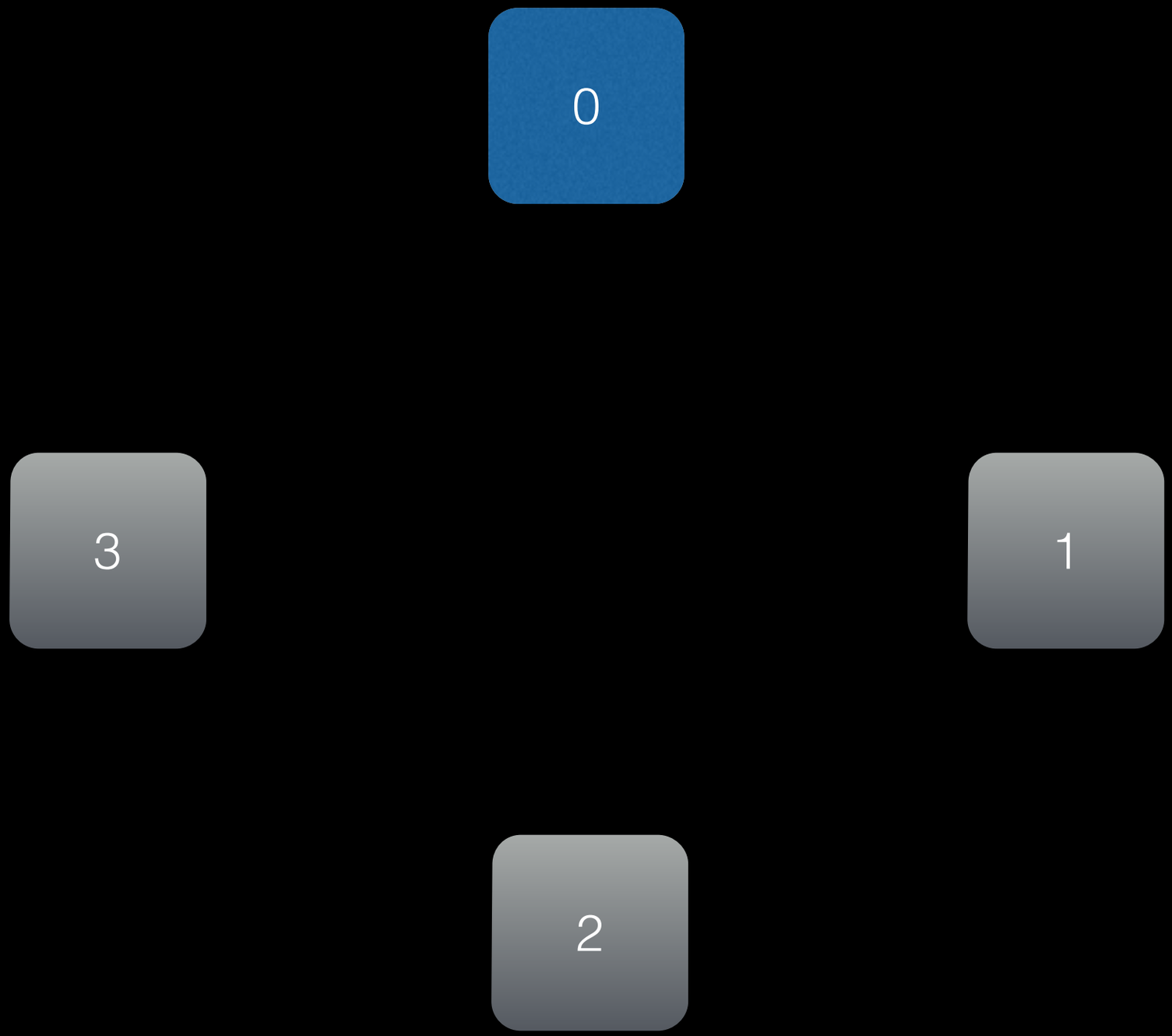
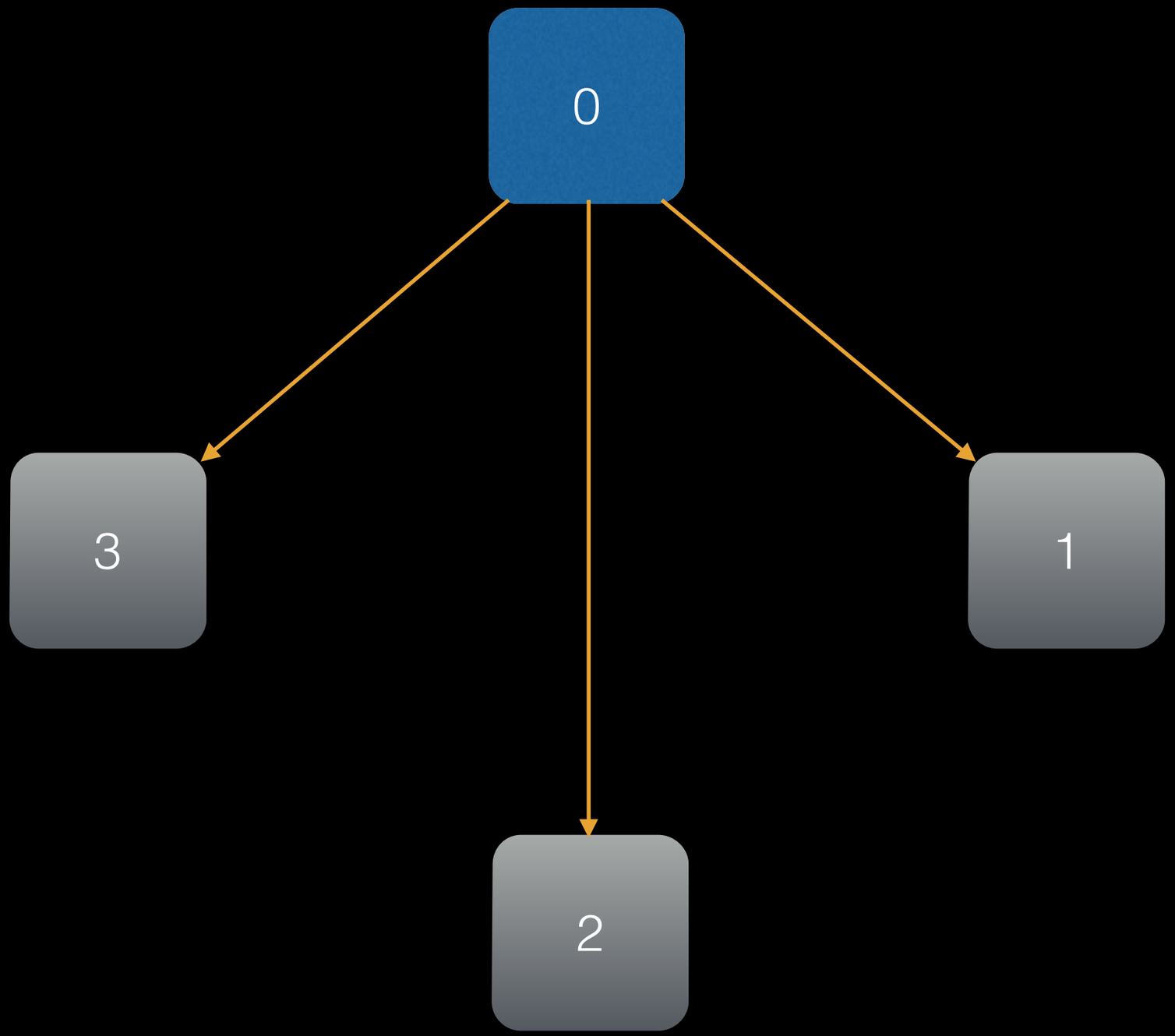
Estudo de caso: protocolo distribuído de detecção de término

