) $\triangleq$
(表示窓.ボタン列を設定する(a 鍵);
錠.解錠する()
);

public
施錠する : () $\xrightarrow{o}$ ()
施錠する () $\triangleq$
(ボタン.ボタンを設定する(L ボタン);
錠.施錠する()
);

public
取手を開ける : () $\xrightarrow{o}$ ()
取手を開ける () $\triangleq$ 取手.
開く();

public
取手を閉める : () $\xrightarrow{o}$ ()
取手を閉める () $\triangleq$ 取手.
閉める();

public
施錠鍵を登録する : 「鍵」 $\xrightarrow{o}$ ()
施錠鍵を登録する (a 鍵) $\triangleq$
(表示窓.ボタン列を設定する(a 鍵);
ボタン.ボタンを設定する(L ボタン);
錠.登録する()
);

以下はテスト機能である。

public
$test$: () $\xrightarrow{o}$ $\mathbb{B}$
$test$ () $\triangleq$
(return $t1$ () $\wedge$ $t2$ () $\wedge$ $t3$ () $\wedge$ $t4$ () $\wedge$ $t5$ () $\wedge$ $t6$ ()
);

public
$t1$: () $\xrightarrow{o}$ $\mathbb{B}$
$t1$ () $\triangleq$
($Init$();
解錠する([1, 2, 3, 4]);
return 施錠灯.消えている ()
);

public
$t2$: () $\xrightarrow{o}$ $\mathbb{B}$
$t2$ () $\triangleq$
($Init$();
解錠する([1, 2, 3, 4]);
取手を開ける();
return 施錠灯.消えている () $\wedge$
取手.開いている ()
);

public
$t3$: () $\xrightarrow{o}$ $\mathbb{B}$
$t3$ () $\triangleq$
($Init$();
解錠する([1, 2, 3, 4]);
取手を開ける();
施錠鍵を登録する([1, 9, 1, 9]);
return 施錠灯.消えている () $\wedge$
錠.鍵が一致 ([1, 9, 1, 9])
);

public
$t4$: () $\xrightarrow{o}$ $\mathbb{B}$
$t4$ () $\triangleq$
($Init$();
解錠する([1, 2, 3, 4]);
取手を開ける();
施錠鍵を登録する([1, 9, 1, 9]);
取手を閉める();
施錠する();
return 施錠灯.点いている () $\wedge$
表示窓.消えている ()
);

public

```
        t5 : () →o 𝔹
        t5 () ≜
          (   Init() ;
              解錠する([1, 2, 3, 4]) ;
              取手を開ける() ;
              施錠鍵を登録する([1, 9, 1, 9]) ;
              取手を閉める() ;
              施錠する() ;
              解錠する([1, 9, 1, 9]) ;
              return 施錠灯.消えている () ∧
                     取手.閉まっている ()
          );
public
        t6 : () →o 𝔹
        t6 () ≜
          (   Init() ;
              解錠する([1, 2, 3, 4]) ;
              取手を開ける() ;
              施錠鍵を登録する([1, 9, 1, 9]) ;
              取手を閉める() ;
              施錠する() ;
              解錠する([1, 9, 1, 9]) ;
              return 施錠灯.消えている () ∧
                     取手.閉まっている ()
          )
end Store
```

0.0.2 KeyCommon

共通の型や値を定義する。

```
class KeyCommon
values
public
        鍵桁数 = 4;
public
        L ボタン = 'L';
public
        C ボタン = 'C'
types
        public「鍵」= ℕ*
end KeyCommon
```

0.0.3 『錠』

電子錠である。

```
class『錠』is subclass of KeyCommon
instance variables
        public 施錠鍵 :「鍵」;
        public 取手 :『取手』;
        public 施錠灯 :『施錠灯』;
        public 表示窓 :『表示窓』;
        public ボタン :『ボタン』;
        inv 正しい鍵 (施錠鍵)

functions
public static
        正しい鍵 :「鍵」→ 𝔹
        正しい鍵 (a 鍵) ≜
          len a 鍵 = 鍵桁数 post len a 鍵 = 鍵桁数
operations
public
        鍵が一致 :「鍵」→o 𝔹
        鍵が一致 (a 鍵) ≜
          return 施錠鍵 = a 鍵;
public
        Init :『施錠灯』×『取手』×『表示窓』×『ボタン』→o ()
        Init (a 施錠灯, a 取手, a 表示窓, a ボタン) ≜
          (   施錠鍵 := [1, 2, 3, 4];
              施錠灯 := a 施錠灯;
              取手 := a 取手;
              表示窓 := a 表示窓;
              ボタン := a ボタン
          );
public
        『錠』: () →o『錠』
        『錠』() ≜
          (   return self
          );
public
        『錠』:「鍵」→o『錠』
        『錠』(a 鍵) ≜
          (   施錠鍵 := a 鍵;
              return self
          )
        pre 正しい鍵 (a 鍵) ;
public
        施錠する : () →o ()
        施錠する () ≜
          (   if 取手.閉まっている ()
              then (   表示窓.消す() ;
                       施錠灯.点ける()
                   )
              else skip
          )
        pre
表示窓.点いている () ∧ 施錠灯.消えている () ∧
取手.閉まっている () ∧ ボタン.L()

post
表示窓.消えている () ∧ 施錠灯.点いている () ∧
取手.閉まっている ();
public
```

