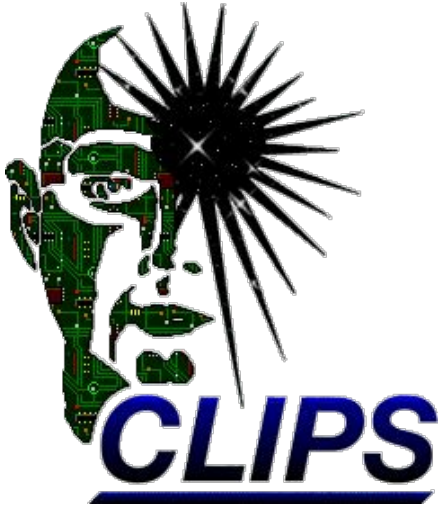
Inteligencia Artificial: CLIPS



José Marcos Moreno-Vega (jmmoreno@ull.es)
Israel López-Plata (ilopezpl@ull.es)
Christopher Expósito-Izquierdo (cexposit@ull.es)

Memoria de Trabajo

- La Memoria de Trabajo almacena el conocimiento factual del Sistema Experto
- Las reglas del sistema utilizan este conocimiento para determinar el comportamiento del sistema. También se puede mostrar al usuario
- La Memoria de Trabajo consta de:
  - **Lista de hechos:** Un hecho es la unidad básica de conocimiento que define el estado del sistema. Es la unidad de datos fundamental utilizada en las reglas
  - **Lista de instancias:** Una instancia es un caso particular de un objeto. Un objeto se utiliza para almacenar información de forma estructurada, y se compone por una serie de campos con un valor asociado

Hecho: luz_encendida

Instancia: habitación150
- num: 150
- piso: 1º
- baños: 1
- superficie: 50

MT. Constructor

- Estructura que permite definir:
  - Funciones
  - Reglas
  - Hechos
  - Clases
  - etc.

- No devuelven ningún valor

- Los elementos que añaden se incluyen en la Base de Conocimientos

MT. Tipos de datos

- Los campos que se encuentran en la Base de Conocimiento, así como los que componen los hechos, se pueden definir con diferentes tipos de datos según su utilización. En CLIPS se dispone de los siguientes tipos:

- **Símbolos**
  - SYMBOL: Secuencia de caracteres imprimibles. No permite espacios ni caracteres raros.
  - STRING: Conjunto de caracteres imprimibles encerrados entre comillas dobles (“).

- **Números**
  - INTEGER: Números enteros.
  - FLOAT: Números en punto flotante (permite decimales).

- **Direcciones de la MT**
  - EXTERNAL-ADDRESS: Dirección de una estructura de datos externa devuelta por alguna función escrita en otro lenguaje y que ha sido integrada en CLIPS
  - FACT-ADDRESS: Dirección de los hechos en la MT
  - INSTANCE-ADDRESS: Dirección de una instancia de un objeto en la MT

MT. Hecho

- Representa una lista de valores atómicos de la Memoria de Trabajo.

- Un hecho puede ser añadido (asertado), modificado o eliminado de la Memoria de Trabajo tanto por el usuario como por el propio programa CLIPS, ya sea a través de **reglas** o **constructores**.

- Se distinguen 2 tipos de hechos según como se pueden referenciar sus valores atómicos:
  - **Hechos ordenados:** Sus valores se indexan por la posición que ocupan.
  - **Hechos no ordenados:** A cada uno de sus valores se le asigna un nombre con el que puede ser referenciado.

- A un hecho se puede acceder mediante un índice o a través de una dirección.

MT. Hechos ordenados

- Conjunto de hechos cuyos valores se indexan por la posición que ocupan.
- Para que sus valores puedan ser utilizados en una regla, se debe conocer la posición en la que se encuentran.
- No es necesario declararlos.
- Los campos de un hecho ordenado pueden ser de cualquier tipo, excepto el identificador, que debe ser un SYMBOL.
- **Ejemplo:**

componentes_ordenador teclado raton torre pantalla

identificador *valores*

MT. Hechos no ordenados

- Conjunto de hechos cuyos valores se indexan por nombre.
- Para acceder a sus valores en una regla, se indexan por nombre.
- Se declaran con el constructor ***deftemplate***, donde se define su nombre así como los diferentes slots.
- Un slot es un conjunto de nombre-valor que conforma el hecho no ordenado.
- **Ejemplo:**

Huesped
- Nombre
- Sexo
- Fuma?
- Alojado

```
(deftemplate huesped
   (slot nombre)
   (slot sexo)
   (slot fuma?)
   (slot alojado))
```

Habitación
- Número
- Capacidad
- Sexos
- Fuman?
- Plazas-libres
- Ocupantes

```
(deftemplate habitación
   (slot número)
   (slot capacidad)
   (slot sexos)
   (slot fuman?)
   (slot plazas-libres)
   (multislot
ocupantes))
```

MT. Hechos no ordenados

- Existen 2 tipos de slots.
- Los slots representan un valor simple. Esto significa que si se asigna un valor al slot el antiguo se reemplaza.
- Los multislot representan un campo que contiene una **lista de valores.** A un multislot se le puede añadir y eliminar elementos, sin sustituir los existentes.
- A un slot se le puede definir un conjunto de atributos, que pueden ser:
  - Tipo de datos
  - Valor por defecto
  - Valores permitidos
  - etc.