David Déharbe

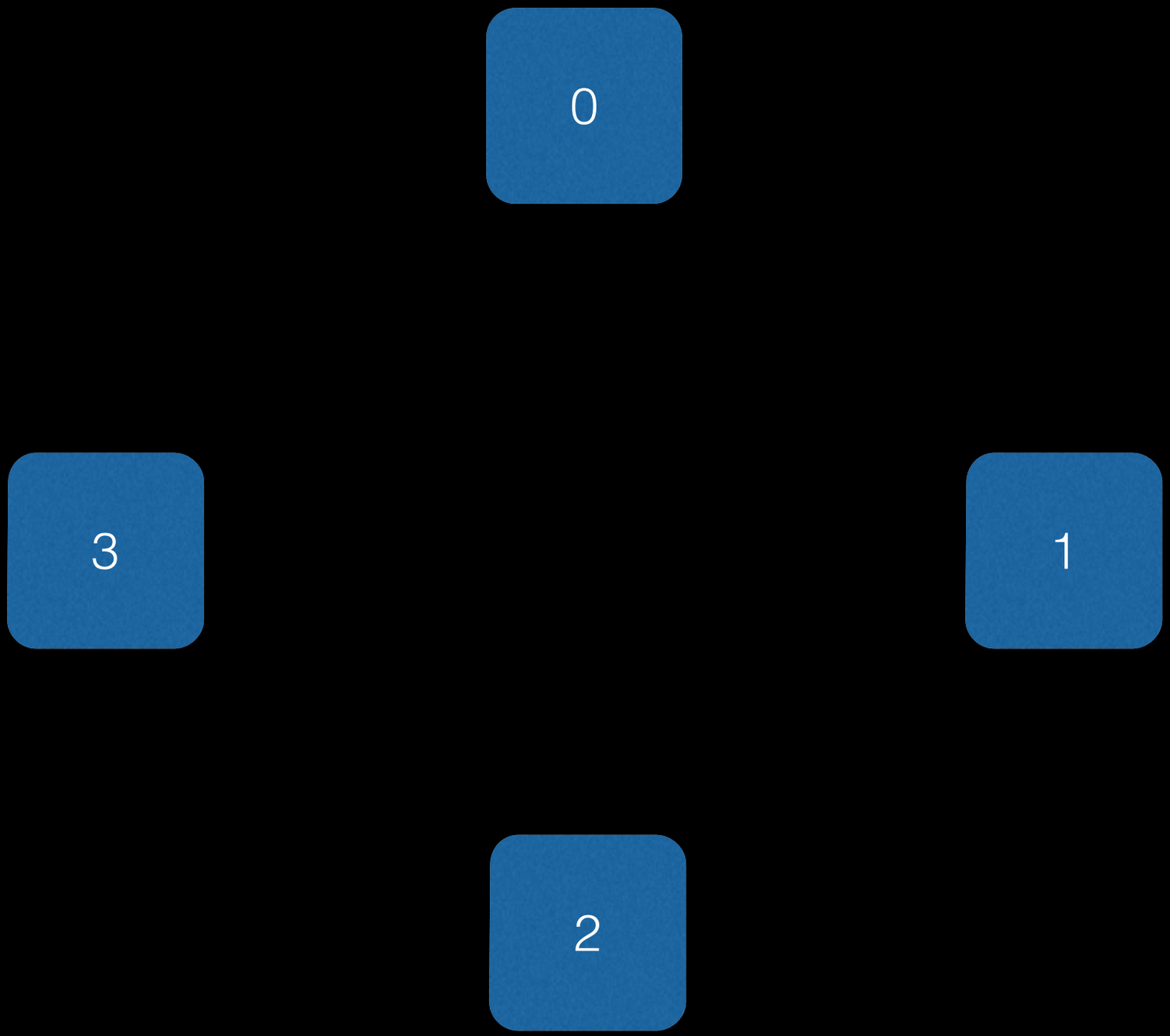
Processadores: 0, 1, 2 e 3



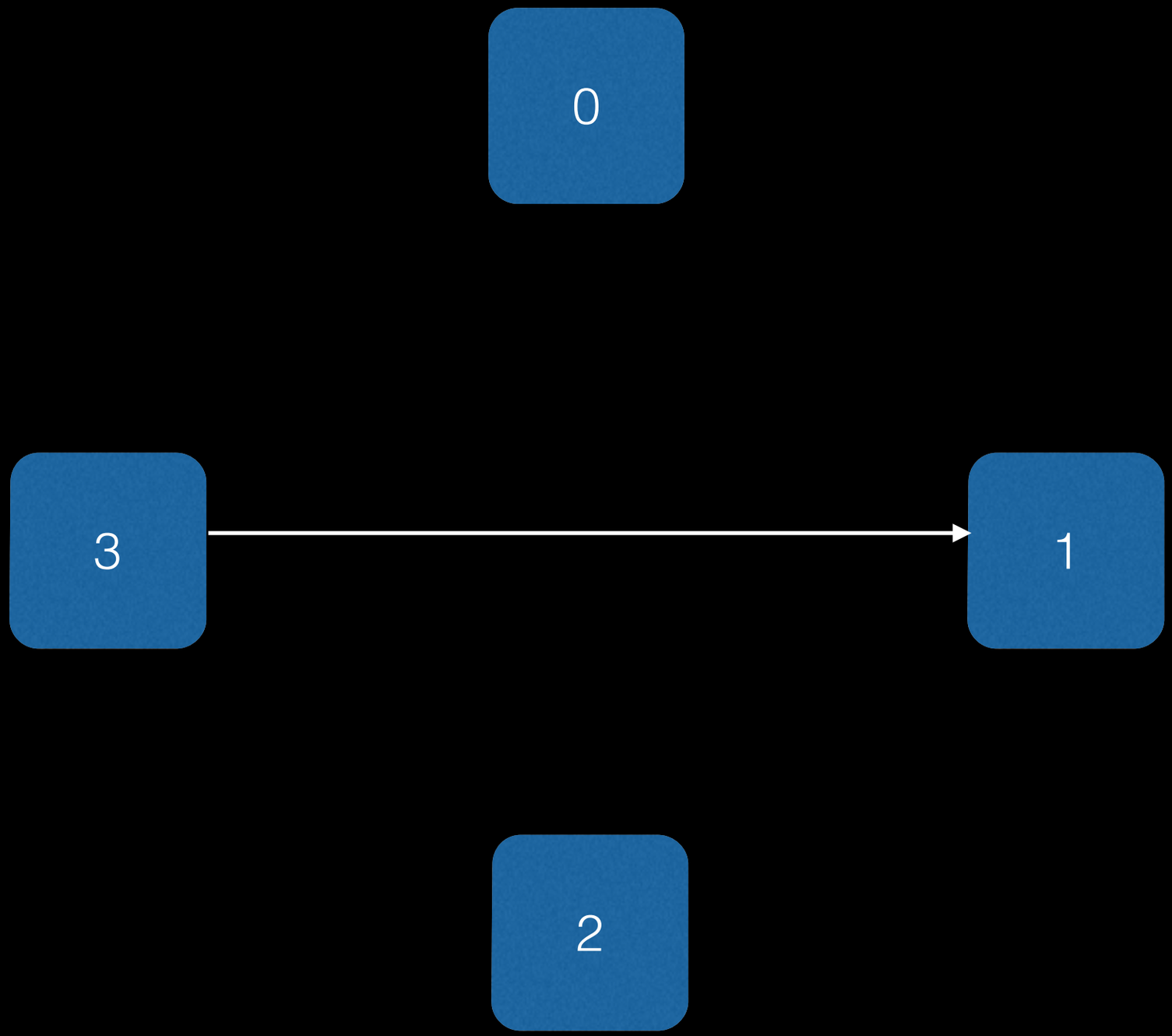
0 envia mensagens a 1, 2 e 3