解錠する : () $\xrightarrow{o}$ ()
解錠する () $\triangleq$
(if 表示窓.点いている ()∧鍵が一致 (表示窓.内容) ∧ 正しい鍵 (施錠鍵)
then (表示窓.消す() ;
施錠灯.消す()
)
else skip
)
pre 表示窓.点いている () ∧ 施錠灯.点いている () ∧ 正しい鍵 (施錠鍵)

post
表示窓.消えている () ∧ 施錠灯.消えている ();
public
登録する : () $\xrightarrow{o}$ ()
登録する () $\triangleq$
施錠鍵 := 表示窓.内容
pre 表示窓.点いている () ∧ 施錠灯.消えている () ∧ ボタン.L ()

post
鍵が一致 (表示窓.内容)
end『錠』

0.0.4 『施錠灯』

class『施錠灯』is subclass of $KeyCommon$
instance variables
public 点灯 : $\mathbb{B}$;

operations
public
点ける : () $\xrightarrow{o}$ ()
点ける () $\triangleq$
点灯 := truepost 点いている ();
public
消す : () $\xrightarrow{o}$ ()
消す () $\triangleq$
点灯 := falsepost 消えている ();
public
点いている : () $\xrightarrow{o}$ $\mathbb{B}$
点いている () $\triangleq$
return 点灯post 点灯;
public
消えている : () $\xrightarrow{o}$ $\mathbb{B}$
消えている () $\triangleq$
return ¬ 点灯post ¬ 点灯
end『施錠灯』

0.0.5 『表示窓』

class『表示窓』is subclass of $KeyCommon$
instance variables
public 内容 :「鍵」:= [];

operations
public
設定する :『ボタン』‘「ボタン」$\xrightarrow{o}$ ()
設定する (a ボタン) $\triangleq$
if len 内容 $\geq$ 鍵桁数
then skip
else 内容 := 内容 $\curvearrowright$ [a ボタン]
pre『ボタン』‘数字 (a ボタン)

post
if len 内容 $\geq$ 鍵桁数
then 内容 = $\overleftarrow{\text{内容}}$
else 内容 = $\overleftarrow{\text{内容}}$ $\curvearrowright$ [a ボタン];

以下の操作は、ボタン列の最後から鍵桁数分だけを内容に設定する。Drop は、自作ライブラリで定義されている関数で、Drop(n)(列) と呼ぶと、列から頭の n 個を削除した列を返す。

public
ボタン列を設定する :『ボタン』‘「ボタン」* $\xrightarrow{o}$ ()
ボタン列を設定する (a ボタン列) $\triangleq$
内容 := $FSequence$‘$Drop$[$\mathbb{N}$]
(
len 内容 + len a ボタン列 − 鍵桁数)
(
内容 $\curvearrowright$ a ボタン列)
pre『ボタン』‘数字列 (a ボタン列)

post
内容 =
$FSequence$‘$Drop$[$\mathbb{N}$]
(
len $\overleftarrow{\text{内容}}$+len a ボタン列−鍵桁数)
(
$\overleftarrow{\text{内容}}$ $\curvearrowright$ a ボタン列);
public
消す : () $\xrightarrow{o}$ ()
消す () $\triangleq$
内容 := []post 消えている ();
public
点いている : () $\xrightarrow{o}$ $\mathbb{B}$
点いている () $\triangleq$
return 内容 $\neq$ [] post 内容 $\neq$ [];
public
消えている : () $\xrightarrow{o}$ $\mathbb{B}$
消えている () $\triangleq$
return 内容 = [] post 内容 = []
end『表示窓』

0.0.6 『ボタン』

class『ボタン』is subclass of $KeyCommon$
types
　　public「ボタン」$= \mathbb{N} \mid$ char
　　inv ボタン $\triangleq$ 数字 (ボタン) $\vee$ ボタン $\in$ $\{'C', 'L'\}$
instance variables
　　public 内容 :「ボタン」$:= 'L'$;

functions
public static
　　数字 :「ボタン」$\to \mathbb{B}$
　　数字 (a ボタン) $\triangleq$
　　a ボタン $\in \{0, \ldots, 9\}$;
public static
　　数字列 :「ボタン」$^* \to \mathbb{B}$
　　数字列 (a ボタン列) $\triangleq$
　　$\forall\, b \in$ elems a ボタン列 $\cdot$ 数字 (b)
operations
public
　　C : () $\xrightarrow{o} \mathbb{B}$
　　C () $\triangleq$
　　return 内容 $= C$ ボタン;
public
　　L : () $\xrightarrow{o} \mathbb{B}$
　　L () $\triangleq$
　　return 内容 $= L$ ボタン;
public
　　ボタンを設定する :「ボタン」$\xrightarrow{o}$ ()
　　ボタンを設定する (a ボタン) $\triangleq$
　　内容 $:=$ a ボタン
end『ボタン』

0.0.7 『取手』

class『取手』is subclass of $KeyCommon$
instance variables
　　public 開閉 : $\mathbb{B} :=$ false;
　　public 施錠灯 :『施錠灯』;

operations
public
　　$Init$:『施錠灯』$\xrightarrow{o}$ ()
　　$Init$ (a 施錠灯) $\triangleq$
　　施錠灯 $:=$ a 施錠灯;
public
　　開く : () $\xrightarrow{o}$ ()
　　開く () $\triangleq$
　　if 施錠灯.消えている ()
　　then 開閉 $:=$ true
　　else 開閉 $:=$ falsepost if 施錠灯.消えている ()
　　　　then 開閉 $=$ true
　　　　else 開閉 $=$ false;
public
　　閉める : () $\xrightarrow{o}$ ()
　　閉める () $\triangleq$
　　開閉 $:=$ falsepost 閉まっている ();
public
　　開いている : () $\xrightarrow{o} \mathbb{B}$
　　開いている () $\triangleq$
　　return 開閉post 開閉;
public
　　閉まっている : () $\xrightarrow{o} \mathbb{B}$
　　閉まっている () $\triangleq$
　　return $\neg$開閉post $\neg$開閉
end『取手』