



Grado en Ingeniería Informática, Universidad de Córdoba

Sistemas Inteligentes

CLIPS

Tema 1: Introducción

Aurora Esteban Toscano
aestebant@uco.es

07 de marzo de 2022



Objetivos

- Introducir los Sistemas Expertos.
- Instalar CLIPS.
- Familiarizarse con el entorno de CLIPS.



Los **Sistemas basados en reglas** son mecanismos para representar el conocimiento con reglas de tipo si-entonces o if-then.

- También pueden conocerse como Sistemas de Producción.

Por ejemplo

SI es de noche, ENTONCES enciende la luz.

Una aplicación fundamental son los **Sistemas Expertos**: sistemas que buscan emular la toma de decisiones de un experto humano para resolver problemas complejos → una de las formas más populares de *Inteligencia Artificial*.



Sistemas Expertos (basados en reglas) I

Los Sistemas Expertos son capaces de llegar a un diagnóstico aplicando reglas de forma encadenada sobre un conjunto de datos por medio de los siguientes componentes:

- **Base de conocimientos:** almacena las reglas previamente definidas → conocimiento estático.
- **Base de afirmaciones:** almacena los hechos sobre los que trabajan las reglas → conocimiento dinámico.
- **Motor de inferencia:** aplica las reglas a los hechos para deducir nuevos hechos.
- **Interfaz de usuario:** permite introducir la información de partida, seguir el proceso de razonamiento o inferencia y obtener el diagnóstico final.



Por ejemplo

Base de conocimientos:

- Todos los hombres son animales.
- Todos los animales respiran.

Base de afirmaciones:

- Juan es un hombre.

Inferencia:

- ① Juan es un animal.
- ② Juan respira.



Dos partes fundamentales

Antecedente \implies Consecuente

- Antecedente: cero o más *cláusulas* que deben cumplirse para que la regla pueda ejecutarse (dispararse).
- Consecuente: cero o más *acciones* que se llevarán a cabo si la regla se dispara.
 - Entre esas acciones puede estar crear (afirmar) más hechos, eliminar hechos obsoletos, llegar a conclusiones finales...

Por ejemplo

SI voy a clase Y atiando Y hago los ejercicios \implies apruebo la asignatura



El motor de inferencia determina cómo se pasa de los hechos al conocimiento.

- Las reglas se ejecutan hacia adelante: si se satisface el antecedente se efectúan las acciones del consecuente.
- Un grupo de reglas que contienen un problema y una solución se llama cadena:
 - Encadenamiento hacia delante o basado en datos: va desde los hechos iniciales, pasando por hechos inferidos y acabando en una conclusión.
 - **Nos centraremos en este.**
 - Encadenamiento hacia atrás o basado en objetivos: parte de una hipótesis objetivo que deberá probarse mediante hipótesis intermedias que deberán estar sostenida por hechos.



El **Control de razonamiento** es un mecanismo del motor de inferencia que se encarga de seleccionar *qué regla disparar si hay varias disponibles*.

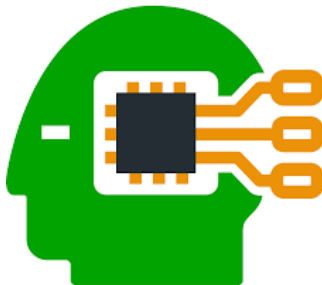
Métodos de resolución de conflictos:

- 1 Ordenación de las reglas.
- 2 Ordenar las cláusulas dentro de cada regla.
- 3 Añadir nuevas cláusulas relacionadas con las inferencias.
- 4 Control mediante agenda.
- 5 Agendas con patrocinadores.
- 6 Conjuntos de reglas.
- 7 Modelos de reglas y metarreglas.
- 8 Mecanismos basados en la sensibilidad y estabilidad del sistema.



CLIPS (*C Language Integrated Production System*) es una herramienta para el desarrollo de Sistemas Expertos.