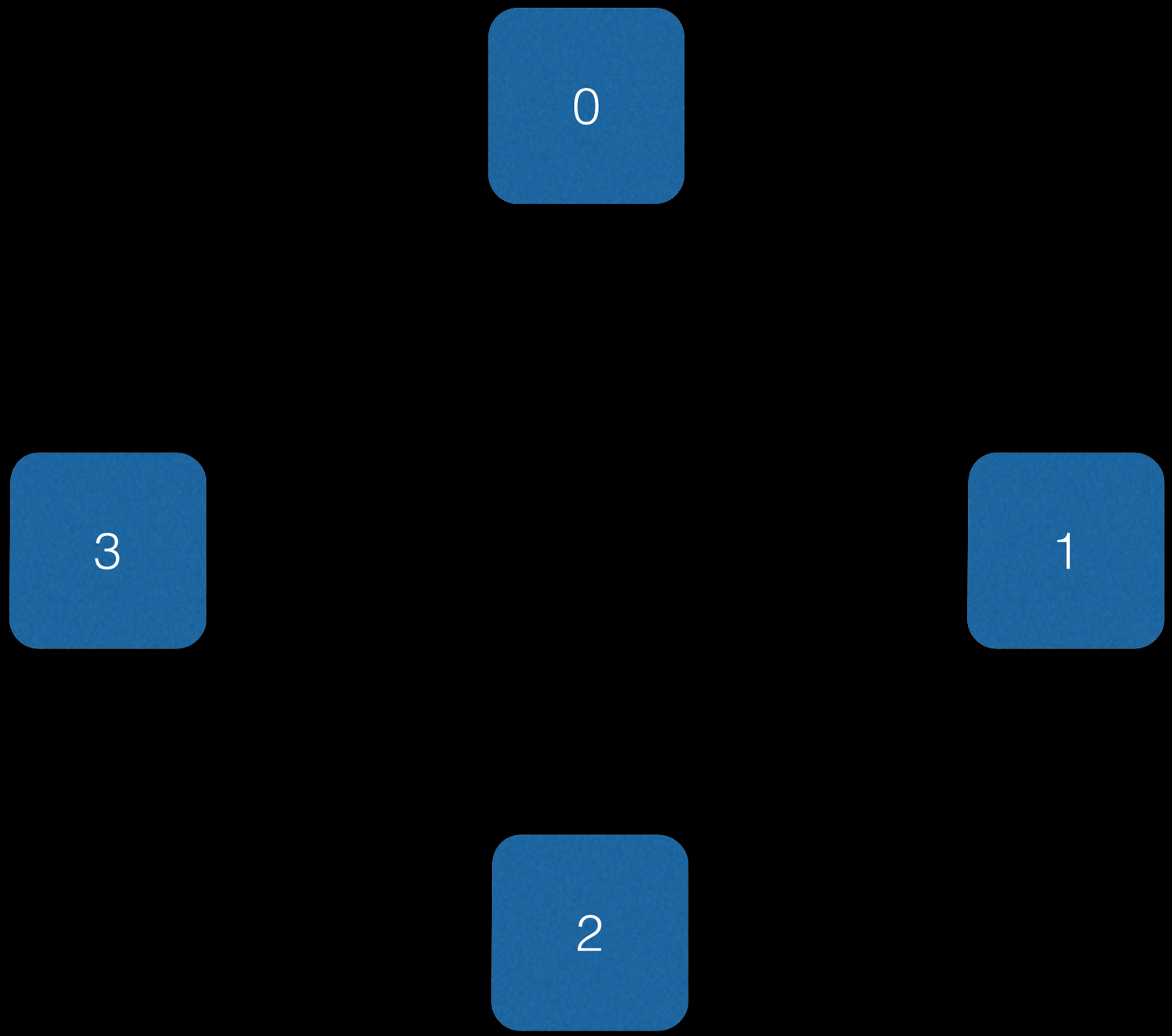
1, 2 e 3 realizam computações



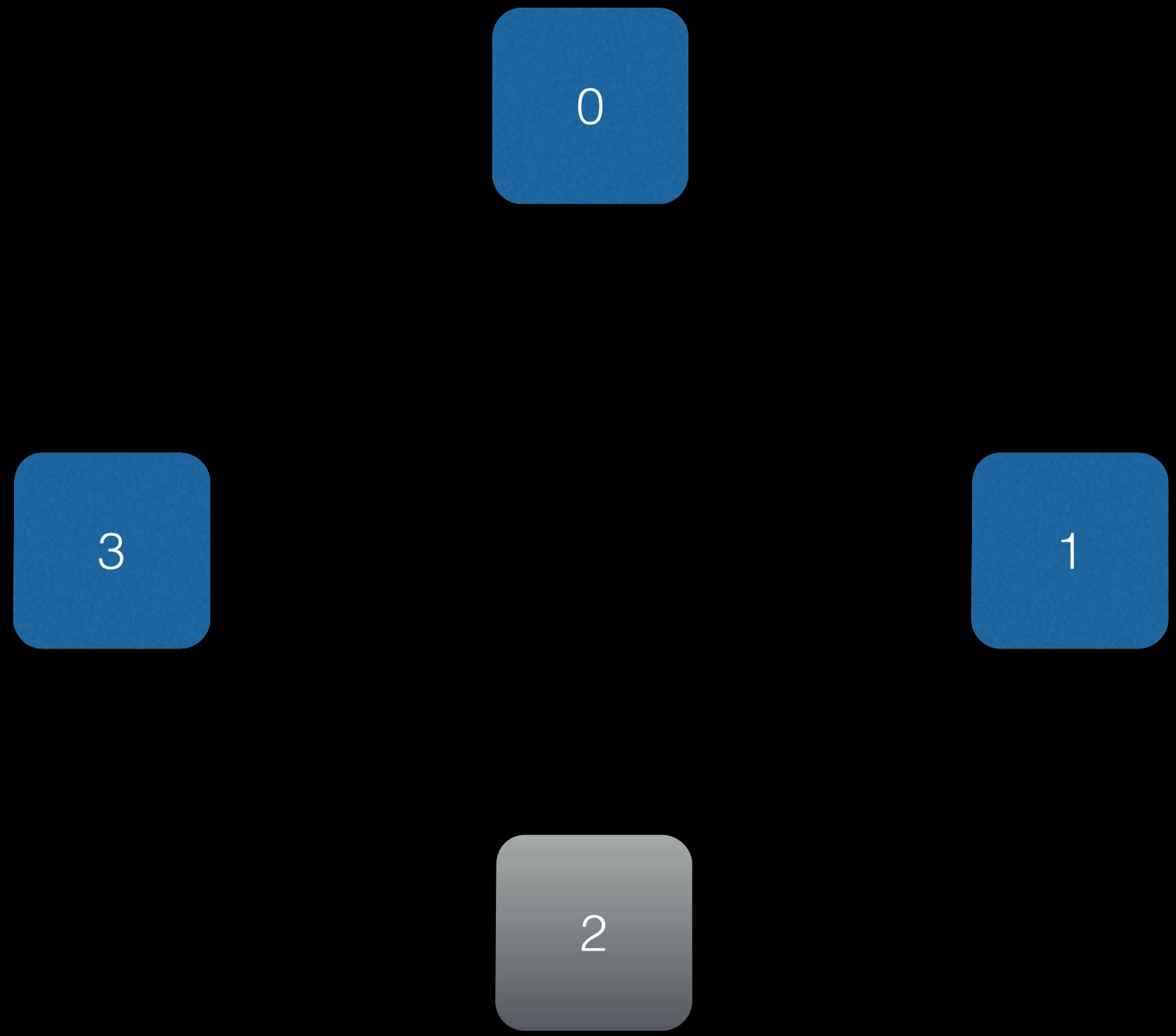
3 envia dados para 1



