Hechos y reglas

José A. Alonso y Francisco J. Martín

Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Entrando y saliendo de CLIPS

- Sesión

```
merlin:~> clips

==============================
Clips para IA2  (Version 6.04)
==============================

CLIPS> (+ 2 3)
5
CLIPS> (* 2 (+ 3 4 5))
24
CLIPS> (exit)
>
```

- Comentarios
  - Versión de CLIPS
  - Semejanza con LISP

Introducción de hechos

- Sesión

```
CLIPS> (assert (rojo))
<Fact-0>
CLIPS> (assert (negro))
<Fact-1>
CLIPS>
```

- Comentarios
  - `(assert <hecho>)`
  - Identificador de un hecho
  - Indice de un hecho

Lista de hechos

- Sesión

```
CLIPS> (facts)
f-0     (rojo)
f-1     (negro)
For a total of 2 facts.
CLIPS>
```

- Comentarios
  - `(facts)`
  - Lista de hechos

Eliminación de un hecho

- Sesión

```
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
For a total of 1 fact.
CLIPS> (assert (rojo))
<Fact-1>
CLIPS> (facts)
f-0     (initial-fact)
f-1     (rojo)
For a total of 2 facts.
CLIPS>
```

- Comentarios
  - `(reset)`
  - `(initial-fact)`

Duplicación de hechos

- Sesión

```
CLIPS> (assert (rojo))
FALSE
CLIPS> (facts)
f-0     (initial-fact)
f-1     (rojo)
For a total of 2 facts.
CLIPS>
```

- Comentarios
  - **Imposibilidad de duplicación**
  - `FALSE`

Introducción de varios hechos

- Sesión

```
CLIPS> (assert (blanco) (azul))
<Fact-3>
CLIPS> (facts)
f-0     (initial-fact)
f-1     (rojo)
f-2     (blanco)
f-3     (azul)
For a total of 4 facts.
CLIPS>
```

- Comentario
  - (assert <hecho>+)

Retorno al estado inicial

- Sesión

```
CLIPS> (facts)
f-0     (initial-fact)
f-1     (rojo)
f-2     (blanco)
f-3     (azul)
For a total of 4 facts.
CLIPS> (clear)
CLIPS> (facts)
CLIPS>
```

- Comentarios
  - `(clear)`
  - Diferencias entre `(reset)` y `(clear)`

Lista de algunos hechos

- Sesión

```
CLIPS> (clear)
CLIPS> (assert (a) (b) (c) (d))
<Fact-3>
CLIPS> (facts)
f-0     (a)
f-1     (b)
f-2     (c)
f-3     (d)
For a total of 4 facts.
CLIPS> (facts 2)
f-2     (c)
f-3     (d)
For a total of 2 facts.
CLIPS> (facts 1 2)
f-1     (b)
f-2     (c)
For a total of 2 facts.
CLIPS> (facts 0 2 1)
f-0     (a)
For a total of 1 fact.
CLIPS>
```

- Comentario
  - (facts [<inicial> [<final> [<máximo>]]])

Hechos con múltiples campos

- Sesión

```
CLIPS> (clear)
CLIPS> (assert (blanca))
<Fact-0>
CLIPS> (assert (bandera blanca))
<Fact-1>
CLIPS> (assert (bandera verde))
<Fact-2>
CLIPS> (assert (bandera blanca verde))
<Fact-3>
CLIPS> (facts)
f-0     (blanca)
f-1     (bandera blanca)
f-2     (bandera verde)
f-3     (bandera blanca verde)
For a total of 4 facts.
CLIPS>
```

- Comentarios
  - `hecho ::= (<simbolo> <campo>*)`
  - **Nombre de la relación**
  - **Estilo de programación**

Tipos de campos: Enumeración

- Tipos de campos como en LISP
  - Enteros.
  - Números en punto flotante.
  - Símbolos.
  - Cadenas.
- Tipos de campos propios
  - Direcciones de hechos.
  - Direcciones externas.
  - Instancias de nombres.
  - Instancias de direcciones.
- Comentarios
  - Determinación del tipo de campo por su valor
  - Caracteres prohibidos en los símbolos:

    ```
    < | & ( ) ; " ~ ,
    ```

  - Caracteres prohibidos al comienzo:

    ```
    $ ? + -
    ```

Tipos de campos: Ejemplo

- Sesión

```
CLIPS> (clear)
CLIPS> (assert (numero 1))
<Fact-0>
CLIPS> (assert (x 1.5))
<Fact-1>
CLIPS> (assert (y -1))
<Fact-2>
CLIPS> (assert (Numero 1))
<Fact-3>
CLIPS> (assert (distancia 3.5e5))
<Fact-4>
CLIPS> (assert (coordenadas 1 2 3))
<Fact-5>
CLIPS> (assert (coordenadas 1 3 2))
<Fact-6>
CLIPS> (facts)
f-0     (numero 1)
f-1     (x 1.5)
f-2     (y -1)
f-3     (Numero 1)
f-4     (distancia 350000.0)
f-5     (coordenadas 1 2 3)
f-6     (coordenadas 1 3 2)
For a total of 7 facts.
CLIPS> (assert (3 cosa))
[PRNTUTIL2] Syntax Error:  Check appropriate syntax
for first field of a RHS pattern.
CLIPS>
```

Tipos de campos: Ejemplo

- Comentarios
  - Diferenciación entre mayúsculas y minúsculas
  - Expansión de flotante
  - Importancia del orden
  - Error sintáctico

Tipos de campos: Cadenas

- Sesión

```
CLIPS> (clear)
CLIPS> (assert (La   pelicula   "rojo"
es
muy     buena))
<Fact-0>
CLIPS> (facts)
f-0     (La pelicula "rojo" es muy buena)
For a total of 1 fact.
CLIPS> (assert (La pelicula "ro      jo
" es muy buena))
<Fact-1>
CLIPS> (facts)
f-0     (La pelicula "rojo" es muy buena)
f-1     (La pelicula "ro      jo
" es muy buena)
For a total of 2 facts.
```

- Comentario
  - Delimitadores en símbolos y cadenas

Norma de estilo

- Estilo incorrecto

```
CLIPS> (clear)
CLIPS> (assert (Rojo) (Blanco) (Azul))
<Fact-2>
CLIPS> (facts)
f-0     (Rojo)
f-1     (Blanco)
f-2     (Azul)
For a total of 3 facts.
CLIPS>
```

- Estilo correcto

```
CLIPS> (clear)
CLIPS> (assert (pelicula Rojo))
<Fact-0>
CLIPS> (assert (pelicula Blanco))
<Fact-1>
CLIPS> (assert (pelicula Azul))
<Fact-2>
CLIPS> (facts)
f-0     (pelicula Rojo)
f-1     (pelicula Blanco)
f-2     (pelicula Azul)
For a total of 3 facts.
CLIPS>
```

Eliminación de algunos hechos

- Sesión

```
CLIPS> (facts)
f-0     (pelicula Rojo)
f-1     (pelicula Blanco)
f-2     (pelicula Azul)
For a total of 3 facts.
CLIPS> (retract 1)
CLIPS> (facts)
f-0     (pelicula Rojo)
f-2     (pelicula Azul)
For a total of 2 facts.
CLIPS> (retract 0 2)
CLIPS> (facts)
CLIPS> (assert (pelicula Goldeneye))
<Fact-3>
CLIPS> (facts)
f-3     (pelicula Goldeneye)
For a total of 1 fact.
CLIPS> (assert (pelicula Highlander))
<Fact-4>
CLIPS> (facts)
f-3     (pelicula Goldeneye)
f-4     (pelicula Highlander)
For a total of 2 facts.
CLIPS> (retract *)
CLIPS> (facts)
CLIPS> (retract 2)
[PRNTUTIL1] Unable to find fact f-2.
CLIPS>
```

Eliminación de algunos hechos

- Comentarios
  - `(retract <indice-de-hecho>+ | *)`
  - **Numeración después de eliminar**
  - **Eliminación de hecho inexistente**

Vigilancia de hechos

- Sesión

```
CLIPS> (clear)
CLIPS> (watch facts)
CLIPS> (assert (pelicula Rojo))
==> f-0     (pelicula Rojo)
<Fact-0>
CLIPS> (reset)
<== f-0     (pelicula Rojo)
==> f-0     (initial-fact)
CLIPS> (assert (pelicula Blanco))
==> f-1     (pelicula Blanco)
<Fact-1>
CLIPS> (retract 1)
<== f-1     (pelicula Blanco)
CLIPS> (facts)
f-0     (initial-fact)
For a total of 1 fact.
CLIPS>
```

- Comentarios
  - `(watch facts)`
  - `<==` y `==>`

Ayudas

- **Sesión**

```
CLIPS> (help)
Loading help file entries from clips.hlp.
Please wait...


HELP_USAGE                    FUNCTION_SUMMARY
RELEASE_NOTES                 COMMAND_SUMMARY
CONSTRUCT_SUMMARY             INTEGRATED_EDITOR

MAIN Topic? com

COMMAND_SUMMARY

This section gives a general overview of the
available CLIPS commands.

Subtopics:
ENVIRONMENT_COMMANDS          DEFGLOBAL_COMMANDS
DEBUGGING_COMMANDS            DEFFUNCTION_COMMANDS
DEFTEMPLATE_COMMANDS          GENERIC_FUNCTION_COMMANDS
FACT_COMMANDS                 COOL_COMMANDS
DEFFACTS_COMMANDS             DEFMODULE_COMMANDS
DEFRULE_COMMANDS              MEMORY_COMMANDS
AGENDA_COMMANDS               TEXT_PROCESSING_COMMANDS

COMMAND_SUMMARY Topic? fac
```

Ayudas

```
COMMAND_SUMMARY
  FACT_COMMANDS

The following commands display information
about facts.

FACTS: Display the facts in the fact-list.
(facts [<module-name>]
       [<start-integer-expression>
       [<end-integer-expression>
       [<max-integer-expression>]]])

....

COMMAND_SUMMARY Topic?
MAIN Topic?
CLIPS>
```

Definición de reglas

- Sesión

```
;;; REGLA: rojo
;;; SI
;;;   la pelicula es Rojo
;;; ENTONCES
;;;   el director es Kieslowski.
CLIPS> (defrule rojo "Ejemplo 1"          ; Cabecera
          (pelicula Rojo)                 ; Patron
          =>                              ; ENTONCES
          (assert (director Kieslowski))); Accion
CLIPS>
```

- Comentarios
  - Comentarios en CLIPS
  - Documentación de reglas
  - Sintaxis de las reglas

```
(defrule <nombre> [<comentario>]
   <patron>*
   =>
   <accion>*)
```

  - Significado de una regla

Lectura de reglas

- Sesión

```
CLIPS> (rules)
rojo
For a total of 1 defrule.
CLIPS> (ppdefrule rojo)
(defrule MAIN::rojo "Ejemplo 1"
   (pelicula Rojo)
   =>
   (assert (director Kieslowski)))
CLIPS>
```

- Comentarios
  - `(rules)` y `(ppdefrule <nombre>)`

Eliminación de reglas

- Sesión

```
CLIPS> (rules)
rojo
CLIPS> (undefrule rojo)
CLIPS> (rules)
CLIPS>
```

- Comentario
  - `(undefrule)`

Agenda

- Sesión

```
CLIPS> (assert (pelicula Rojo))
<Fact-0>
CLIPS> (defrule rojo
          (pelicula Rojo)
          =>
          (assert (director Kieslowski)))
CLIPS> (agenda)
0      rojo: f-0
For a total of 1 activation.
CLIPS>
```

- Comentarios
  - `(agenda)`
  - **Equiparación de condiciones**
  - **Activación de reglas**
  - **Agenda**
  - **Prioridad de una regla de la agenda**

