



Leilão Online

Danilo Soares - dlsb@cin.ufpe.br

Fábio Soares - fss6@cin.ufpe.br



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Introdução

Leilão Online

Modalidade de negociação do qual compradores e vendedores necessitam solucionar de maneira simples e rápida compra(s) e/ou venda(s) de bens e produtos.



40
anos
1974-2014



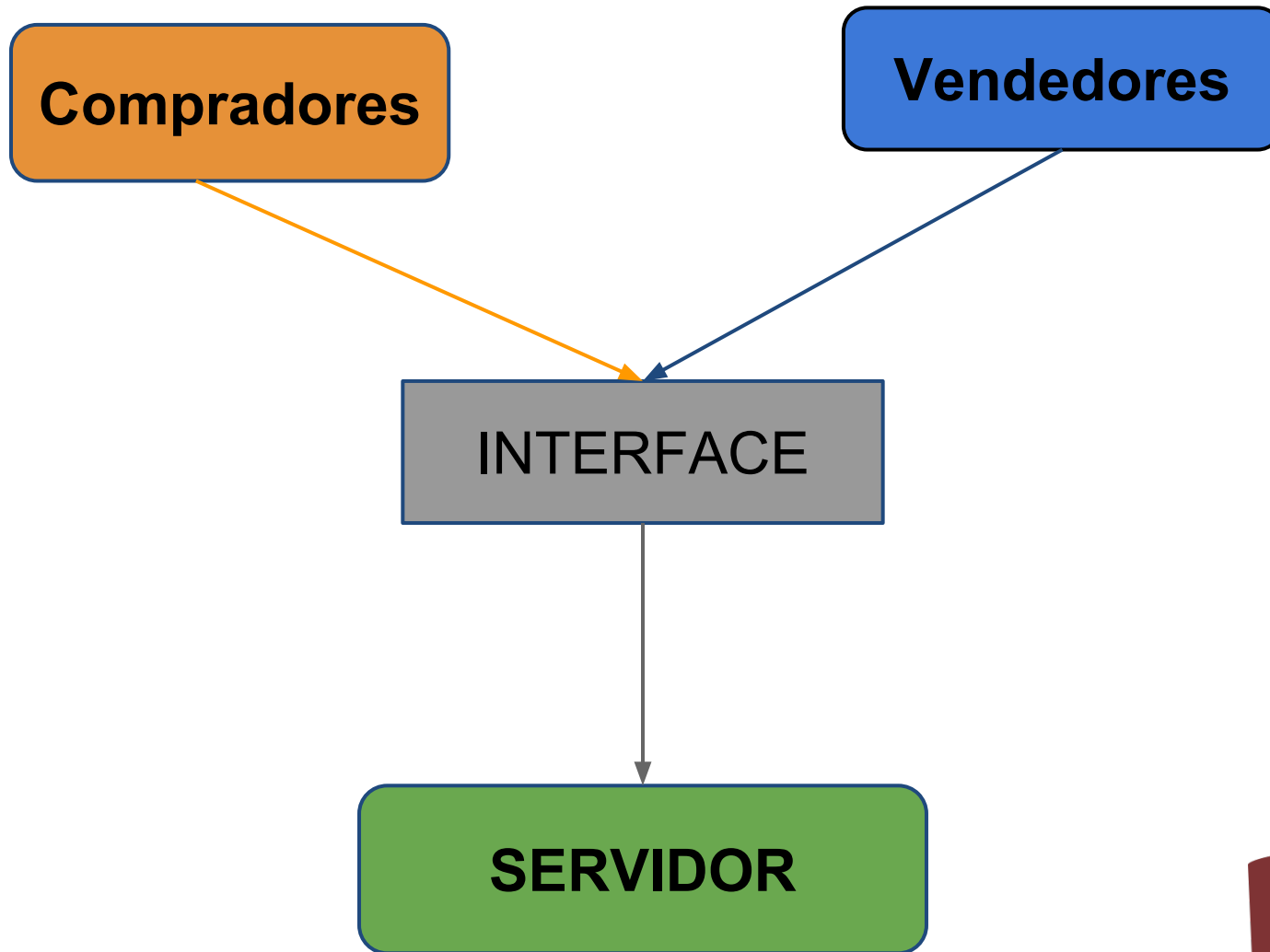
UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Cenário



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO





Vendedores

- id
 - Produto
-
- Vender!id!produto

Compradores

- id
 - Produto
-
- Comprar!id!produto



INTERFACE

- Vender?id?produto
- Comprar?id?produto
- EfetuarLance?id?produto
- iniciarLeilao!id!produto
- cadastrar!comprador!produto
- registrarLance!comprador!produto

SERVIDOR

- Vendedores:{(vendedor,produto)...}
- Compradores:{(comprador,produto)...}
- -----
- iniciarLeilao!id!produto
- cadastrar!comprador!produto
- registrarLance!comprador!produto
- cadastrarComprador
- validarLance
- encerrarLeilao



Tipos/Dados - Canais/Eventos

```
1  -----
2  -- TIPOS / DADOS
3  -----
4  N = 3
5  id_vendedor = { 1..N }
6  id_comprador = { 1..N }
7  datatype produto = notebook | tablet
8
9  -----
10 -- CANAIS / EVENTOS
11 -----
12
13 channel vender, iniciarLeilao : id_vendedor.produto
14 channel comprar, cadastrar, efetuarLance, registrarLance : id_comprador.produto
15 channel encerrarLeilao, cadastrarComprador, validarLance
16
```