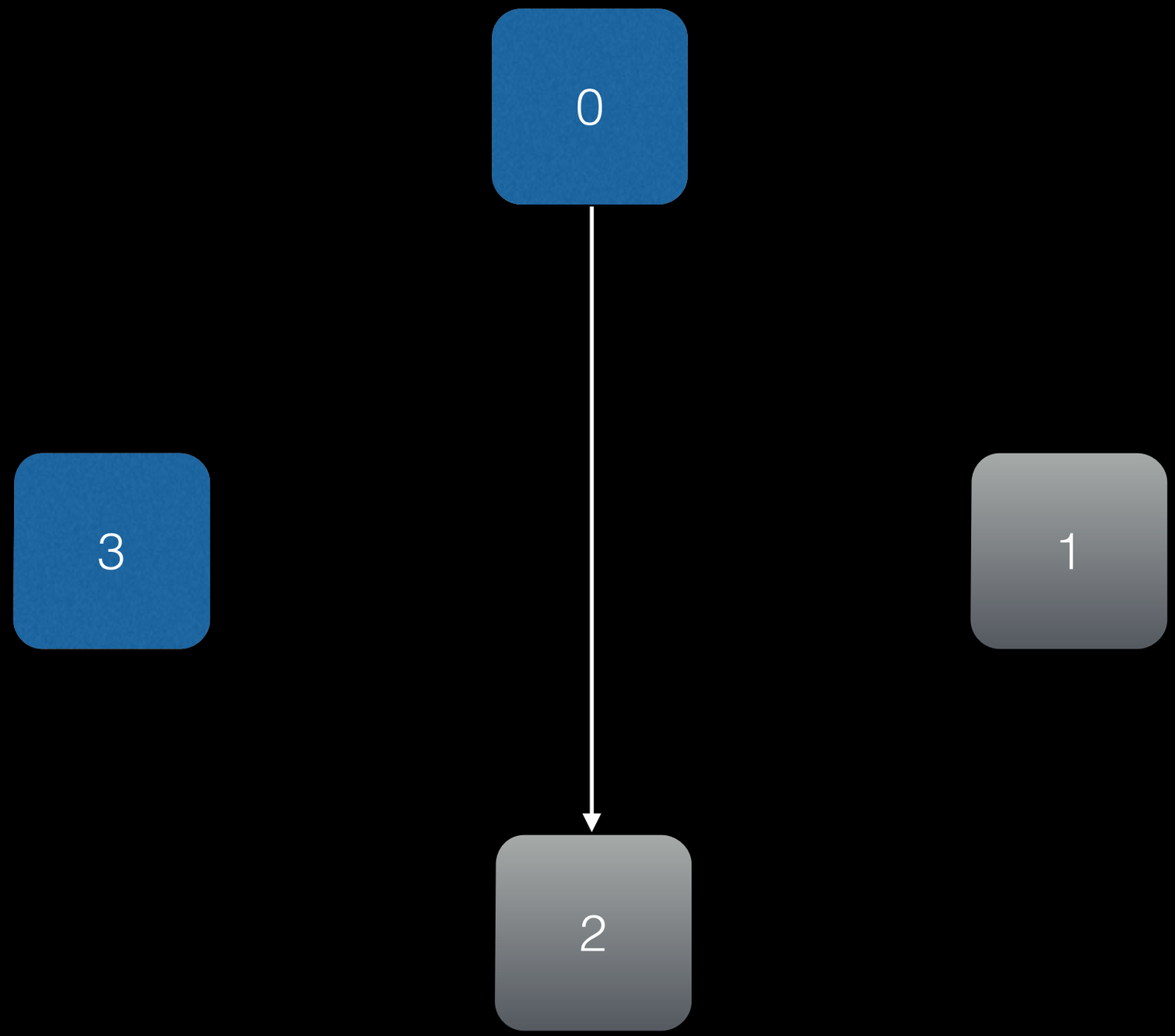
atividade prossegue



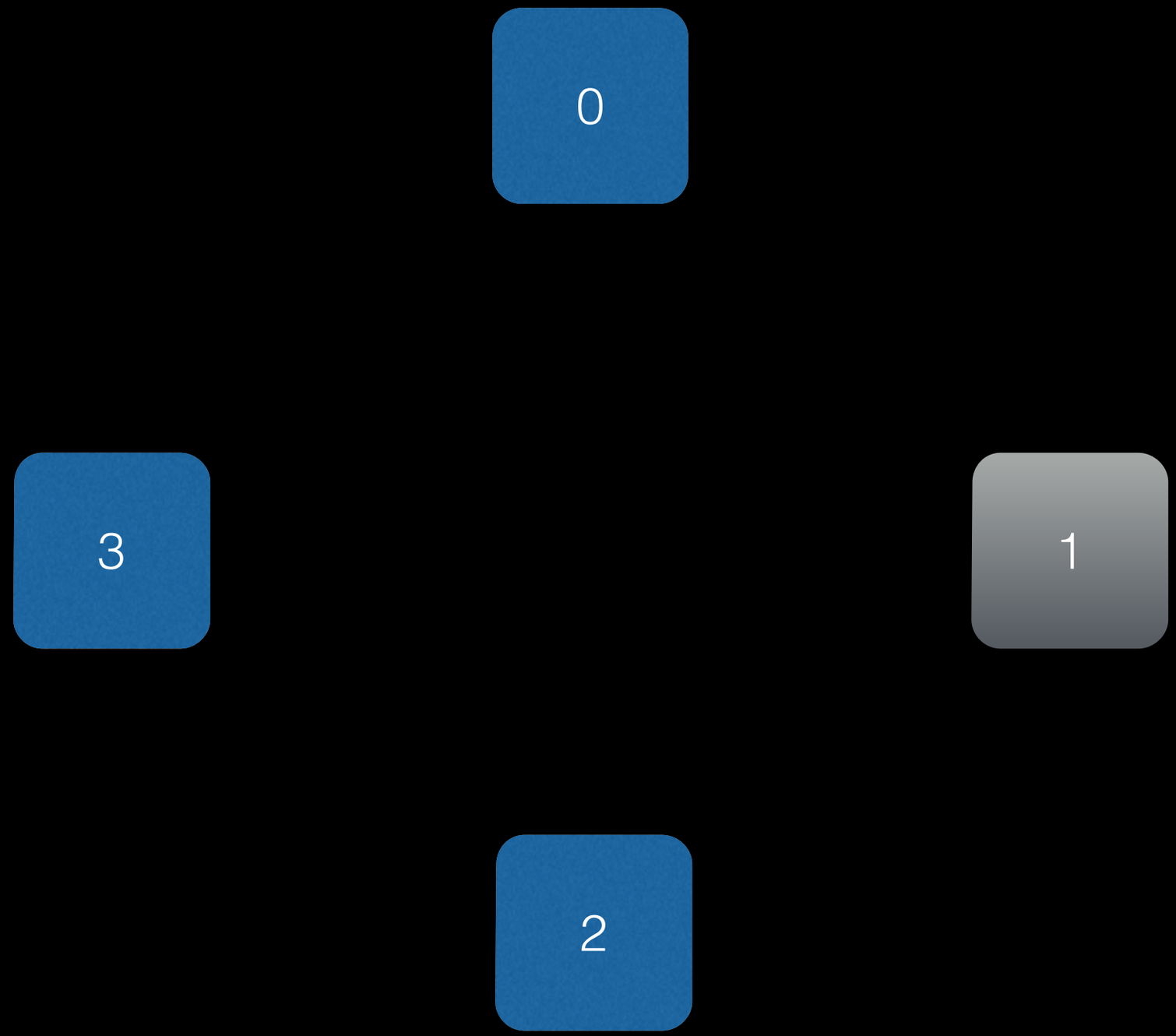
2 termina sua computação, se torna ocioso



