5分間で Reactive Programming について説明する

shelarcy
(shelarcy@capella.freemail.ne.jp)

Reactive Programming とは

- ある出来事に対し反応 (reaction) を返すという形でプログラミングを行うシステムのこと
- 主な対象
  - Robot
  - Animation
  - GUI

Reactive Programming の実現方法

- 言語処理系に持たせる
  - Reactive-C
  - The SL Language
- フレームワークで実装する
  - Squeak の Morphic
  - Functional Reactive Programming

Reactive Programming の実装方法

- Stream (信号処理)を使う
- 継続を使う

- Callback でも実装できるが、準備や更新される変数の手続きなどの処理が必要なためコードが醜くなる

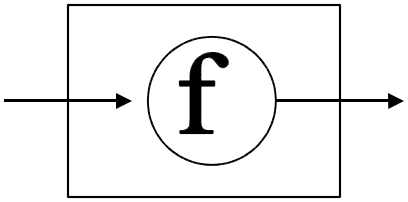
Functional Reactive Programming

- FrTime
  - 昨年本家 LL3 の Functional Reactive Programming in Scheme で取り扱われた PLT Scheme による実装
- Yampa
  - Haskell による Arrowised Functional Reactive Programming という実装

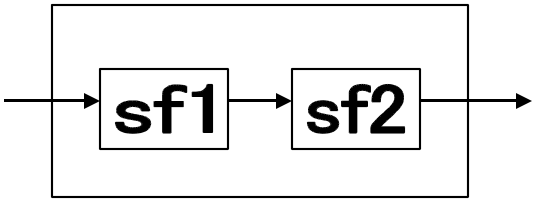
Arrow とは

- Monad の代わり
- 当初の論文 “Generalising Monad to Arrow” の頃から Signal を扱うものとして考えられている
- GHC で利用できる

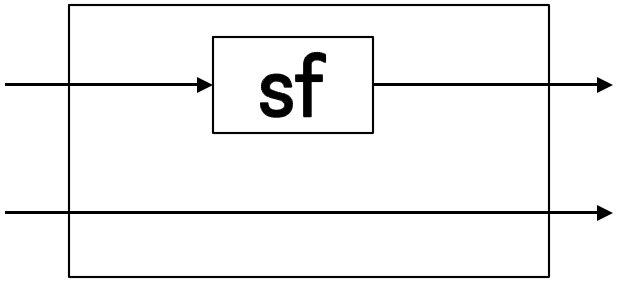
Arrow の関数



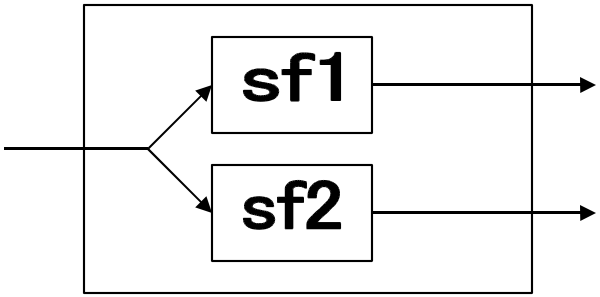
(a) arr f



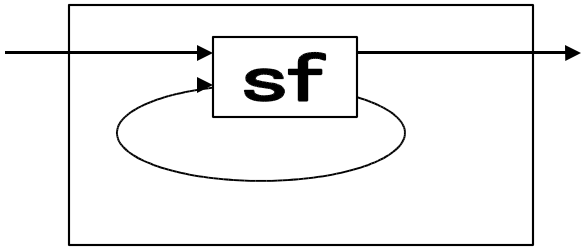
(b) sf1 >>> sf2



(c) first sf



(d) sf1 &&& sf2

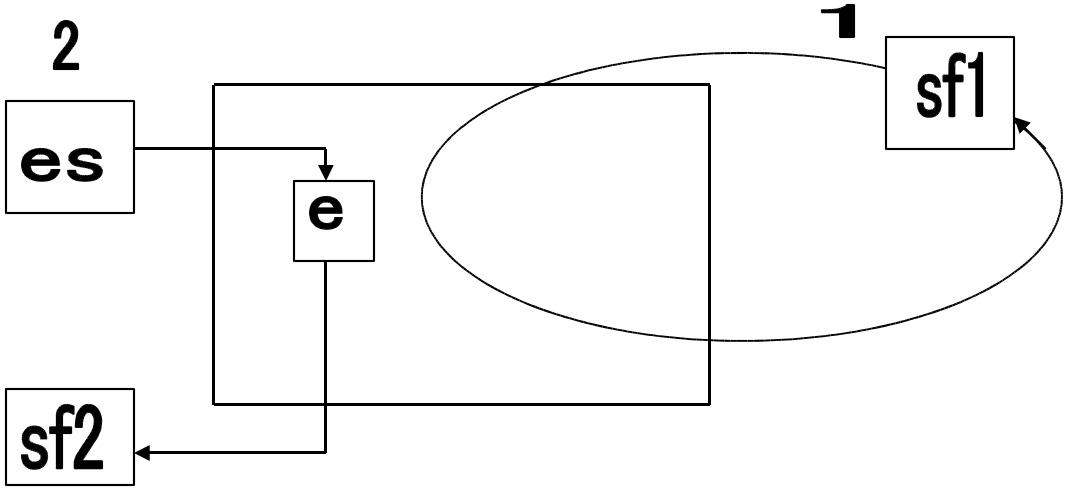


(e) loop sf

Yampa の有用な関数

- after
- never
- mergeBy
- repeatedly
- switch

Switch 系の関数



(sf1 &&& es) `switch` \e → sf2

- d の接頭辞がつくと delayed (遅延処理)
- r の接頭辞がつくと rec のついた変数を再帰的に処理