Comprador(es) e Vendedor(es)

```
17 -----
18 -- COMPRADOR(ES) / VENDEDOR(ES)
19 -----
20
21 COMPRADOR(id, produto) =
22     comprar!id!produto -> COMPRADOR(id, produto)
23     □
24     efetuarLance!id!produto -> COMPRADOR(id, produto)
25
26 COMPRADORES = ||| comprador:id_comprador, p:produto @ COMPRADOR(comprador, p)
27
28 assert COMPRADORES :[deadlock free]
29 assert COMPRADORES :[deterministic]
30
31
32 VENDEDOR(id, produto) =
33     vender!id!produto -> VENDEDOR(id, produto)
34
35 VENDEDORES = ||| id:id_vendedor, p:produto @ VENDEDOR(id, p)
36
37 assert VENDEDORES :[deadlock free]
38 assert VENDEDORES :[deterministic]
39
```




INTERFACE

```
40 -----
41 -- INTERFACE
42 -----
43
44 INTERFACE =
45     vender?vendedor?produto -> iniciarLeilao!vendedor!produto -> INTERFACE
46     □
47     comprar?comprador?produto -> cadastrar!comprador!produto -> INTERFACE
48     □
49     efetuarLance?comprador?produto -> registrarLance!comprador!produto -> INTERFACE
50
51 assert INTERFACE :[deadlock free]
52 assert INTERFACE :[deterministic]
53
```



SERVIDOR

```
76
77 SERVIDOR(leiloes, compradores) =
78   (iniciarLeilao?vendedor?produto ->
79     if not existeProduto(produto, leiloes) then
80       SERVIDOR(union(leiloes,{(vendedor, produto)}), compradores)
81     else
82       SERVIDOR(leiloes, compradores)
83   )
84   □
85   (cadastrar?comprador?produto ->
86     if not existeCompradorProduto(comprador,produto,compradores) then
87       cadastrarComprador -> SERVIDOR(leiloes, union(compradores,{(comprador, produto)}))
88     else
89       SERVIDOR(leiloes, compradores)
90   )
91   □
92   (registrarLance?comprador?produto ->
93     if existeCompradorProduto(comprador,produto,compradores) then
94       validarLance -> ( SERVIDOR(leiloes, compradores)
95         |~| encerrarLeilao -> SERVIDOR(remover(produto,leiloes),remover(produto,compradores)))
96     else
97       SERVIDOR(leiloes, compradores)
98   )
99
100 assert SERVIDOR({},{}) :[deadlock free]
101 assert SERVIDOR({},{}) :[deterministic]
102
```



Funções

```
54 -----
55 -- SERVIDOR
56 -----
57 -- "Verifica se o um dado produto existe em um determinado conjunto."
58 existeProduto(produto, C) =
59   if card(verificacaoProduto(produto, C)) > 0 then True else False
60
61 verificacaoProduto(produto, C) = { p | (v,p) <- C, p == produto}
62 -----
63 -- "Verifica se existe a tupla (comprador,produto) em um dado conjunto."
64 existeCompradorProduto(comprador,produto,C) =
65   if card(verificacaoCompradorProduto(comprador,produto, C)) > 0 then True else False
66
67 verificacaoCompradorProduto(comprador,produto,C) = { (c,p) | (c,p) <- C, (c,p) == (comprador,produto)}
68 -----
69 -- "Remove um dado produto de um determinado conjunto."
70 remover(produto,C) =
71   efetuarRemocao(produto, C)
72
73 efetuarRemocao(produto, C) =
74   {(id,p) | (id,p) <- C, p != produto}
75
```



SISTEMA

```
103 -----
104 -- ASSERTS SISTEMA
105 -----
106
107 SI = SERVIDOR({}, {}) [I{|iniciarLeilao, cadastrar, registrarLance|}] INTERFACE
108 SIC = SI [I{|comprar, efetuarLance|}] COMPRADORES
109 SICV = SIC [I{|vender|}] VENDEDORES
110 assert SICV :[deadlock free]
111 assert SICV :[deterministic]
112
```



REFINAMENTO

```
113 -----
114 -- REFINAMENTO
115 -----
116
117 S2(leiloes, compradores) =
118   (iniciarLeilao?vendedor?produto ->
119     if not existeProduto(produto, leiloes) then
120       S2(union(leiloes,{(vendedor, produto)}), compradores)
121     else
122       S2(leiloes, compradores)
123   )
124   □
125   (cadastrar?comprador?produto ->
126     if not existeCompradorProduto(comprador, produto, compradores) then
127       cadastrarComprador -> S2(leiloes, union(compradores,{(comprador, produto)}))
128     else
129       S2(leiloes, compradores)
130   )
131   □
132   (registrarLance?comprador?produto ->
133     if existeCompradorProduto(comprador, produto, compradores) then
134       validarLance -> ( S2(leiloes, compradores)
135         □ encerrarLeilao -> S2(remover(produto, leiloes), remover(produto, compradores))
136       )
137     else
138       S2(leiloes, compradores)
139   )
140   assert S2({}, {}) :[deadlock free]
141   assert S2({}, {}) :[deterministic]
142   assert S2({}, {}) [T= SERVIDOR({}, {})]
143   assert SERVIDOR({}, {}) [T= S2_({}, {})]
144
```



Verificação no FDR



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO



Dificuldades encontradas

- Requisitos vs Modelagem (Como modelar?);
- Escopo vs Especificação (Estouro de memória);
- Pouca experiência em sistemas paralelos/concorrentes;
- Relevância no refinamento.



Obrigado!



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO