

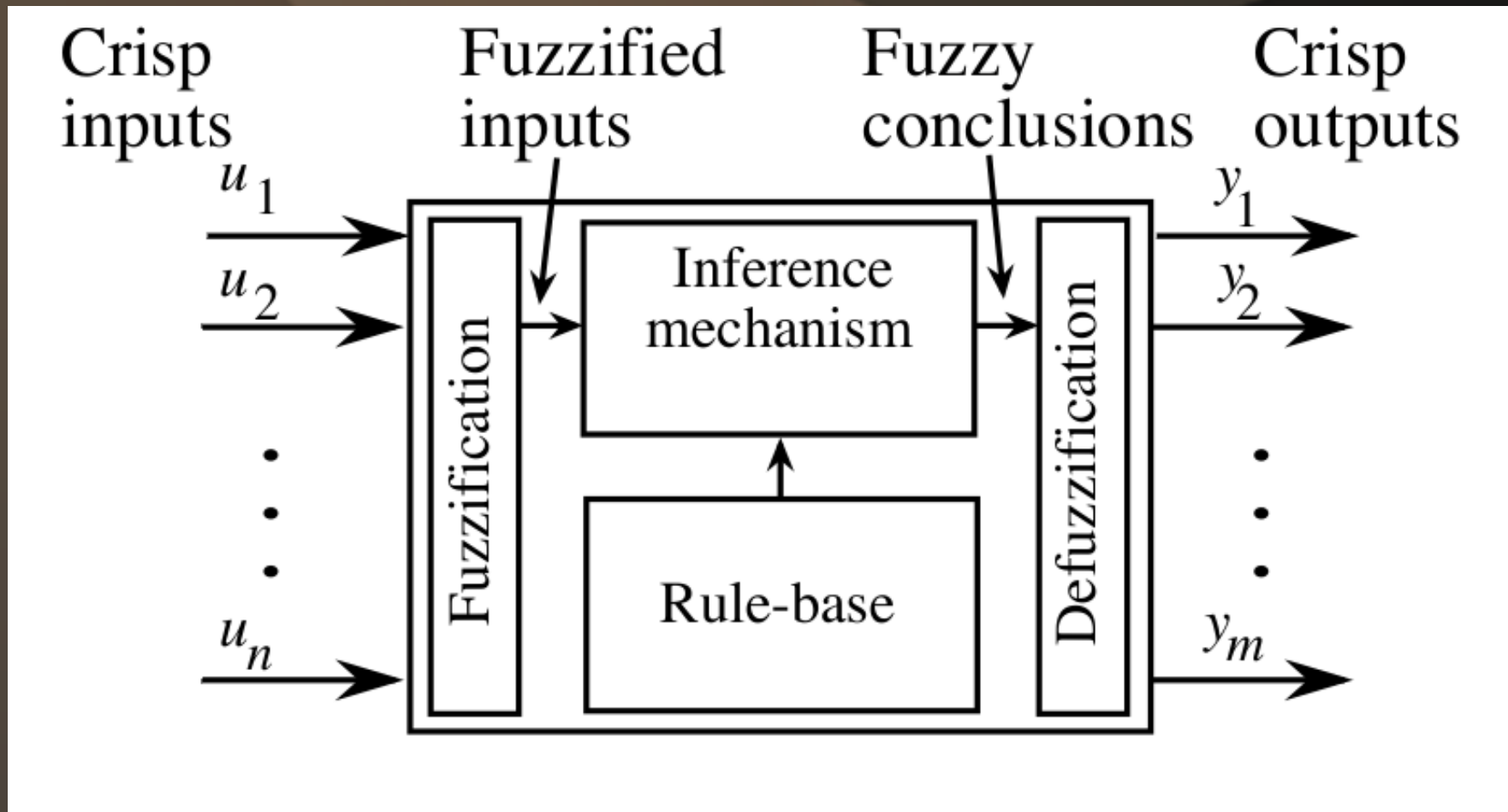
# **Um utilitário para controle fuzzy no EPOS**

