Tema 11: Aplicaciones de SBC: Sistemas de control

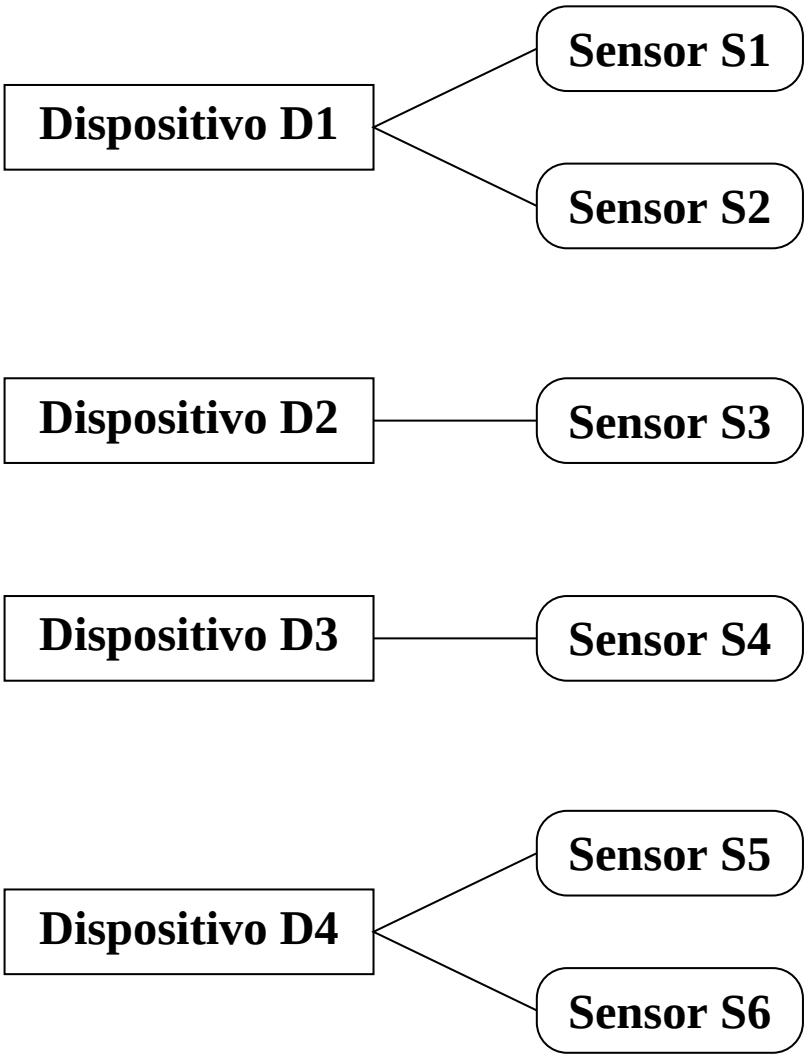
José A. Alonso Jiménez
Miguel A. Gutiérrez Naranjo
Francisco J. Martín Mateos

Dpto. de Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Sistemas de control

- Sistema formado por 4 dispositivos con sensores asociados:



- Giarratano, J.C. y Riley, G. "Expert Systems Principles and Programming (2nd ed.)" (PWS Pub. Co., 1994)
  * Cap. 12.5: "A Monitoring Problem"

Sistemas de control

- **Sensor:**
  - **Límite Crítico Alto**
  - **Límite Peligroso Alto**
  - **Límite Peligroso Bajo**
  - **Límite Crítico Bajo**
- **Límites asociados a los sensores:**

Sensor	LCA	LPA	LPB	LCB	N
S1	130	120	70	60	3c
S2	180	160	40	20	5c
S3	130	120	70	60	4c
S4	130	120	70	60	4c
S5	125	120	70	65	4c
S6	130	125	115	110	2c

- **Dispositivos:**
  - **Estado crítico: Sensor asociado con Valor $\geq$ LCA o Valor $\leq$ LCB. Desconectar dispositivo**
  - **Estado peligroso: Sensor asociado con LCA $>$ Valor $\geq$ LPA o LPB $\geq$ Valor $>$ LCB. Descontectar dispositivo si la situación se mantiene durante N ciclos.**