- Originada en 1986, *NASA Software Technology Branch*.
- En la actualidad va por la versión 6.40, de abril de 2021.
- Proyecto: <https://sourceforge.net/projects/clipsrules/>






Desde la página del proyecto: Files > CLIPS > 6.40


Home / Browse / Development / Interpreters / CLIPS Rule Based Programming Language / Files




CLIPS Rule Based Programming Language Files

Expert System Tool
Brought to you by: [garyriley](#)

Summary	Files	Reviews	Support	Wiki	Tickets ▼	News	Discussion	Donate 	Code
---------	--------------	---------	---------	------	-----------	------	------------	--	------

**Download Latest Version**
clips_ios_640.dmg (3.0 MB)

Get Updates



[Home](#) / [CLIPS](#) / 6.40

Name ▾	Modified ▾	Size ▾	Downloads / Week ▾
--------	------------	--------	--------------------

↶ [Parent folder](#)

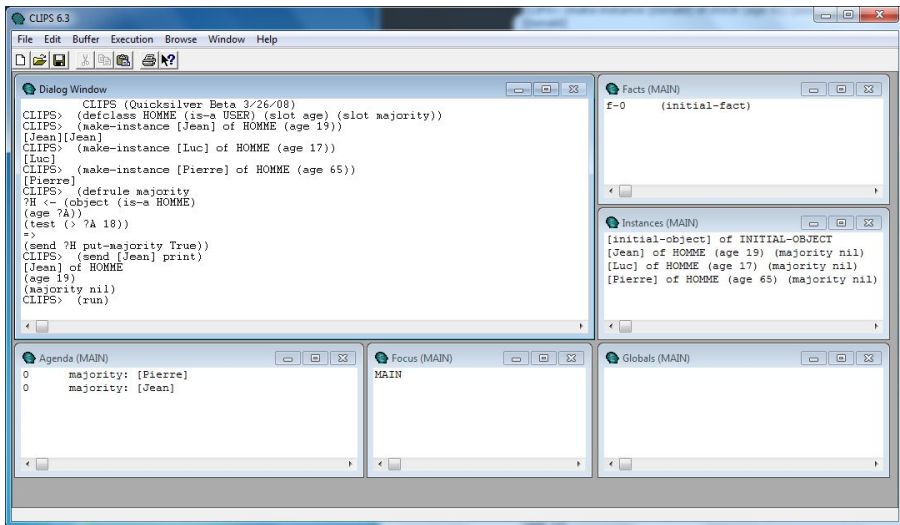


Según tu sistema operativo:

- Windows: clips_windows_64_bit_installer_640.msi y seguir el asistente.
- MacOS: clips_macos_executable_640.dmg y seguir el asistente.
- Linux: clips_jni_640.zip (requiere Java)
 - 1 Compilar la librería en CLIPSJNI/library-src: `make -f makefile.lnx <dist>`
 - Posibles: ubuntu, fedora, debian, mint, centos
 - 2 Copiar libCLIPSJNI.so a la raíz de la carpeta de instalación
 - 3 Ejecutar `java -Djava.library.path=. -jar CLIPSIDE.jar`
- Sistema UCO Thinstation: ya está instalado.



Interfaz de CLIPS





- Memoria de Trabajo (facts): memoria global que contiene los hechos (fact-list) que representan el conocimiento que el sistema ha adquirido del problema particular que intenta resolver.
- Base de reglas (knowledge base): contiene las reglas que representan el conocimiento general de resolución de problemas.
- Intérprete (inference engine): controla la ejecución global de las reglas.
 - CLIPS es un programa dirigido por los hechos → encadenamiento hacia delante.



- CLIPS Reference Manual
 - Volumen I. The Basic Programming Guide.
 - Volumen II. The Advanced Programming Guide.
 - Volumen III. The Interfaces Guide.
- CLIPS User's Guide.