**Edmar Miranda  
Gustavo Roberto Nardon Meira**

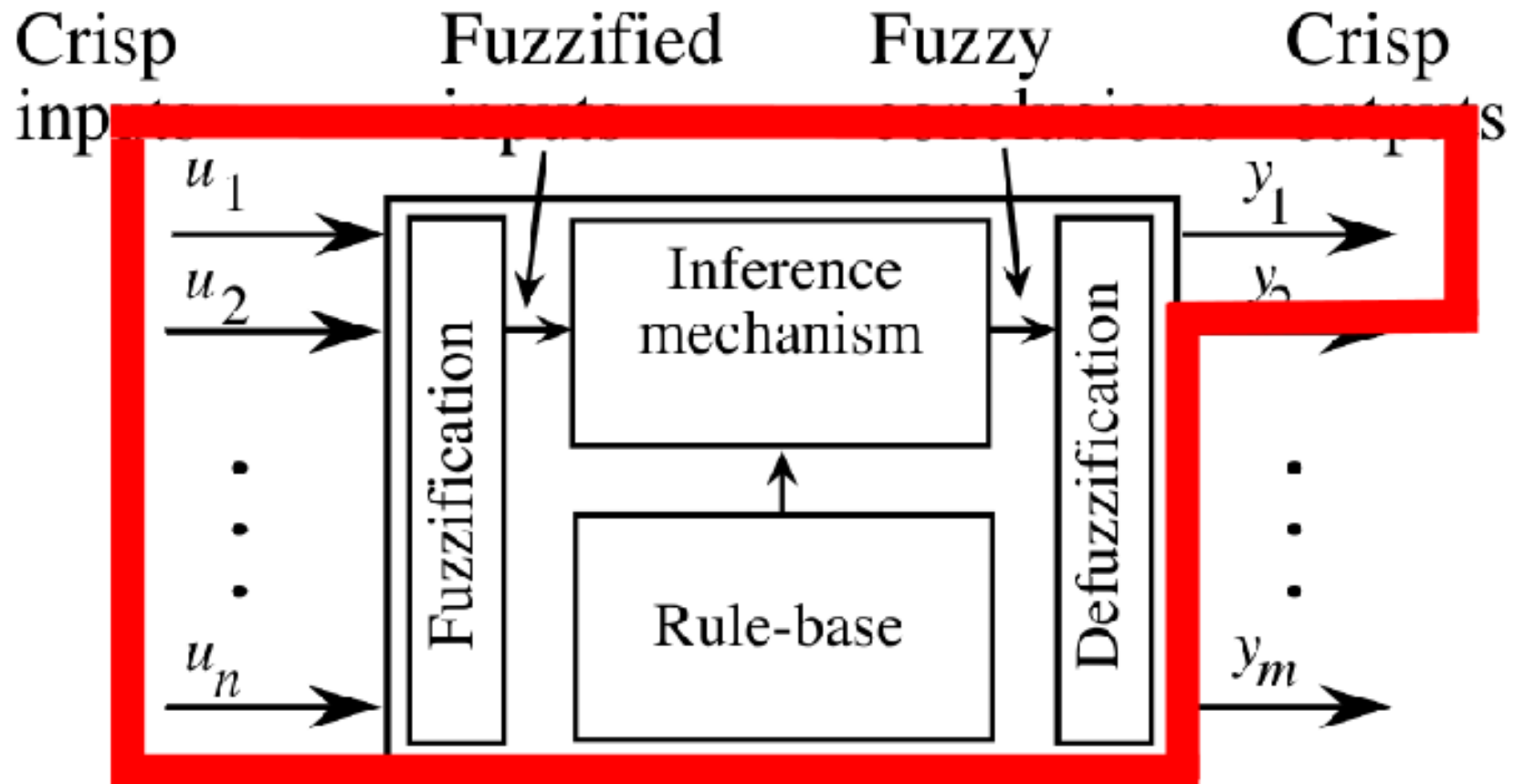
# *Introdução*

- EPOS
  - Aplicações dedicadas
  - ADESD, OOP
  - Componentes
    - Init
    - System
    - Setup
    - Utility
- Objetivo: criar um ambiente para controle Fuzzy

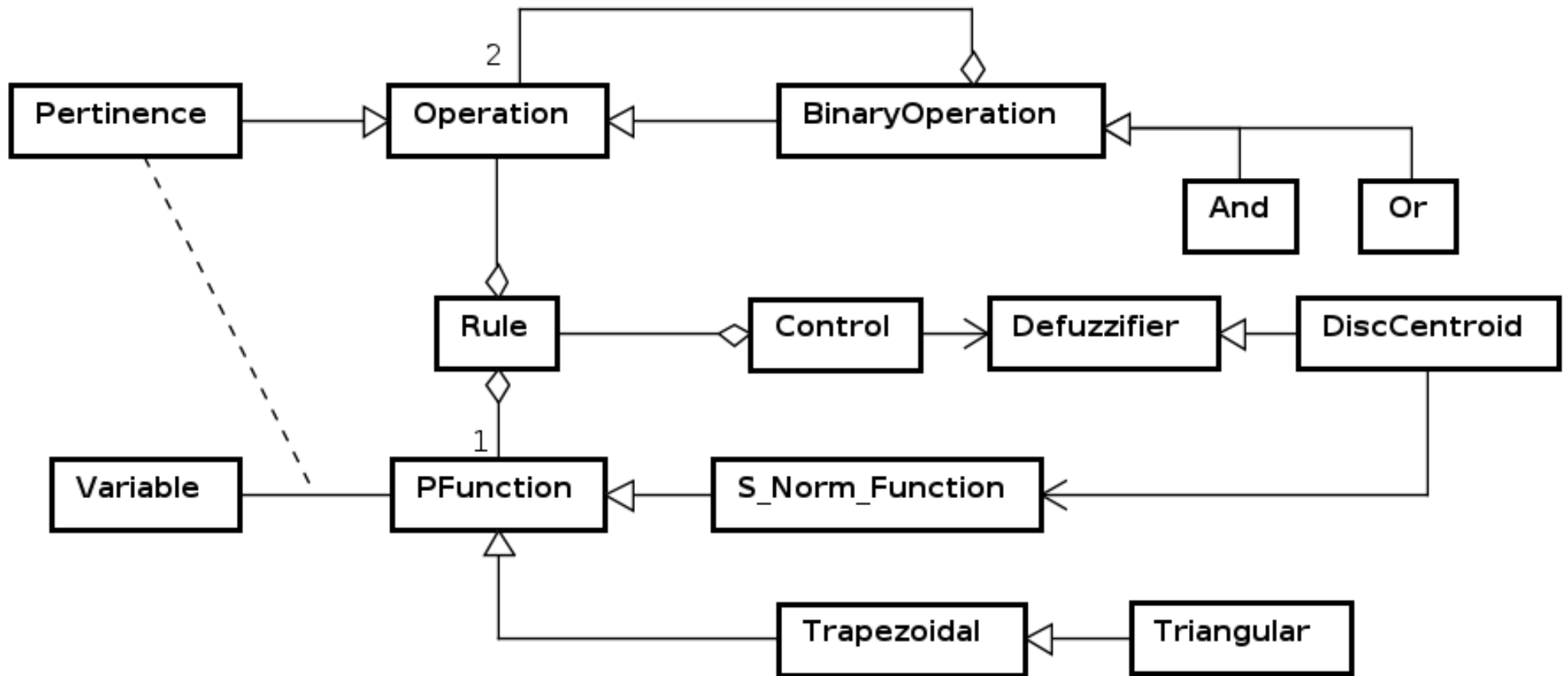
# *Modelo de Controlador*



# Modelo de Controlador



# Projeto



# *Exemplo*

- EPOS @ ARM (Integrator CP no QEMU)
- Aplicação
  - Estufa: controle de umidade e temperatura através de um exaustor
    - temp. baixa e umid. alta: tempo de espera alto
    - temp. alta e umid. baixa: tempo de espera baixo

# *Exemplo*

```
Variable temp(15);
```

```
Variable humi(50);
```

```
DiscCentroid m(10);
```

```
Control c(&m);
```

# *Exemplo*

```
PFunction* t_baixa = new Triangular(0,10,20);
PFunction* t_alta  = new Triangular(10,30,40);

PFunction* h_baixa = new Triangular(0,75/2.0,75);
PFunction* h_alta  = new Triangular(25,25+75/2.0,100);

PFunction* espera_alta = new Trapezoidal(50, 70, 80, 100, 0.7);
PFunction* espera_baixa = new Trapezoidal(30, 40, 60, 70, 0.9);

Pertinence temp_baixa(&temp, t_baixa);
Pertinence temp_alta(&temp, t_alta);

Pertinence humi_baixa(&humi, h_baixa);
Pertinence humi_alta(&humi, h_alta);
```