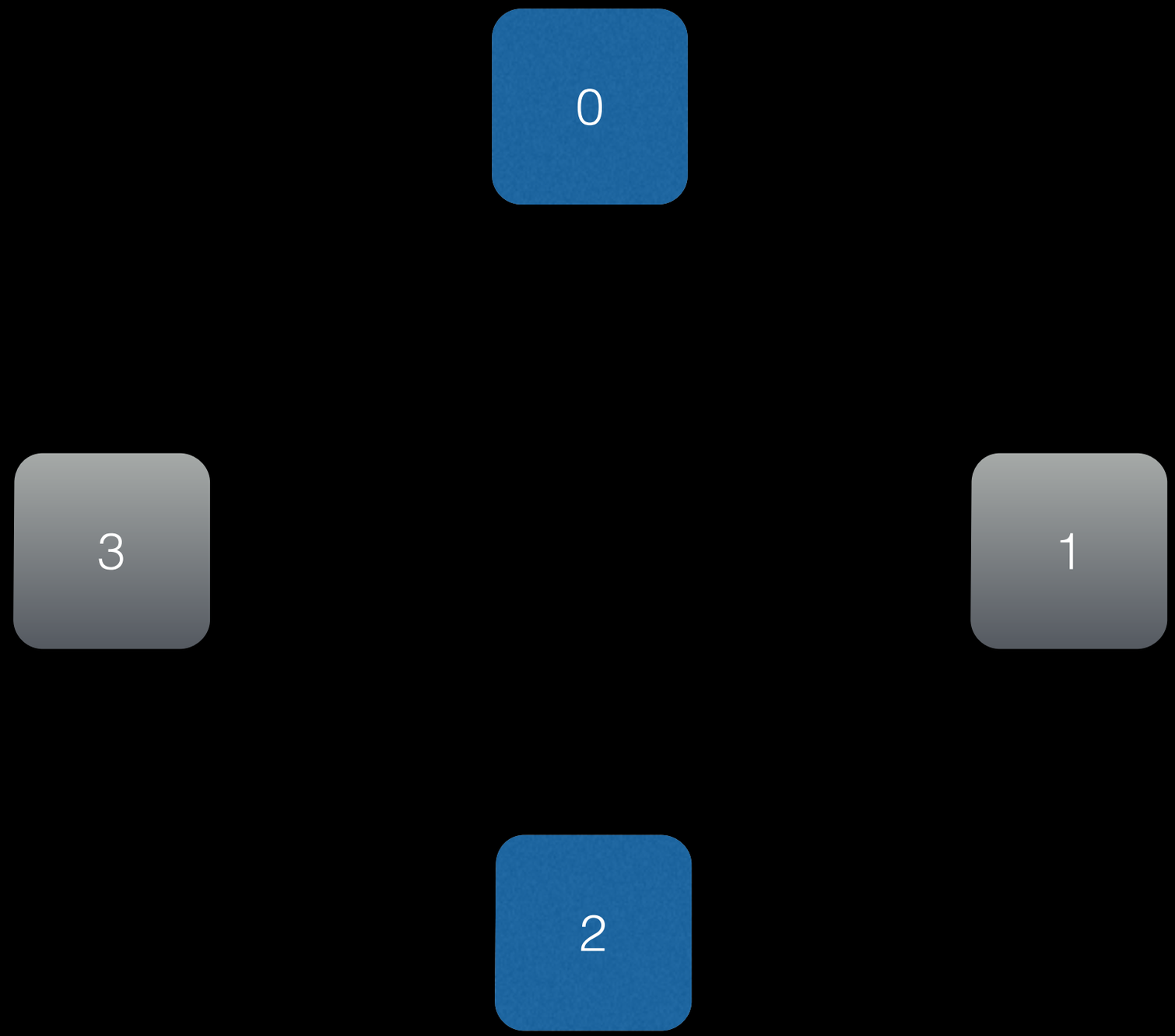
1 torna-se ocioso, e 0 envia dados para 2



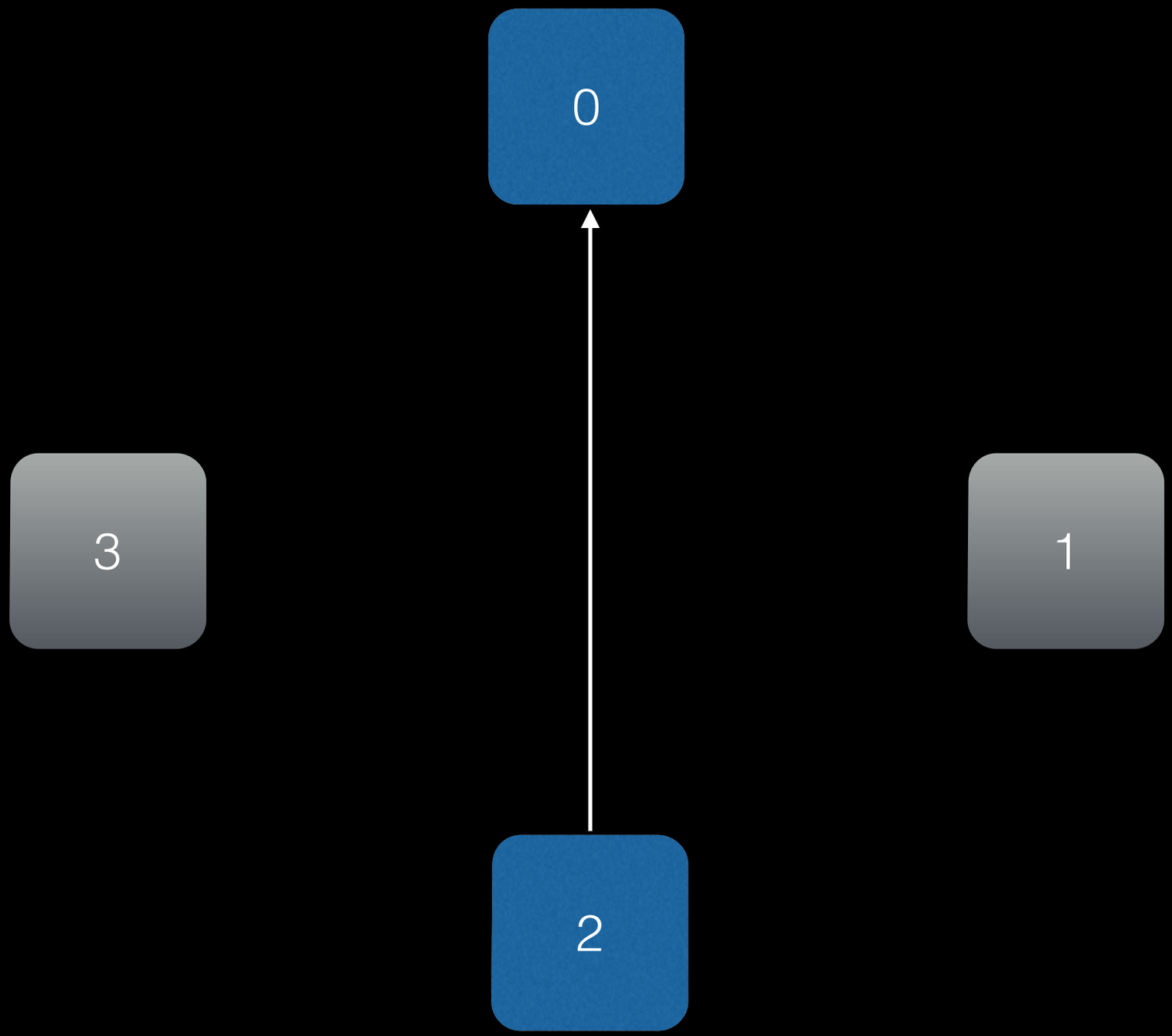
ao receber mensagem, 2 volta a ser ativo