Sistemas de control

- **Ciclo:**
  - **Leer datos de entrada**
  - **Detectar estados de los dispositivos**
  - **Realizar acciones**
- **Fuente de datos:**
  - **Hechos**
  - **Sensores**
  - **Usuario**
  - **Fichero**
- **Final:**
  - **Todos los dispositivos desconectados**
  - **Límite de ciclos**
- **Valores de los sensores por ciclos:**

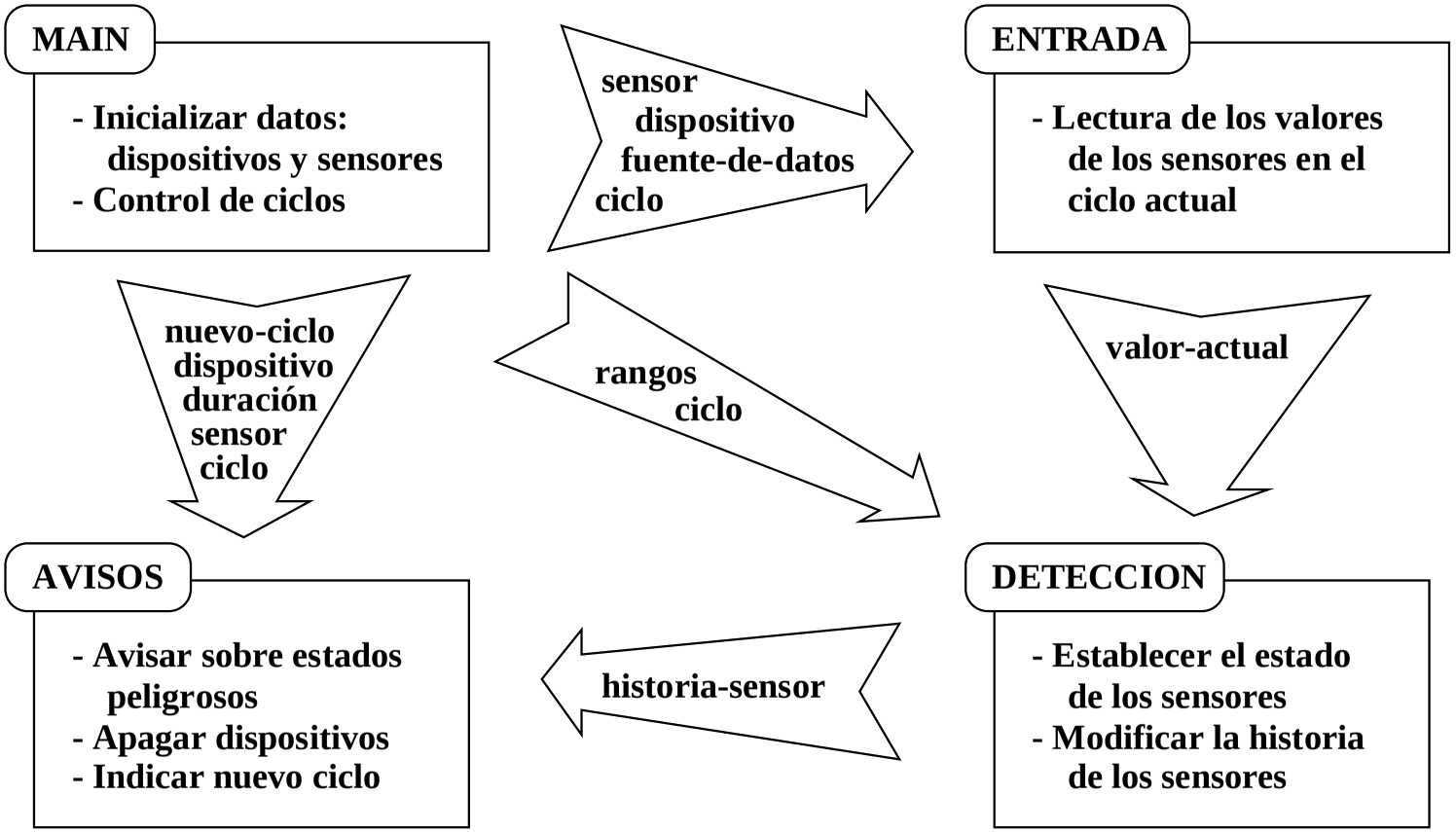
Sensor	Ciclo1	Ciclo2	Ciclo3	Ciclo4	Ciclo5	Ciclo6
S1	100	100	110	110	115	120
S2	110	120	125	130	130	135
S3	100	120	125	130	130	125
S4	120	120	120	125	130	135
S5	110	120	125	130	135	135
S6	115	120	125	135	130	135