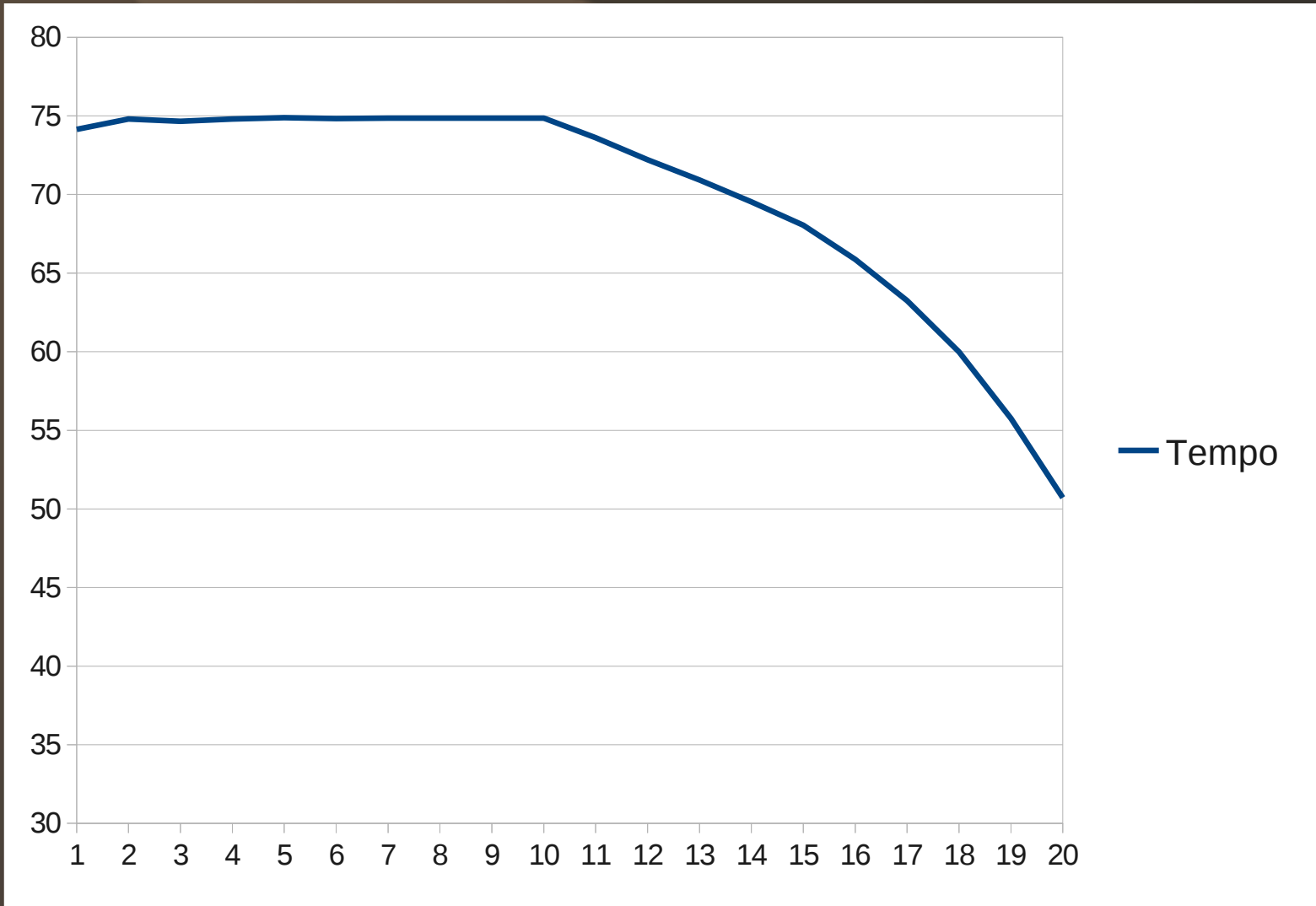
# *Exemplo*

```
And a(&temp_baixa, &humi_alta);
And b(&temp_alta, &humi_baixa);

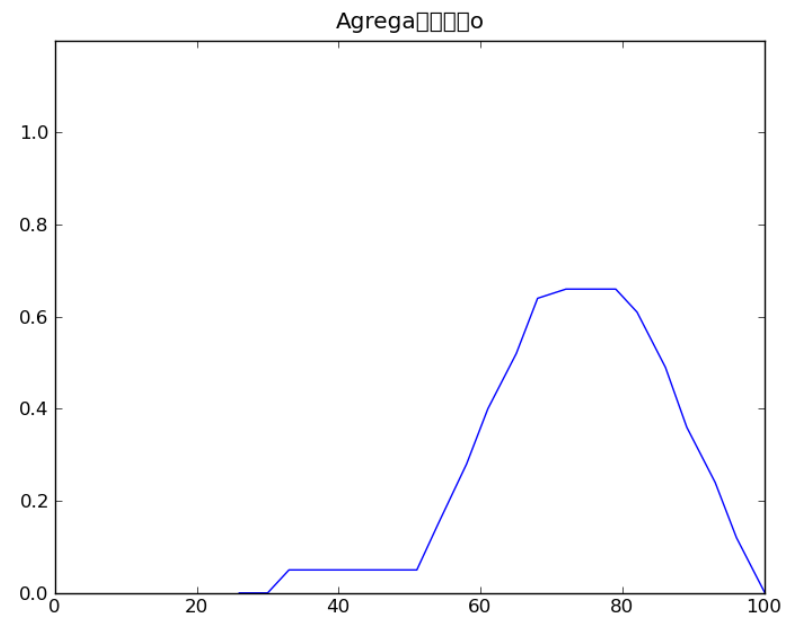
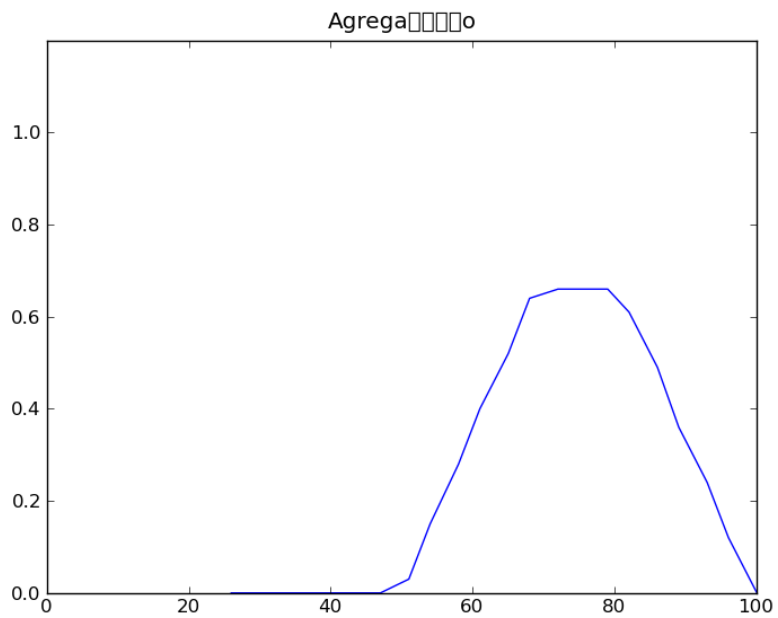
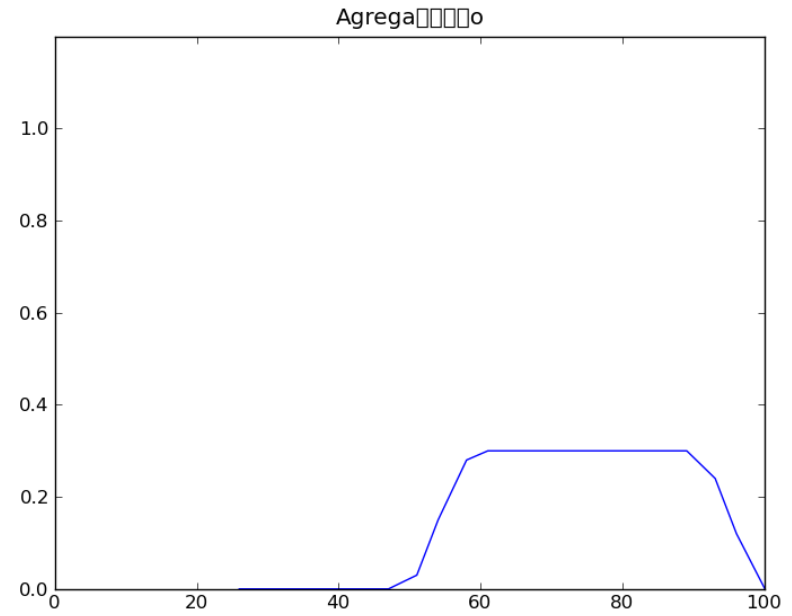