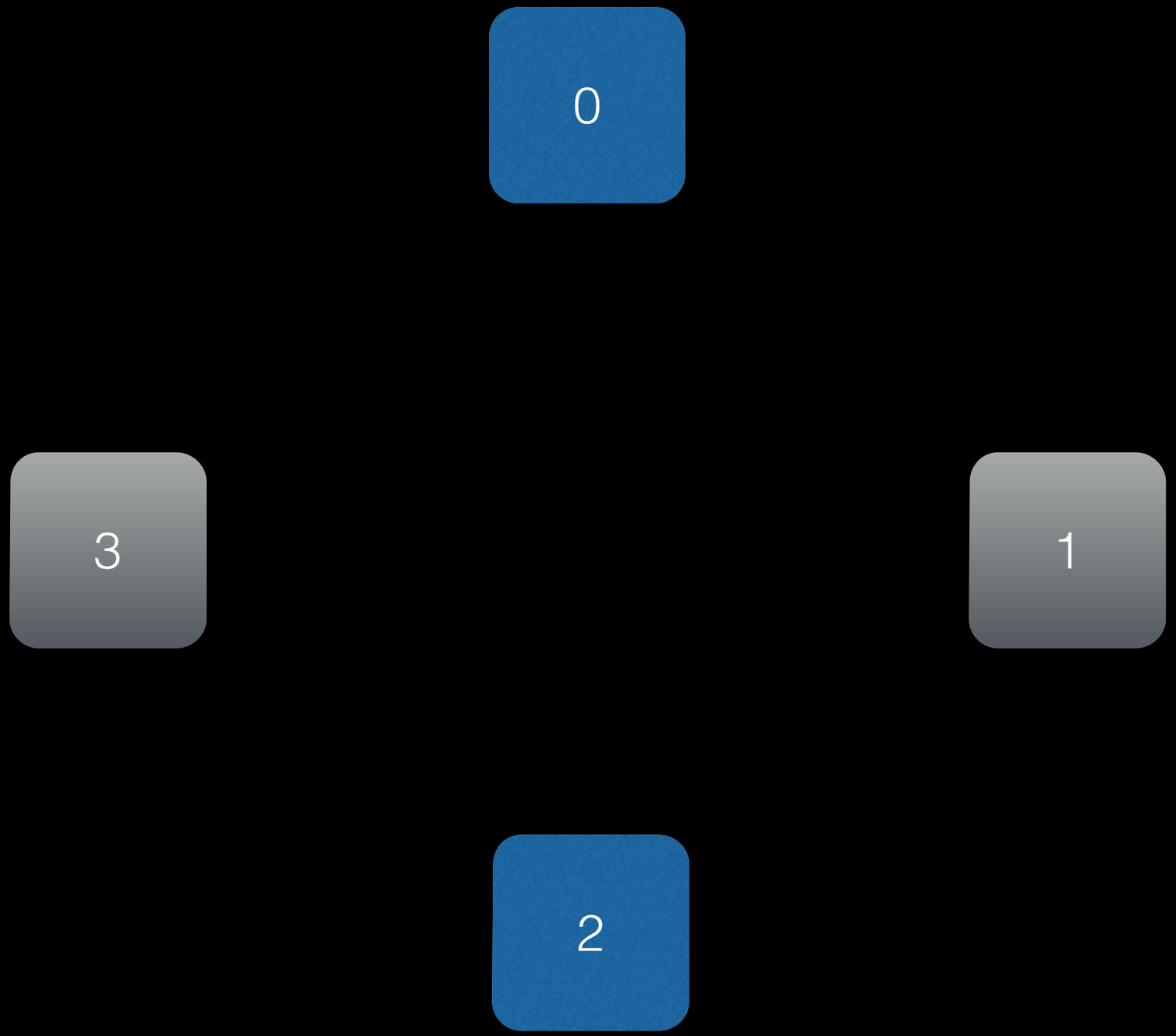
3 torna-se ocioso



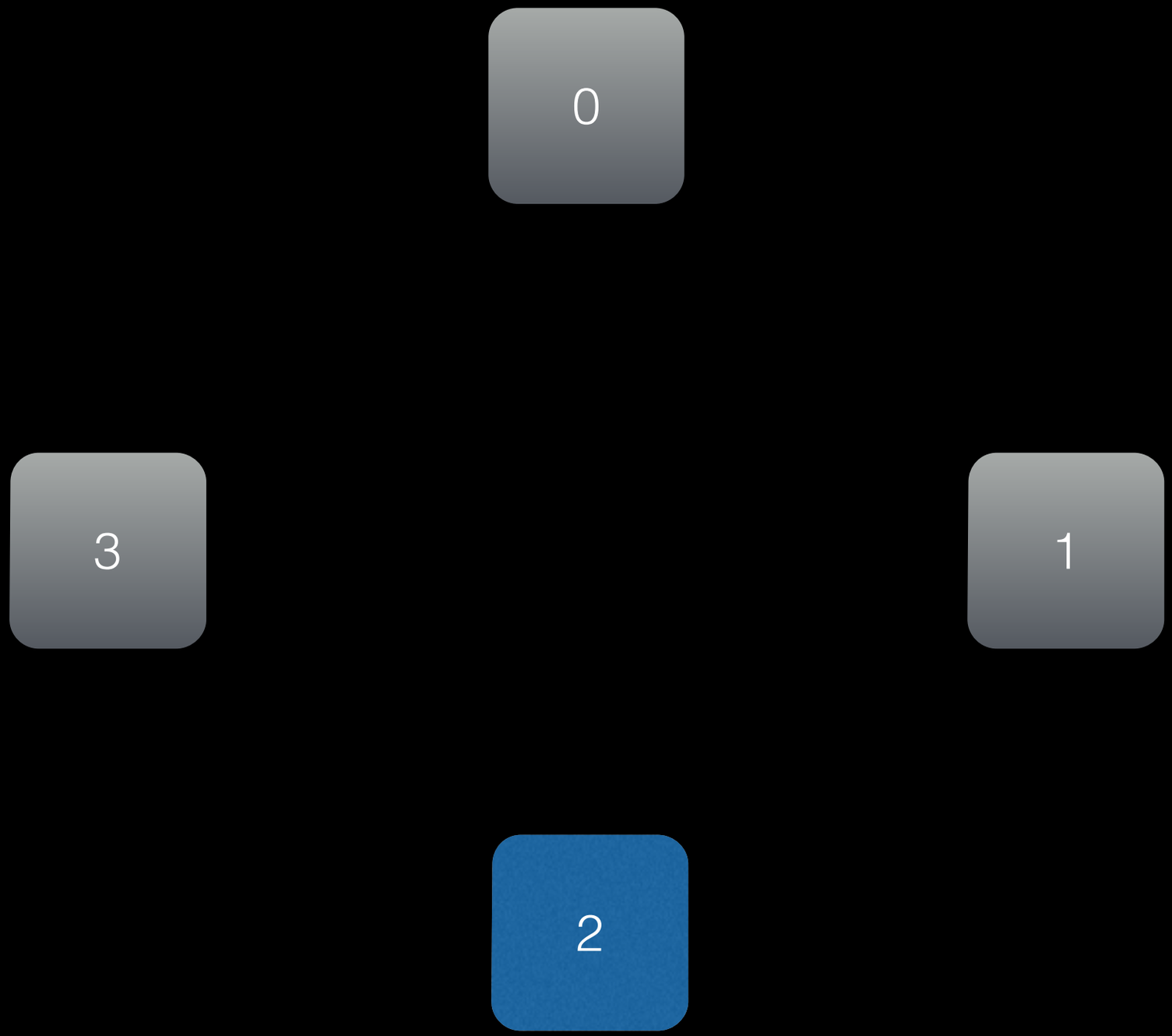
2 envia dados para 0



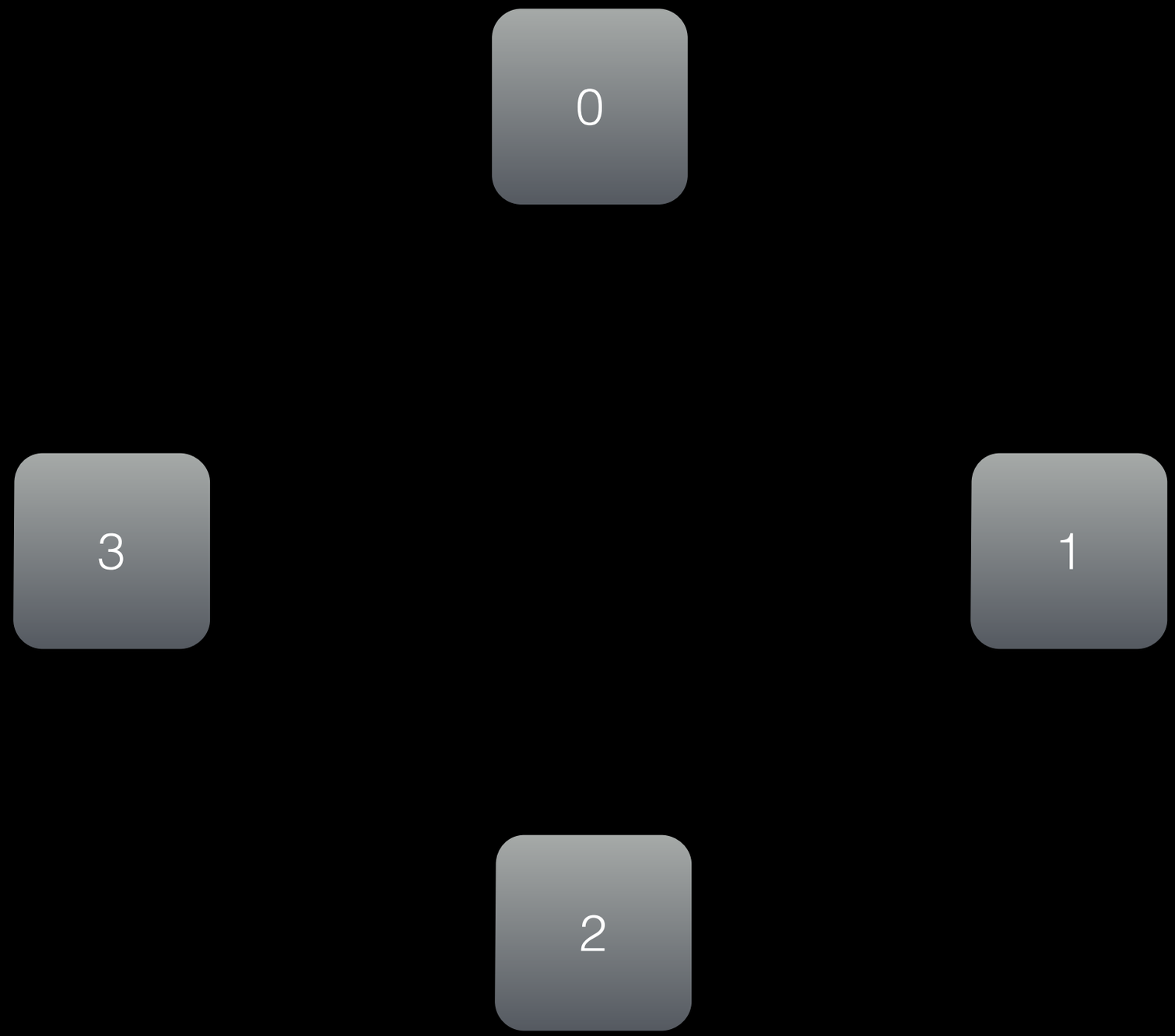
Processadores: 0, 1, 2 e 3



