小規模組み込みシステム向けFRP言語への文脈指向プログラミング機構の導入

渡部卓雄（東京工業大学 情報理工学院）

小規模組み込みシステム向けに設計された関数リアクティブプログラミング(FRP)言語Emfrpのための文脈指向プログラミング機構を提案する．提案方式では時変値の値にもとづく層の活性化機構を導入し，実行時文脈の変化に伴う振る舞いの動的な変更をFRPの枠組み内で宣言的に記述できるようにしている．これにより，小規模組込みシステムにおける適応動作のモジュール化が可能になる．

関数リアクティブプログラミング(FRP)

関数リアクティブプログラミング(FRP)は，時間と共に変化する値を抽象化した時変値(time-varying value)やイベントストリーム等を用いることで，リアクティブシステムの宣言的な記述を支援するプログラミングパラダイムである．当初はHaskellによるGUI等の記述方式として導入された．FRPを用いることで，ハンドラ（コールバック），イベントループ，ポーリングなどを排して，組込みシステムの動作を見通しよく記述することが可能になる．

我々の研究室では，マイクロコントローラなどの小規模システムを対象とする純粋FRP言語Emfrpを開発している[1]．右はEmfrpによるファンの制御プログラムである．温度および湿度センサの計測値から計算される不快指数が閾値以上である間ファンを動作させる．

```
module FanController  # module name
in  tmp : Float,      # temperature sensor
    hmd : Float       # humidity sensor
out fan : Bool        # fan switch
use Std               # standard library

# discomfort (temperature-humidity) index
node di = 0.81 * tmp +
          0.01 * hmd * (0.99 * tmp - 14.3) + 46.3

# fan switch
node fan = di >= th

# threshold
node th = 75.0
```

EmFrpによるファン制御プログラム

```
module FanController2  # module name
in  tmp   : Float,     # temperature sensor
    hmd   : Float,     # humidity sensor
    clock : Int        # POSIX time system clock
out fan   : Bool,      # fan switch
    ctime : Int        # LCD for time
use Std                # standard library

# discomfort (temperature-humidity) index
node di = 0.81 * tmp +
          0.01 * hmd * (0.99 * tmp - 14.3) + 46.3

# fan switch
node init[False] fan = di >= th

# threshold
node th = 75.0

# hysteresis behavior
layer HYSTERESIS where fan
  enter node th = @proceed - 0.5
  exit node th = @proceed + 0.5

# cumulative operating time
layer CTIME where fan
  enter node init[0] ctime0 = clock - ctime@last
  retain node init[0] ctime = clock - ctime0

# continuous operating time
layer OTIME where fan
  enter node init[0] otime0 = clock
  retain node init[0] otime = clock - otime0

# stop the fan for 30 minutes
# after 2 hours continuous operation
layer SHUTDOWN where (otime > 7200 || timer > 0)
  enter node init[0] timer0 = clock + 1800
  retain node init[0] timer = timer0@last - clock
  node fan = False
```

機能が追加されたファン制御プログラムのCOP機構による記述

文脈指向プログラミング(COP)機構

上に示したファン制御プログラムに次の機能を追加したい：(a)不快指数の値が閾値近辺で変化した際にファンのスイッチが頻繁にON/OFFを繰り返さないようにするヒステリシス制御，(b)ファンが現在までにONになった合計時間のLCDへの表示，および (c)ファンのモーターが連続して2時間以上ONになった際に強制的に30分間OFFにする保護機能．

これらの機能は，実行時文脈（例えばファンがONになっている文脈やモーター保護タイマーが有効である文脈等）に依存する動作として条件式(if式)を伴って記述される．しかし文脈に依存する範囲が広くなるにつれて，同様の条件を持つif式がプログラム中に散在しがちになる．また，複数の文脈に依存した処理はネストしたif式となり，プログラムのモジュール性・可読性を低下させる．

実行時文脈に依存する動作をモジュール化するためのプログラミングパラダイムが文脈指向プログラミング(COP)である．典型的なCOPは，文脈に依存する処理を分離して記述するための層(layer)と呼ばれる言語機構，および実行時情報に依って適切な層を活性化する実行時機構によって実現されている．

本研究ではEmfrpにCOP機構を導入し，文脈依存の動作のモジュール化を可能にする．具体的には layer 名前 where 条件 という構文により，条件が満たされている間に活性化する層を定義できるようにしている．加えて，層が活性化（および非活性化）する瞬間における時変値の値を保持する機構を導入することで，文脈に依存する様々な振る舞いを条件式等を多用せずに簡潔に記述できるようになっている．

左は上記(a)〜(c)の機能を追加したファン制御プログラムをCOP機構を用いて記述したものである．追加機能が違いに干渉することなく簡潔に記述できている．詳細は文献[2]を参照されたい．

[1] Sawada, K. & T. Watanabe, "Emfrp: A Functional Reactive Programming Language for Small-Scale Embedded Systems", CROW 2016, ACM, 2016.
[2] Watanabe, T., "A Simple Context-Oriented Programming Extension to an FRP Language for Small-Scale Embedded Systems", COP 2018, ACM, 2018.