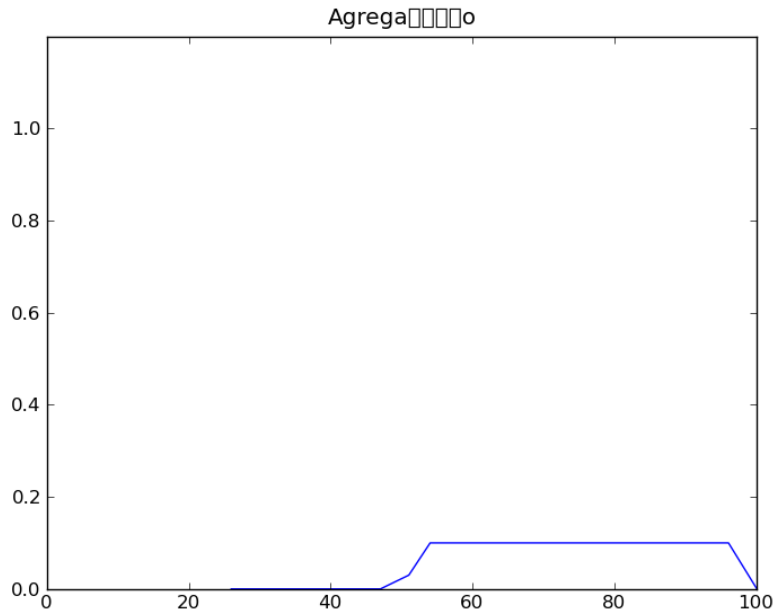
c.add_rule(&a, espera_alta);
c.add_rule(&b, espera_baixa);

for(int i = 0; i <= 20; i++){
    temp.change(i);
    c.refresh();
    cout << i << " " << (int)(100*c.out()) << "\n";
}
```