Ejecución

- Sesión

```
CLIPS> (agenda)
0      rojo: f-0
For a total of 1 activation.
CLIPS> (facts)
f-0     (pelicula Rojo)
For a total of 1 fact.
CLIPS> (run)
CLIPS> (agenda)
CLIPS> (facts)
f-0     (pelicula Rojo)
f-1     (director Kieslowski)
For a total of 2 facts.
CLIPS>
```

- Comentarios
  - `(run)`
  - **Refractación**

Vigilancia de activaciones

- Sesión

```
CLIPS> (defrule rojo "La regla del rojo"
          (pelicula Rojo)
          =>
          (printout t "Usada rojo" crlf))
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (assert (pelicula Rojo))
==> f-0     (pelicula Rojo)
==> Activation 0      rojo: f-0
<Fact-0>
CLIPS> (assert (pelicula Rojo))
FALSE
CLIPS> (agenda)
0      rojo: f-0
For a total of 1 activation.
CLIPS> (run)
Usada rojo
CLIPS> (agenda)
CLIPS> (facts)
f-0     (pelicula Rojo)
For a total of 1 fact.
CLIPS>
```

- Sesión
  - (watch activations)
  - (printout t <expresion>* crlf)

Cargando reglas de ficheros

- Sesión

```
CLIPS> (load "regla.clp")
Defining defrule: rojo +j
TRUE
CLIPS> (rules)
rojo
For a total of 1 defrule.
CLIPS> (ppdefrule rojo)
(defrule MAIN::rojo "Ejemplo 1"
   (pelicula Rojo)
   =>
   (assert (director Kieslowski)))
CLIPS>
```

- Comentarios
  - **Extensión** `.clp`
  - `(load <fichero>)`

Base con varias reglas

- Sesión

```
CLIPS> (defrule luz-roja
          (luz roja)
          =>
          (printout t "No pases" crlf))
CLIPS> (defrule luz-verde
          (luz verde)
          =>
          (printout t "Puedes pasar" crlf))
CLIPS> (rules)
luz-roja
luz-verde
For a total of 2 defrules.
CLIPS> (agenda)
CLIPS> (assert (luz roja))
<Fact-0>
CLIPS> (agenda)
0      luz-roja: f-0
For a total of 1 activation.
CLIPS> (run)
No pases
CLIPS> (agenda)
CLIPS> (assert (luz verde))
<Fact-1>
CLIPS> (facts)
f-0     (luz roja)
f-1     (luz verde)
For a total of 2 facts.
CLIPS> (run)
Puedes pasar
```

Reglas con varias condiciones

- Sesión

```
CLIPS> (clear)
CLIPS> (defrule estar-parado
          (estado parado)
          (luz rojo)
          =>
          (printout t "Estamos parados" crlf))
CLIPS> (defrule estar-andando
          (estado andando)
          (sennal andar)
          =>
          (printout t "Estamos andando" crlf))
CLIPS> (rules)
estar-parado
estar-andando
For a total of 2 defrules.
CLIPS> (assert (estado andando))
<Fact-0>
CLIPS> (agenda)
CLIPS> (assert (luz verde))
<Fact-1>
CLIPS> (agenda)
CLIPS> (assert (sennal andar))
<Fact-2>
CLIPS> (agenda)
0      estar-andando: f-0,f-2
For a total of 1 activation.
CLIPS> (run)
Estamos andando
CLIPS>
```

Estrategia de resolución de conflictos

- Resolución de conflictos
- Estrategias de resolución de conflictos en CLIPS
  - Profundidad. (Por defecto)
  - Anchura.
  - Aleatoria.
  - Complejidad.
  - Simplicidad.
  - LEX.
  - MEA.