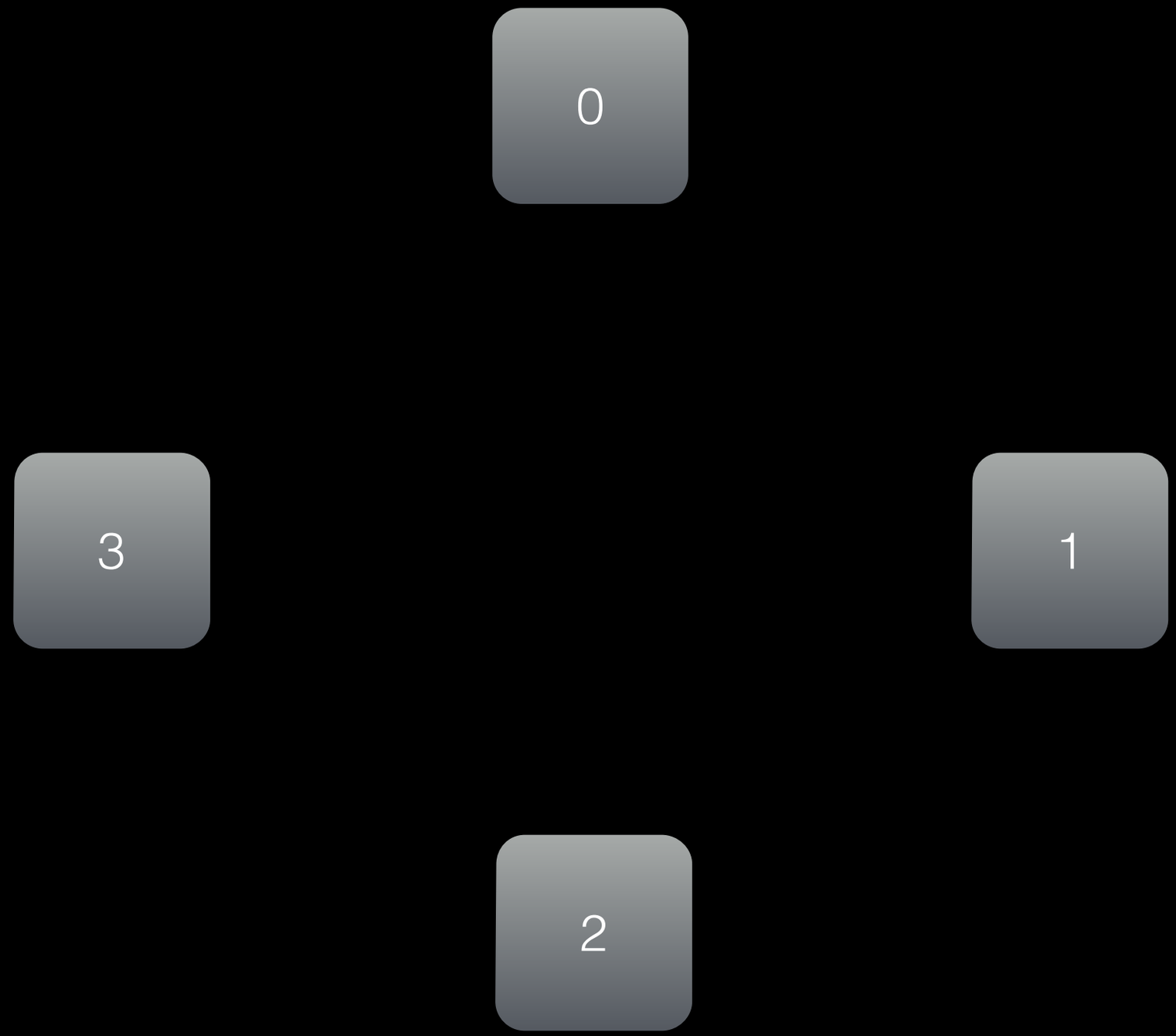
0 torna-se ocioso



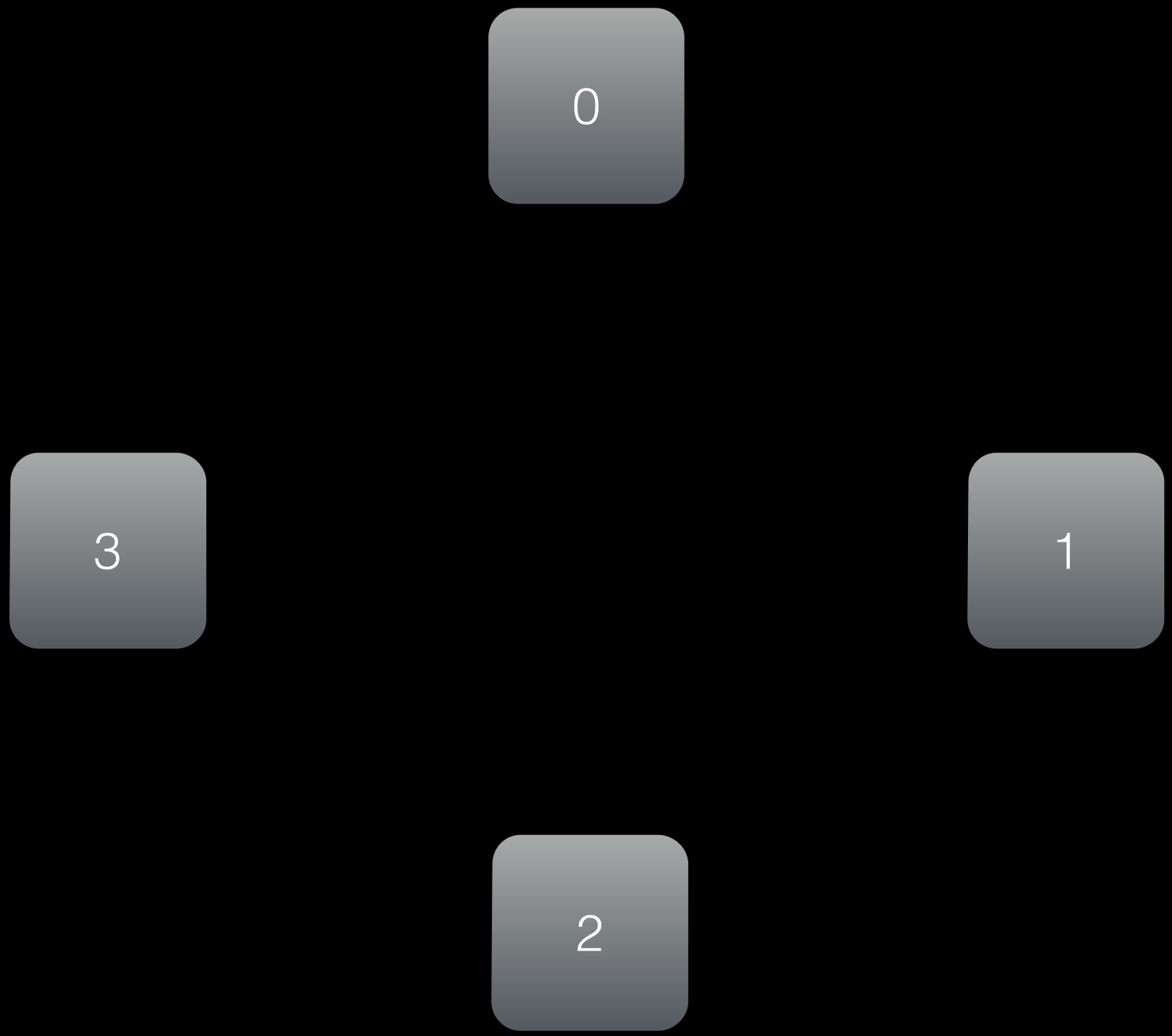
2 torna-se ocioso



Como 0 pode saber que os 1 2 e 3 são ociosos



para informar o término da computação?