Sistemas de control

- **Sesión**

```
CLIPS> (load "monitor.clp")
...
CLIPS> (run)
Ciclo 1 - Sensor S6 en peligroso-bajo  durante 1 ciclos.
Ciclo 1 - Sensor S4 en peligroso-alto  durante 1 ciclos.
Ciclo 2 - Sensor S5 en peligroso-alto  durante 1 ciclos.
Ciclo 2 - Sensor S4 en peligroso-alto  durante 2 ciclos.
Ciclo 2 - Sensor S3 en peligroso-alto  durante 1 ciclos.
Ciclo 3 - Sensor S6 en peligroso-alto  durante 1 ciclos.
Ciclo 3 - Sensor S5 en critico-alto.
  Apagar el dispositivo D4.
Ciclo 3 - Sensor S4 en peligroso-alto  durante 3 ciclos.
Ciclo 3 - Sensor S3 en peligroso-alto  durante 2 ciclos.
Ciclo 4 - Sensor S4 en peligroso-alto  durante 4 ciclos.
  Apagar el dispositivo D3.
Ciclo 4 - Sensor S3 en critico-alto.
  Apagar el dispositivo D2.
Ciclo 6 - Sensor S1 en peligroso-alto  durante 1 ciclos.
```

Esquema del sistema de control



Sistemas de control

- **Módulo MAIN**

```
(defmodule MAIN
  (export deftemplate ?ALL))

(deftemplate MAIN::dispositivo
  (slot nombre)
  (slot estado
    (default activo)))

(deffacts MAIN::informacion-dispositivos
  (dispositivo (nombre D1))
  (dispositivo (nombre D2))
  (dispositivo (nombre D3))
  (dispositivo (nombre D4)))

(deftemplate MAIN::sensor
  (slot nombre)
  (slot dispositivo-asociado))

(deffacts MAIN::informacion-sensores
  (sensor (nombre S1) (dispositivo-asociado D1))
  (sensor (nombre S2) (dispositivo-asociado D1))
  (sensor (nombre S3) (dispositivo-asociado D2))
  (sensor (nombre S4) (dispositivo-asociado D3))
  (sensor (nombre S5) (dispositivo-asociado D4))
  (sensor (nombre S6) (dispositivo-asociado D4)))
```

Sistemas de control

```
(deftemplate MAIN::rangos
  (slot nombre)
  (slot minimo-critico) (slot minimo-peligroso)
  (slot maximo-peligroso) (slot maximo-critico))

(deffacts MAIN::rangos-de-los-sensores
  (rangos (nombre S1) (minimo-critico 60)
          (minimo-peligroso 70)
          (maximo-peligroso 120)
          (maximo-critico 130))
  (rangos (nombre S2) (minimo-critico 20)
          (minimo-peligroso 40)
          (maximo-peligroso 160)
          (maximo-critico 180))
  (rangos (nombre S3) (minimo-critico 60)
          (minimo-peligroso 70)
          (maximo-peligroso 120)
          (maximo-critico 130))
  (rangos (nombre S4) (minimo-critico 60)
          (minimo-peligroso 70)
          (maximo-peligroso 120)
          (maximo-critico 130))
  (rangos (nombre S5) (minimo-critico 65)
          (minimo-peligroso 70)
          (maximo-peligroso 120)
          (maximo-critico 125))
  (rangos (nombre S6) (minimo-critico 110)
          (minimo-peligroso 115)
          (maximo-peligroso 125)
          (maximo-critico 130)))
```