MT. Atributos. Tipo

```
type <especificación_tipo>
```

- Define el tipo de datos que posee un slot. Los tipos permitidos son los siguientes:
  - SYMBOL
  - STRING
  - LEXEME (SYMBOL+STRING)
  - INTEGER
  - FLOAT
  - NUMBER (INTEGER+FLOAT)
  - ?VARIABLE. El slot puede tomar cualquier tipo de datos. Tipo por defecto.

- **Ejemplo:** (slot edad (type INTEGER)))

MT. Atributos. Valores permitidos

```
allowed-<tipo_permitido>  <lista_valores>
```

- Indica los valores que se permiten para un tipo específico. Los tipos son los siguientes:
  - allowed-symbols
  - allowed-strings
  - allowed-lexemes
  - allowed-integers
  - allowed-floats
  - allowed-numbers
  - allowed-values. El slot puede tomar cualquier tipo de datos.

- Si en lugar de una lista de valores se quiere indicar que todos los elementos están permitidos, se utiliza ? VARIABLE. Es el valor por defecto.

- **Ejemplo:** (slot sexo (type SYMBOL)(allowed-symbols hombre mujer)))

MT. Atributos. Rango de valores permitidos

```
range <limite_inferior> <limite_superior>
```

- Permite definir un rango de valores permitidos en los slots de tipo numéricos (INTEGER, FLOAT o NUMBER).
- Los límites se definen como valores numéricos.
- Se puede indicar un límite como ∞ utilizando el valor ?VARIABLE (valor por defecto).
- **Ejemplo:** (range ?VARIABLE 3) = [-∞, 3]
  (range 14 ?VARIABLE) = [14, ∞]
  (range ?VARIABLE ?VARIABLE) = [-∞, ∞]

MT. Atributos. Cardinalidad

```
cardinality <limite_inferior> <limite_superior>
```

- Permite definir el número mínimo y máximo que un **multislot** puede contener.
- Los límites se definen como valores numéricos.
- Se puede indicar un límite como ∞ utilizando el valor ?VARIABLE (valor por defecto).
- **Ejemplo:** (multislot jugadores (type STRING) (cardinality 6 6))
  (multislot alternativos (type STRING) (cardinality 0 2)))

MT. Atributos. Valor por defecto

```
default <valor_defecto>
```

- Permite definir el valor por defecto de un slot.

- Este valor por defecto puede ser.
  - **Valor simple.** Para los slots simples. *(slot a (default 3))*
  - **Lista de valores.** Para los multislot. *(multislot c (default a b c))*
  - **Función.** El valor por defecto será el retornado por la función. *(slot b (default (+ 3 4)))*
  - **?NONE.** Indica que el slot debe tener siempre un valor. Cuando se crea el hecho se obliga a poner un valor al slot. *(slot DNI (default ?NONE))*
  - **?DERIVE.** Establece un valor por defecto predeterminado, el cual debe satisfacer el resto de atributos del slot. Es el valor por defecto.

MT. Atributos. Valor por defecto. ? DERIVE

- Los valores por defecto del tipo ?DERIVE se obtienen dependiendo del tipo del slot.
  - **SYMBOL:** nil
  - **STRING:** “”
  - **LEXEME:** nil
  - **INTEGER:** 0
  - **FLOAT:** 0.0
  - **NUMBER:** 0
  - **multislot:** Lista vacía ()

- **Ejemplos:** (slot a): nil
  (slot b (type INTEGER)): 0
  (slot c (allowed-values rojo verde azul)): rojo
  (multislot d): ()
  (multislot e (cardinality 2 2) (type FLOAT) (range 3.5 10.0))): (3.5 3.5)

MT. Hechos iniciales

deffacts <nombre> [<comentarios>] <slots>

- La sentencia deffacts permite definir un conjunto de hechos como iniciales.
- Los hechos iniciales se añaden a la Memoria de Trabajo con la sentencia **reset**.
- El comando reset elimina todos los hechos que hubiera en la lista de hechos actual, y a continuación añade los hechos correspondientes a sentencias deffacts.
- **Ejemplos:**

```
(deffacts arranque
   (frigorífico interruptor encendido)
   (frigorífico puerta abierta)
   (frigorífico temperatura (get-
temp)))
```

```
(deffacts estudiante "Todos los estudiantes iniciales"
   (estudiante (nombre Juan)(sexo varón)(fuma? no)(alojado no))
   (estudiante (nombre Pepe)(sexo varón)(fuma? si)(alojado no))
   (estudiante (nombre Luisa)(sexo mujer)(fuma? no)(alojado
no))
   (estudiante (nombre Pedro)(sexo varón)(fuma? no)(alojado
no)))
```