Descarga (comprobar que es la versión 6.40)

https://sourceforge.net/projects/clipsrules/files/CLIPS/6.40/clips_examples_640.zip/download

- Ayuda de CLIPS: Menú Help > CLIPS HELP
 - Contenido: sumario de comandos, constructores, funciones.
 - Índice: para buscar por palabra.



Abrir el fichero hombre.clp (descarga de Moodle).

- Base de conocimiento: 2 reglas: r-hombre-animal y r-animal-respira
- Base de afirmaciones: Juan
- Fíjate en el uso de variables.
- Carga las reglas y afirma el hecho.
- Fíjate en la ventana de hechos y/o ejecuta (facts)
- Fíjate en la ventana de agenda y/o ejecuta (agenda)
- Ejecuta (run) para disparar el motor de inferencia.



- **Tipos de datos:** representan información.
- **Funciones:** manipular los datos.
- **Constructores:** añadir conocimiento a la Base de Conocimiento.
- **Comentarios:** añadir interpretabilidad a la información representada.



Tipos de datos primitivos:

- INTEGER: número entero. Por ejemplo: 23125, +89, -71
- FLOAT: número real. Por ejemplo: 46.5, -3.14, 45e5, +1e10
- SYMBOL: símbolo. Por ejemplo: Hola, hola, DNI34321E, coche_rojo
- STRING: cadena. Por ejemplo: "coche rojo", "carácter \" especial escapado"
- EXTERNAL-ADDRESS: dirección externa.
- FACT-ADDRESS: dirección de hecho.
- Nombre de instancia
- INSTANCE-ADDRESS: dirección de instancia.



Tipos de campos:

- **Monocampo:** está compuesto de un valor primitivo. Por ejemplo: Perro, 45, "Juan Manuel"
- **Multicampo:** está compuesto de cero o más valores primitivos agrupados entre paréntesis. Por ejemplo: (), (Perro), (Perro, 45, "Juan Manuel")

Valores de los campos:

- **Constante:** (Perro, 45, "Juan Manuel")
- **Variable:**
 - ?<nombre>: referencia a un monocampo. Ámbito local.
 - \$?<nombre>: referencia a un multicampo. Ámbito local.
 - ?*<nombre>*: referencia a un monocampo o a un multicampo. Ámbito global.



Tipos:

- Órdenes: ejecutan una acción.
- Funciones: devuelven un valor.

Definición:

- Del sistema. Por ejemplo:
 - Tipos de datos: `integerp`, `floatp`, `numberp`, `symbolp`, `stringp`, `lexemep`
 - Booleana: `eq`, `neq`, `and`, `or`, `not`
 - Lógica mates: `=`, `<>`, `<`, `<=`, `>`, `>=`
 - Operaciones: `+`, `-`, `*`, `/`, `div`, `mod`, `sqrt`, `**`, `round`, `abs`, `max`, `min`
- Definidas por el usuario

Llamada mediante notación prefija

`(funcion argumentos+)`

`(+ (* 3 4) 8)`



Modifican el entorno de CLIPS.

- `defmodule`: agrupar base de conocimiento en módulos.
- `defrule`: definir reglas.
- `deffacts`: afirmar hechos que se afirmarán cada vez que se reinicie el sistema.
- `deftemplate`: definir plantillas para hechos.
- `defglobal`: definir variables globales.
- `deffunction`: crear funciones.
- `defgeneric`: crear funciones genéricas.



Dos modalidades:

- Documentación por defecto: en todos los constructores (excepto defglobal) a continuación del nombre del constructor y entre comillas.

Por ejemplo

```
(defrule regla-1 "Regla de prueba" a  $\Rightarrow$  b)
```

- Comentario de línea: en cualquier parte del código comenzando por punto y coma.

Por ejemplo

```
; Esto es un comentario
```

Grado en Ingeniería Informática, Universidad de Córdoba

Sistemas Inteligentes

CLIPS

Tema 1: Introducción

Aurora Esteban Toscano

aestebant@uco.es

07 de marzo de 2022