Sistemas de control

```
(deftemplate MAIN::duracion
  (slot sensor)
  (slot tiempo))

(deffacts MAIN::duracion-sensores-en-estado-peligroso
  (duracion (sensor S1) (tiempo 3))
  (duracion (sensor S2) (tiempo 5))
  (duracion (sensor S3) (tiempo 4))
  (duracion (sensor S4) (tiempo 4))
  (duracion (sensor S5) (tiempo 4))
  (duracion (sensor S6) (tiempo 2)))

(deffacts MAIN::comienzo
  (fuente-de-datos hechos)
  (ciclo 0)
  (nuevo-ciclo 1))
```

Sistemas de control

```
;;; REGLA: control
;;; SI
;;;   estamos en un ciclo y
;;;   ya ha sido insertado el nuevo ciclo y
;;;   el nuevo ciclo es menor que 7
;;; ENTONCES
;;;   insertamos el nuevo ciclo como ciclo actual y
;;;   pasamos el control a los restantes módulos en
;;;     el orden correcto.

(defrule MAIN::control
  ?h1 <- (ciclo ?)
  ?h2 <- (nuevo-ciclo ?nciclo&:(< ?nciclo 7))
  =>
  (retract ?h1 ?h2)
  (assert (ciclo ?nciclo))
  (focus ENTRADA DETECCION AVISOS))
```

Sistemas de control

• Módulo ENTRADA

```
(defmodule ENTRADA
  (import MAIN deftemplate dispositivo
                           sensor
                           ciclo
                           fuente-de-datos)
  (export deftemplate valor-actual))

(deftemplate ENTRADA::valor-actual
  (slot sensor)
  (slot valor)
  (slot ciclo))

(deffacts ENTRADA::hechos-de-valores
  (hecho-de-datos S1 100 100 110 110 115 120)
  (hecho-de-datos S2 110 120 125 130 130 135)
  (hecho-de-datos S3 100 120 125 130 130 125)
  (hecho-de-datos S4 120 120 120 125 130 135)
  (hecho-de-datos S5 110 120 125 130 135 135)
  (hecho-de-datos S6 115 120 125 135 130 135))
```

Sistemas de control

```
;;; REGLA: no-hay-dispositivos-encendidos
;;; SI
;;;   no hay dispositivos encendidos
;;; ENTONCES
;;;   lo indica en pantalla y
;;;   detiene la ejecución.

(defrule ENTRADA::no-hay-dispositivos-encendidos
  (not (dispositivo (estado activo)))
  =>
  (printout t "Todos los dispositivos "
            "estan apagados. " crlf)
  (halt))
```

Sistemas de control

```
;;; REGLA: toma-datos-de-hechos
;;; SI
;;;   la fuente de datos son los hechos y
;;;   hay un sensor asociado con un dispositivo
;;;     activo y
;;;   hay un valor para dicho sensor y
;;;   el sensor todavía no ha registrado ningún
;;;     valor en el ciclo actual
;;; ENTONCES
;;;   indicamos el valor actual de dicho sensor.

(defrule ENTRADA::toma-datos-de-hechos
  (fuente-de-datos hechos)
  (ciclo ?ciclo)
  (sensor (nombre ?id)
          (dispositivo-asociado ?dis))
  (dispositivo (nombre ?dis)
               (estado activo))
  ?h <- (hecho-de-datos ?id ?valor $?resto)
  (not (valor-actual (sensor ?id)
                     (ciclo ?ciclo)))
  =>
  (retract ?h)
  (assert (hecho-de-datos ?id ?resto))
  (assert (valor-actual (sensor ?id)
                        (valor ?valor)
                        (ciclo ?ciclo))))
```