Solução EWD 840

- Edsger W. Dijkstra, W.H.J. Feijen, A.J.M. van Gasteren
- Derivation of a termination detection algorithm for distributed computations

Modelo B

- N processadores [#0, #1, #2, … #(N-1)]

Modelo B

MACHINE EWD840

CONSTANTS NUM

PROPERTIES NUM ∈ NAT1

DEFINITIONS PROC == 0 .. (NUM-1)

- N processadores [#0, #1, #2, … #(N-1)]

Modelo B

- Estado dos processadores
  - ativo
  - ocioso

Modelo B

- Estado dos processadores
  - ativo
  - ocioso

VARIABLES passive
INVARIANT
passive ⊆ PROC
INITIALISATION
passive :∈ $\mathcal{P}$(PROC)

Modelo B

- Detectar término

Modelo B

- Detectar término

VARIABLES terminated
INVARIANT
terminated ∈ BOOL
INITIALISATION
terminated := FALSE

Modelo B

- Requisito funcional:
  - término quando todos os processadores estão ociosos

Modelo B

INVARIANT
terminated = TRUE ⇒ passive = PROC

- Requisito funcional:
  - término quando todos os processadores estão ociosos

Abordagem

- Uma ficha circula entre os processadores
- Passa ao sucessor quando terminou
- Hipóteses:
  - Comunicação instantânea
  - #0 → #(N-1) … #2 →#1 →#0

Abordagem

- Uma ficha circula entre os processadores
- Passa ao sucessor quando terminou
- Hipóteses:
  - Comunicação instantânea
  - #0 → #(N-1) … #2 →#1 →#0

VARIABLES token
INVARIANT token ∈ PROC
INITIALISATION token := 0

Abordagem

- Propriedade desejada:
  - Todos os processa… cima da ficha estão ociosos

mas… e se uma mensagem chegar em P depois da ficha ter saido ?

Abordagem

INVARIANT token+1..num ⊆ passive

- Propriedade desejada:
  - Todos os processa… cima da ficha estão ociosos

mas… e se uma mensagem chegar em P depois da ficha ter saido ?

Soluções

- mas… e se uma mensagem chegar em P depois da ficha ter saido ?
  - A ficha já passou em P,
    - Quem enviou a ficha deve ser ativo 😎
    - Então está entre 0 e token
    - Pode “marcar” a ficha para indicar que mais uma rodada será necessária

Solução

SETS
COLOR = { BLACK, WHITE }
VARIABLE
token_color
INVARIANT
token_color ∈ COLOR ∧
(token+1..num ⊆ passive ∨ token_color = BLACK)
INITIALISATION
color_token := WHITE

- ma P depois da
  - A

- Quem enviou a ficha deve ser ativo 😎
- Então está entre 0 e token
- Pode “marcar” a ficha para indicar que mais uma rodada será necessária

Solução

- ma P depois da
  - A
- Quem enviou a ficha d
- Então está entre 0 e tok
- Pode “marcar” a ficha para indicar que mais uma rodada será necessária

```
SETS
  COLOR = { BLACK, WHITE }
VARIABLE
  token_color
INVARIANT
  token_color ∈ COLOR ∧
  (token+1..num ⊆ passive ∨ token_color = BLACK)
INITIALISATION
  color_token := WHITE
```

mas… e se o emissor da mensagem não está com a ficha?

Soluções

- mas… e se o emissor da mensagem não está com a ficha?A ficha já
  - O emissor está entre 0 e token
    - Quando um processador envia uma mensagem para um outro com índice maior, ele ativa um marcador.
    - Quando um processador está com um marcador ativo, ele atribui a cor preto à ficha.

Soluções

- mas… e se o m não está com a ficha?A

  - O emissor e

VARIABLES
tainted
INVARIANT
tainted ⊆ PROC ∧
(token+1..num ⊆ passive ∨
tainted ∩ 0..token ≠ ∅ ∨
token_color = BLACK)
INITIALISATION
tainted := ∅

  - Quando um ma mensagem para um outro com índice maior, ele ativa um marcador.

  - Quando um processador está com um marcador ativo, ele atribui a cor preto à ficha.

Síntese parcial

Síntese parcial

MACHINE EWD840
CONSTANTS NUM
PROPERTIES NUM ∈ NAT1
SETS
  COLOR = { BLACK, WHITE }
DEFINITIONS PROC == 0 .. (NUM-1)
VARIABLES
  passive, terminated, token, tainted, token_color
INVARIANT
  passive ⊆ PROC ∧ terminated ∈ BOOL ∧ token ∈ PROC ∧
  tainted ⊆ PROC ∧ token_color ∈ COLOR ∧
 (token+1..num ⊆ passive ∨
   tainted ∩ 0..token ≠ ∅ ∨
   token_color = BLACK)
INITIALISATION
  passive :∈ $\mathcal{P}$(PROC) || terminated := FALSE || token := 0 ||
color_token := WHITE || tainted := ∅

Modelo do comportamento

- Um processador ativo pode tornar-se ocioso a qualquer momento.

Modelo do comportamento

- Um processador ativo pode tornar-se ocioso a qualquer momento.

```
Finish(pr) =
PRE pr ∈ PROC ∧ pr ∉ passive THEN
  passive := passive ∪ {pr}
END;
```

Modelo do comportamento

- Um processador ativo PRi pode mandar uma mensagem para um processador PRj.

Modelo do comportamento

```
Send_Message(pri,prj) =
PRE pri ∈ PROC ∧ prj ∈ PROC ∧ pri ∉ passive THEN
  IF prj ∈ passive THEN
    passive := passive - {prj}
  END ||
  IF prj > pri THEN
    tainted := tainted ∪ {pri}
  END
END
```

- U
n

Modelo do comportamento

- Quando um processador torna-se ocioso e possui uma ficha, esta passa ao processador seguinte.

- O processador 0 tem um comportamento especial (reinicia a cor do TOKEN, etc.)

Modelo do comportamento

- ```
  Pass_Token = PRE token ≠ 0 ∧ token ∈ passive THEN
    token := token - 1 ||
    IF token ∈ tainted THEN
      color_token := BLACK
    END ||
    tainted := tainted - { token }
  END
  ```
- O processador 0 tem um comportamento especial (reinicia a cor do TOKEN, etc.)
  ```

Modelo do comportamento

- O processador 0:
  - reinicia a cor da ficha
  - passa a ficha ao processador de índice mais elevado

Modelo do comportamento

Initiate_Probe = **PRE** token = 0 ∧ color_token = BLACK **THEN**
  tainted := tainted - {0} ||
  color_token := WHITE ||
  token := NUM - 1
END

- passa a ficha ao processador de índice mais elevado

Modelo do comportamento

- O processador 0:
  - reinicia a cor da ficha
  - passa a ficha ao processador de índice mais elevado

Modelo do comportamento

Terminated = **PRE** token = 0 ∧ color_token = WHITE ∧ 0 ∉ tainted ∧ 0 ∈ passive
THEN
terminated := TRUE
END

- passa a ficha ao processador de índice mais elevado

Próximos passos

- Verificação de sintaxe
- Verificação de tipos
- Verificar que a especificação é consistente
- Simular o comportamento de uma instância da especificação (NUM = 3)
- Verificar ausência de inter-travamento.

Próxim

- Verifi
- Verificar que a execução é consistente
- Simular o travamento de uma instância da es
- Verificar ausência de inter-travamento.

Próximos passos

- Projetar implementação
- O processador 0 é diferente dos demais
- Dois módulos diferentes devem ser especificados
  - Leader
  - Processor
- Cada módulo pode ser implantado isoladamente

Próxim

- Projet
- Cada módulo pode ser implantado isoladamente