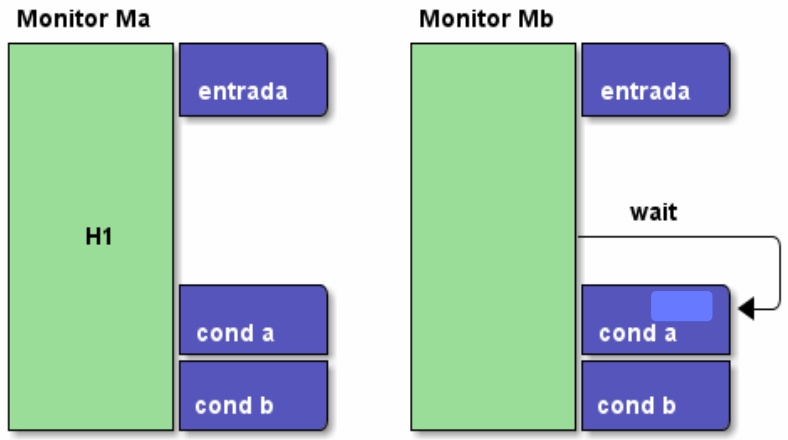
Monitor.- invocaciones cruzadas

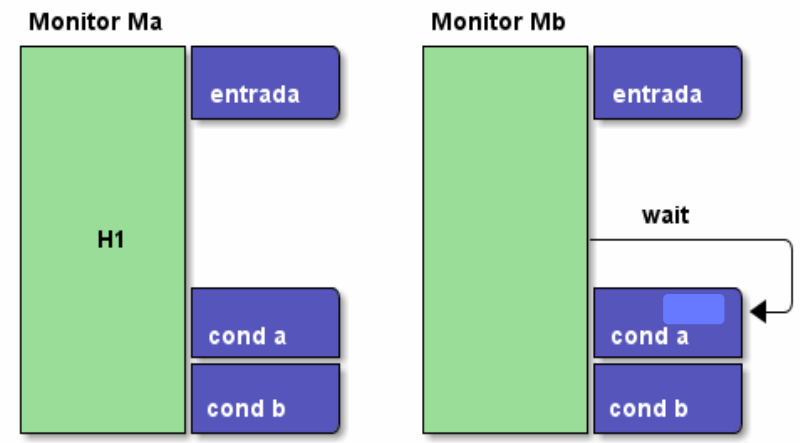
- Invocar desde un monitor a un método de otro monitor puede:
  - reducir la concurrencia
  - incluso provocar interbloqueos



- Ejemplo: Suponemos
  - 2 monitores Ma y Mb:
    - desde un método de Ma se invoca un método de Mb y viceversa
  - 2 hilos H1 y H2

Monitor.- invocaciones cruzadas

- H1 activo en Ma, invoca un método de Mb, dentro del cual ejecuta **a.wait()**
  - Pasa a la cola de espera "a" del monitor Mb
  - Libera el monitor Mb, pero no Ma
    - Nadie puede usar Ma → reducimos concurrencia
- Si H2 entra en Mb (que estaba libre) e invoca un método del monitor Ma
  - Espera en la cola de entrada de Ma (Ma está ocupado)
  - No deja libre el monitor Mb
  - Hemos llegado a un **interbloqueo**



Invocaciones cruzadas.- Ejemplo 1 de interbloqueo

```
public class Problema {
  public synchronized void hola() {...}
  public synchronized void test (Problema x) { x.hola(); }
}
```

- Dos hilos H1, H2
- Dos objetos `Problema p1, p2`

H1 { p1.test(p2) }
H2 { p2.test(p1) }

¿Qué ocurre aquí?

Invocaciones cruzadas.- Ejemplo 2 de interbloqueo

- Definimos dos monitores (p,q) tipo Bcell
- Suponemos 2 hilos concurrentes H1 y H2
  - H1 invoca ***p.swap(q)***, obtiene acceso al monitor "p", e inicia la ejecución de *p.swap*
  - H2 invoca ***q.swap(p)***, obtiene acceso al monitor "q", e inicia la ejecución de *q.swap*

¿Qué ocurre aquí?

```
class BCell {
  int value;
  public synchronized void getValue() {
    return value;
  }
  public synchronized void setValue(int i) {
    value=i;
  }
  public synchronized void swap(BCell x) {
    int temp= getValue();
      setValue(x.getValue());
      x.setValue(temp);
  }
}
```

Invocaciones cruzadas.- Ejemplo 2 de interbloqueo

- **Aparece un interbloqueo**
  - Dentro de ***p.swap***, H1 invoca ***q.getValue()***, pero debe esperar porque el monitor q no está libre
  - Dentro de ***q.swap***, H2 invoca ***p.getValue()***, pero debe esperar porque el monitor p no está libre
  - Ambos se esperan mutuamente, y la situación no puede evolucionar → **INTERBLOQUEO**

```
class BCell {
  int value;
  public synchronized void getValue() {
    return value;
  }
  public synchronized void setValue(int i) {
    value=i;
  }
  public synchronized void swap(BCell x) {
    int temp= getValue();
      setValue(x.getValue());
      x.setValue(temp);
  }
}
```