Sistemas de control

```
(defrule ENTRADA::toma-datos-del-sensor
  (fuente-de-datos sensores)
  (ciclo ?ciclo)
  (sensor (nombre ?id) (dispositivo-asociado ?dis))
  (dispositivo (nombre ?dis) (estado activo))
  =>
  (bind ?valor (toma-valor-del-sensor ?id))
  (assert (valor-actual (sensor ?id) (valor ?valor)
                        (ciclo ?ciclo))))

(defrule ENTRADA::toma-datos-del-usuario
  (fuente-de-datos usuario)
  (ciclo ?ciclo)
  (sensor (nombre ?id) (dispositivo-asociado ?dis))
  (dispositivo (nombre ?dis) (estado activo))
  =>
  (printout t "Introduce el valor para el sensor "
            ?id "." crlf)
  (bind ?valor (read))
  (if (not (numberp ?valor)) then (halt))
  (assert (valor-actual (sensor ?id) (valor ?valor)
                        (ciclo ?ciclo))))

(defrule ENTRADA::toma-datos-de-fichero
  (fuente-de-datos fichero)
  ...)
```

Sistemas de control

• Módulo DETECCION

```
(defmodule DETECCION
  (import MAIN deftemplate rangos
                           ciclo)
  (import ENTRADA deftemplate valor-actual)
  (export deftemplate historia-sensor))

(deftemplate DETECCION::estado-sensor
  (slot sensor)
  (slot estado)
  (slot valor)
  (slot ciclo))

(deftemplate DETECCION::historia-sensor
  (slot sensor)
  (slot estado)
  (slot valor)
  (slot ciclo-comienzo)
  (slot ciclo-final))

(deffacts DETECCION::historia-inicial
  (historia-sensor (sensor S1))
  (historia-sensor (sensor S2))
  (historia-sensor (sensor S3))
  (historia-sensor (sensor S4))
  (historia-sensor (sensor S5))
  (historia-sensor (sensor S6)))
```

Sistemas de control

```
;;; REGLA: estado-normal
;;; SI
;;;   el valor registrado por un sensor en el ciclo
;;;     actual está entre los límites normales
;;; ENTONCES
;;;   indicamos que el estado del sensor en el ciclo
;;;     actual es normal.

(defrule DETECCION::estado-normal
  (ciclo ?ciclo)
  (valor-actual (sensor ?id)
                (valor ?valor)
                (ciclo ?ciclo))
  (rangos (nombre ?id)
          (minimo-peligroso ?v1&:(> ?valor ?v1))
          (maximo-peligroso ?v2&:(< ?valor ?v2)))
  =>
  (assert (estado-sensor (sensor ?id)
                         (estado normal)
                         (valor ?valor)
                         (ciclo ?ciclo))))
```

Sistemas de control

```
;;; REGLA: estado-peligroso-bajo
;;; SI
;;;   el valor registrado por un sensor en el ciclo
;;;     actual está entre el mínimo crítico y el
;;;     mínimo peligroso
;;; ENTONCES
;;;   indicamos que el estado del sensor en el ciclo
;;;     actual es peligroso-bajo.

(defrule DETECCION::estado-peligroso-bajo
  (ciclo ?ciclo)
  (valor-actual (sensor ?id)
                (valor ?valor)
                (ciclo ?ciclo))
  (rangos (nombre ?id)
          (minimo-critico ?v1&:(> ?valor ?v1))
          (minimo-peligroso ?v2&:(<= ?valor ?v2)))
  =>
  (assert (estado-sensor (sensor ?id)
                         (estado peligroso-bajo)
                         (valor ?valor)
                         (ciclo ?ciclo))))
```

Sistemas de control

```
;;; REGLA: estado-peligroso-alto
;;; SI
;;;   el valor registrado por un sensor en el ciclo
;;;     actual está entre el máximo crítico y el
;;;     máximo peligroso
;;; ENTONCES
;;;   indicamos que el estado del sensor en el ciclo
;;;     actual es peligroso-alto.

(defrule DETECCION::estado-peligroso-alto
  (ciclo ?ciclo)
  (valor-actual (sensor ?id)
                (valor ?valor)
                (ciclo ?ciclo))
  (rangos (nombre ?id)
          (maximo-peligroso ?v1&:(>= ?valor ?v1))
          (maximo-critico ?v2&:(< ?valor ?v2)))
  =>
  (assert (estado-sensor (sensor ?id)
                         (estado peligroso-alto)
                         (valor ?valor)
                         (ciclo ?ciclo))))
```