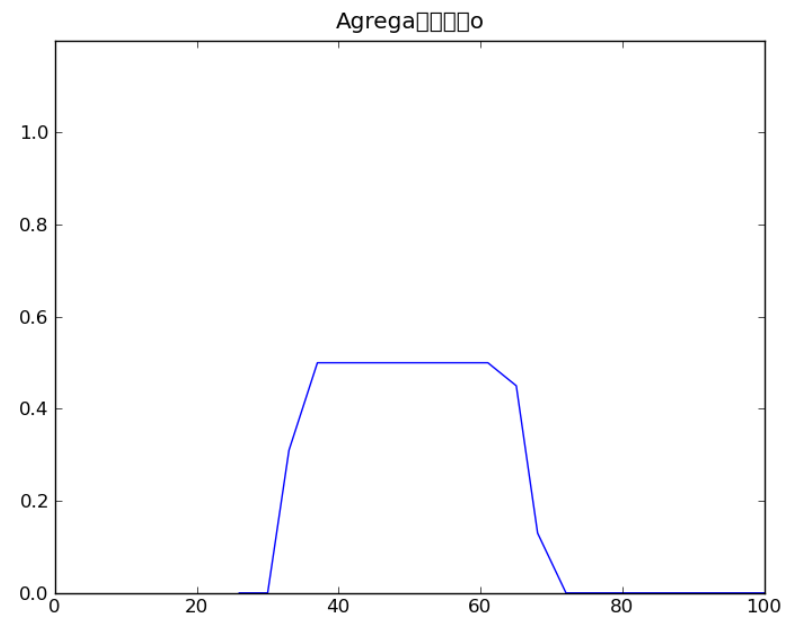
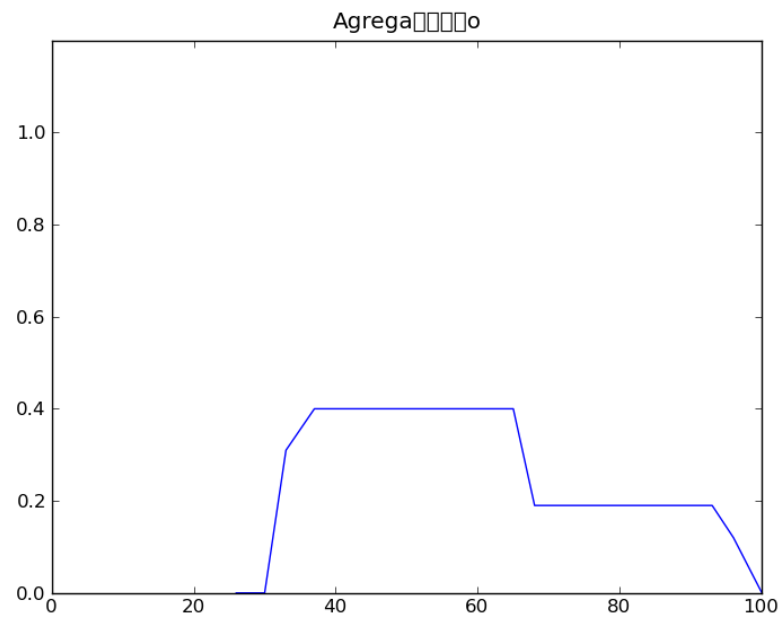
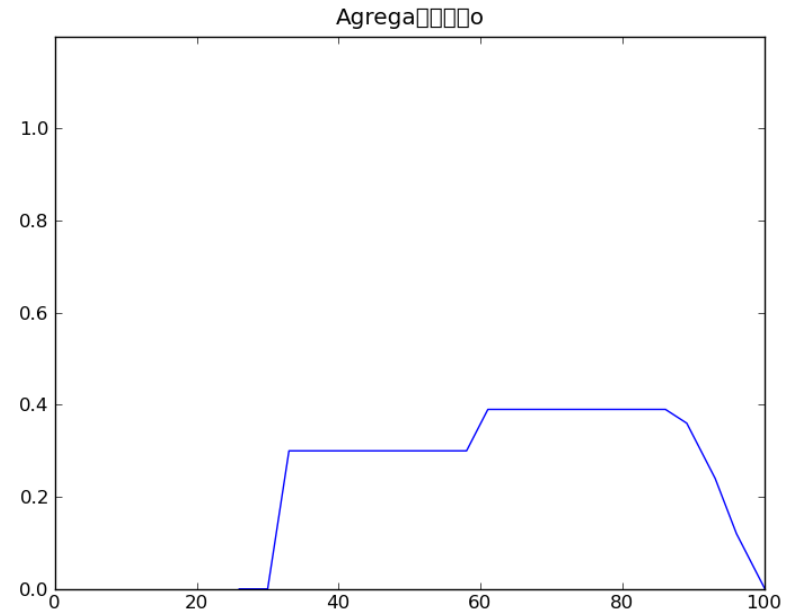
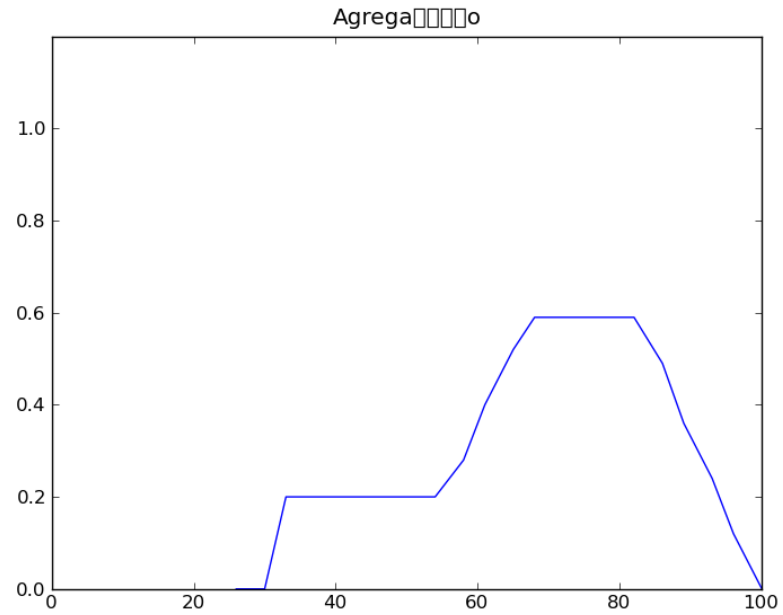
# *Resultado*



# Resultado



# Resultado



# *Conclusões*

- Ruim para sistemas embarcados com altas restrições de memória e processamento
- A framework está simples para ser estendido e pode ser utilizado em aplicações EPOS