Hechos iniciales

- Sesión

```
CLIPS> (clear)
CLIPS> (deffacts andar "Algunos hechos sobre andar"
          (estado andando)
          (sennal andar))
CLIPS> (facts)
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (estado andando)
f-2     (sennal andar)
For a total of 3 facts.
CLIPS> (clear)
CLIPS> (facts)
CLIPS>
```

- Comentario
  - `(deffacts <nombre> [comentario] <hecho>+)`
  - Generalización de `(reset)`
  - Diferencia de `(reset)` y `(clear)`

Reset y clear (I)

- Sesión

```
CLIPS> (clear)
CLIPS> (defrule luz-roja
         (luz roja)
         =>
         (printout t "Estamos parados" crlf))
CLIPS> (assert (animal pato))
<Fact-0>
CLIPS> (defrule pato
         (animal pato)
         =>
         (printout t "Hay un pato" crlf))
CLIPS> (deffacts andar "algunos hechos sobre andar"
         (luz roja)
         (sennal andar))
```

Reset y clear (II)

- Sesión

```
CLIPS> (facts)
f-0     (animal pato)
For a total of 1 fact.
CLIPS> (rules)
luz-roja
pato
For a total of 2 defrules.
CLIPS> (agenda)
0      pato: f-0
For a total of 1 activation.
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (luz roja)
f-2     (sennal andar))
For a total of 2 facts.
CLIPS> (agenda)
0      luz-roja: f-1
For a total of 1 activation.
CLIPS> (clear)
CLIPS> (facts)
CLIPS> (rules)
CLIPS> (agenda)
CLIPS>
```

Eliminación de hechos definidos

- Sesión

```
CLIPS> (deffacts andar "algunos hechos sobre andar"
         (luz verde)
         (sennal andar))
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (luz verde)
f-2     (sennal andar)
For a total of 3 facts.
CLIPS> (undeffacts andar)
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
For a total of 1 fact.
CLIPS>
```

- Comentario
  - (undeffacts <nombre>+)

Desvío de la salida a un fichero

- `(dribble-on <nombre-del-fichero>)`
- `(dribble-off)`

Ejecución parcial (I)

- Sesión

```
CLIPS> (clear)
CLIPS> (defrule luz-roja
         (luz roja)
         =>
         (printout t "Estamos parados" crlf))
CLIPS> (defrule luz-verde
         (luz verde)
         =>
         (printout t "Podemos andar" crlf))
CLIPS> (defrule andando
         (estado andando)
         =>
         (printout t "Estamos andando" crlf))
CLIPS> (deffacts inicio
         (luz roja)
         (luz verde)
         (estado andando))
```

Ejecución parcial (II)

- Sesión

```
CLIPS> (facts)
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (luz roja)
f-2     (luz verde)
f-3     (estado andando)
For a total of 4 facts.
CLIPS> (agenda)
0      andando: f-3
0      luz-verde: f-2
0      luz-roja: f-1
For a total of 3 activations.
CLIPS> (run 2)
Estamos andando
Podemos andar
CLIPS> (agenda)
0      luz-roja: f-1
For a total of 1 activation.
CLIPS>
```

- Comentario
  - `(run <numero>)`

Equiparaciones (I)

- Sesión

```
CLIPS> (defrule tomar-vacaciones
         (trabajo hecho)
         (dinero suficiente)
         (reservas hechas)
         =>
         (printout t "Me voy a Australia" crlf))
CLIPS> (assert (trabajo hecho))
<Fact-0>
CLIPS> (matches tomar-vacaciones)
Matches for Pattern 1
f-0
Matches for Pattern 2
None
Matches for Pattern 3
None
Partial matches for CEs 1 - 2
None
Partial matches for CEs 1 - 3
None
Activations
None
```

Equiparaciones (II)

- Sesión

```
CLIPS> (assert (dinero suficiente))
<Fact-1>
CLIPS> (assert (reservas hechas))
<Fact-2>
CLIPS> (matches tomar-vacaciones)
Matches for Pattern 1
f-0
Matches for Pattern 2
f-1
Matches for Pattern 3
f-2
Partial matches for CEs 1 - 2
f-0,f-1
Partial matches for CEs 1 - 3
f-0,f-1,f-2
Activations
f-0,f-1,f-2
CLIPS>
```

- Comentario
  - (matches <nombre-de-regla>)