Sistemas de control

```
;;; REGLA: estado-critico-bajo
;;; SI
;;;   el valor registrado por un sensor en el ciclo
;;;     actual es menor o igual al mínimo crítico
;;; ENTONCES
;;;   indicamos que el estado del sensor en el ciclo
;;;     actual es critico-bajo.

(defrule DETECCION::estado-critico-bajo
  (ciclo ?ciclo)
  (valor-actual (sensor ?id)
                (valor ?valor)
                (ciclo ?ciclo))
  (rangos (nombre ?id)
          (minimo-critico ?v1&:(<= ?valor ?v1)))
  =>
  (assert (estado-sensor (sensor ?id)
                         (estado critico-bajo)
                         (valor ?valor)
                         (ciclo ?ciclo))))
```

Sistemas de control

```
;;; REGLA: estado-critico-alto
;;; SI
;;;   el valor registrado por un sensor en el ciclo
;;;     actual es mayor o igual al máximo crítico
;;; ENTONCES
;;;   indicamos que el estado del sensor en el ciclo
;;;     actual es critico-alto.

(defrule DETECCION::estado-critico-alto
  (ciclo ?ciclo)
  (valor-actual (sensor ?id)
                (valor ?valor)
                (ciclo ?ciclo))
  (rangos (nombre ?id)
          (maximo-critico ?v1&:(>= ?valor ?v1)))
  =>
  (assert (estado-sensor (sensor ?id)
                         (estado critico-alto)
                         (valor ?valor)
                         (ciclo ?ciclo))))
```

Sistemas de control

```
;;; REGLA: historia-no-cambia
;;; SI
;;;   el estado de un sensor en el ciclo actual no ha
;;;     cambiado con respecto al del ciclo anterior
;;; ENTONCES
;;;   modificamos el historial del sensor para
;;;     indicar que lleva un ciclo más en cierto
;;;     estado.

(defrule DETECCION::historia-no-cambia
  (ciclo ?ciclo)
  ?h <- (historia-sensor
           (sensor ?id)
           (estado ?estado)
           (ciclo-comienzo ?ini)
           (ciclo-final ?fin&=(- ?ciclo 1)))
  (estado-sensor (sensor ?id)
                 (estado ?estado)
                 (valor ?valor)
                 (ciclo ?ciclo))
  =>
  (modify ?h (valor ?valor)
             (ciclo-final ?ciclo)))
```

Sistemas de control

```
;;; REGLA: historia-cambia
;;; SI
;;;   el estado de un sensor en el ciclo actual
;;;     difiere del que tenía en el ciclo anterior
;;; ENTONCES
;;;   modificamos el historial del sensor para
;;;     indicar dicho cambio.

(defrule DETECCION::historia-cambia
  (ciclo ?ciclo)
  ?h <- (historia-sensor
          (sensor ?id)
          (estado ?estado)
          (ciclo-comienzo ?ini)
          (ciclo-final ?fin&=(- ?ciclo 1)))
  (estado-sensor (sensor ?id)
                 (estado ?nuevo&~?estado)
                 (valor ?valor)
                 (ciclo ?ciclo))
  =>
  (modify ?h (estado ?nuevo)
             (valor ?valor)
             (ciclo-comienzo ?ciclo)
             (ciclo-final ?ciclo)))
```

Sistemas de control

```
;;; REGLA: elimina-hechos
;;; SI
;;;   hay hechos sobre los estados de los sensores
;;;     o sus valores actuales que no se
;;;     corresponden con el ciclo actual
;;; ENTONCES
;;;   los eliminamos.

(defrule DETECCION::elimina-hechos
  (ciclo ?ciclo)
  (or ?h <- (estado-sensor (ciclo ~?ciclo))
      ?h <- (valor-actual (ciclo ~?ciclo)))
  =>
  (retract ?h))
```

Sistemas de control

• Módulo AVISOS

```
(defmodule AVISOS
  (import MAIN deftemplate dispositivo
                           sensor
                           duracion
                           ciclo
                           nuevo-ciclo)
  (import DETECCION deftemplate historia-sensor))

;;; REGLA: inserta-nuevo-ciclo
;;; SI
;;;   estamos en cierto ciclo
;;; ENTONCES
;;;   indicamos el nuevo ciclo.

(defrule AVISOS::inserta-nuevo-ciclo
  (ciclo ?ciclo)
  =>
  (assert (nuevo-ciclo (+ 1 ?ciclo))))
```

Sistemas de control

```
;;; REGLA: apaga-en-region-critica
;;; SI
;;;   un sensor se encuentra en estado crítico,
;;;     tanto alto como bajo, y
;;;   el dispositivo asociado está activo
;;; ENTONCES
;;;   desactivamos el dispositivo asociado.

(defrule AVISOS::apaga-en-region-critica
  (ciclo ?ciclo)
  (historia-sensor
          (sensor ?id)
          (estado ?estado&critico-bajo|critico-alto))
  (sensor (nombre ?id)
          (dispositivo-asociado ?dis))
  ?h <- (dispositivo (nombre ?dis)
                     (estado activo))
  =>
  (printout t
    "Ciclo " ?ciclo " - Sensor " ?id " en " ?estado
    "." crlf "  Apagar el dispositivo " ?dis "." crlf)
  (modify ?h (estado inactivo)))
```

Sistemas de control

```
;;; REGLA: apaga-en-region-peligrosa
;;; SI
;;;   un sensor se encuentra en estado peligroso,
;;;     tanto alto como bajo, y
;;;   el tiempo que lleva en dicho estado es
;;;     excesivo (igual al límite) y
;;;   el dispositivo asociado está activo
;;; ENTONCES
;;;   entonces desactivamos el dispositivo asociado.

(defrule AVISOS::apaga-en-region-peligrosa
  (ciclo ?ciclo)
  (historia-sensor
       (sensor ?id)
       (estado ?estado&peligroso-alto|peligroso-bajo)
       (ciclo-comienzo ?ini)
       (ciclo-final ?fin))
  (duracion (sensor ?id)
            (tiempo ?tiempo&=(+ (- ?fin ?ini) 1)))
  (sensor (nombre ?id)
          (dispositivo-asociado ?dis))
  ?h <- (dispositivo (nombre ?dis)
                     (estado activo))
  =>
  (printout t
    "Ciclo " ?ciclo " - Sensor " ?id " en " ?estado
    " durante " (+ (- ?fin ?ini) 1) " ciclos." crlf
    "  Apagar el dispositivo " ?dis "." crlf)
  (modify ?h (estado inactivo)))
```

Sistemas de control

```
;;; REGLA: sensor-en-region-peligrosa
;;; SI
;;;   un sensor se encuentra en estado peligroso,
;;;     tanto alto como bajo, y
;;;   el tiempo que lleva en dicho estado no es
;;;     excesivo (menor que el límite) y
;;;   el dispositivo asociado está activo
;;; ENTONCES
;;;   avisamos al usuario de la situación.

(defrule AVISOS::sensor-en-region-peligrosa
  (ciclo ?ciclo)
  (historia-sensor
       (sensor ?id)
       (estado ?estado&peligroso-alto|peligroso-bajo)
       (ciclo-comienzo ?ini)
       (ciclo-final ?fin))
  (duracion (sensor ?id)
            (tiempo ?tiempo
                  &:(< (+ (- ?fin ?ini) 1) ?tiempo)))
  (sensor (nombre ?id)
          (dispositivo-asociado ?dis))
  (dispositivo (nombre ?dis)
               (estado activo))
  =>
  (printout t
    "Ciclo " ?ciclo " - Sensor " ?id " en " ?estado
    " durante " (+ (- ?fin ?ini) 1) " ciclos." crlf))
```