

CLIPS y FUZZY-CLIPS.

José A. Olivas

viu

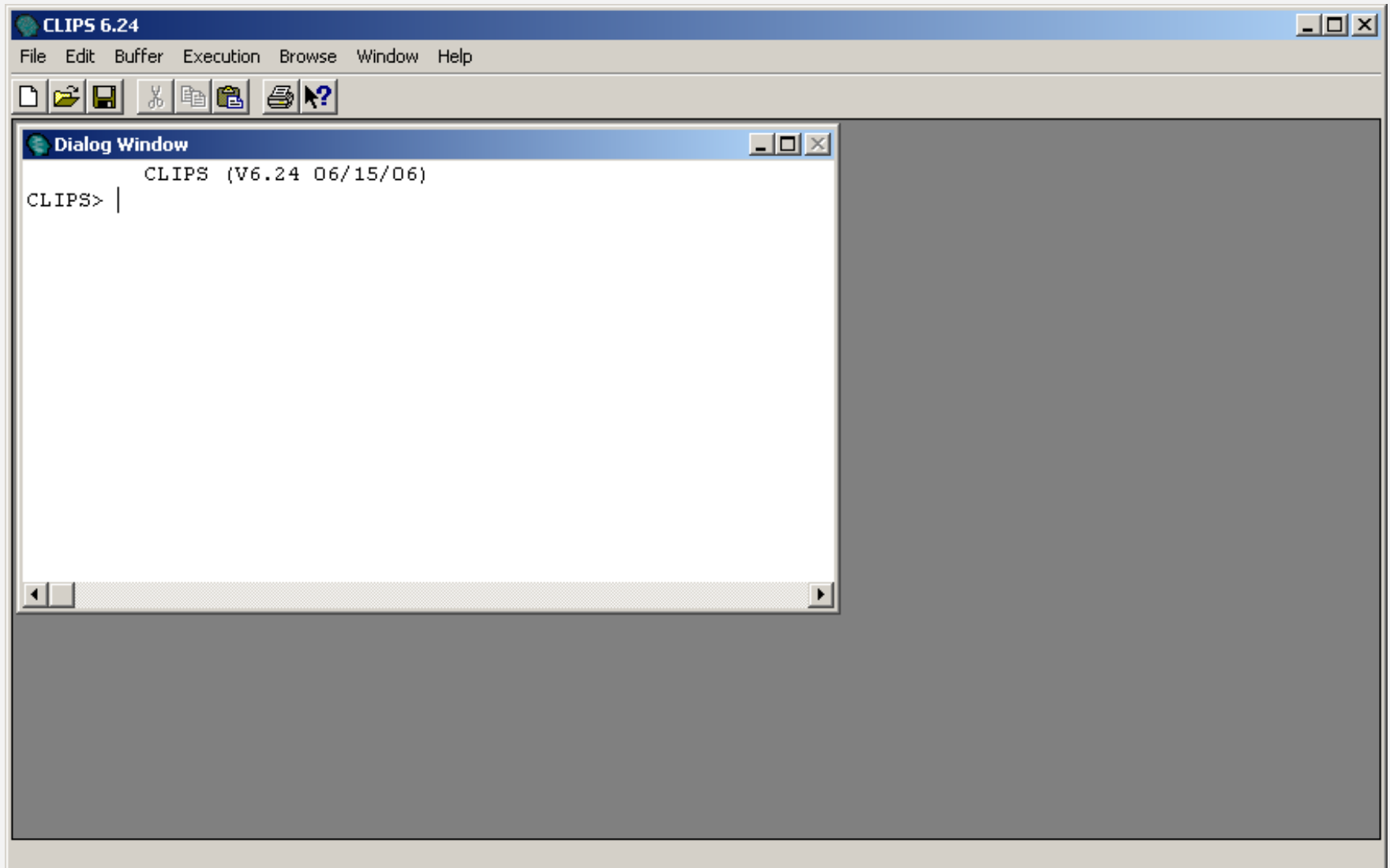
Universidad
Internacional
de Valencia

¿Qué es CLIPS?

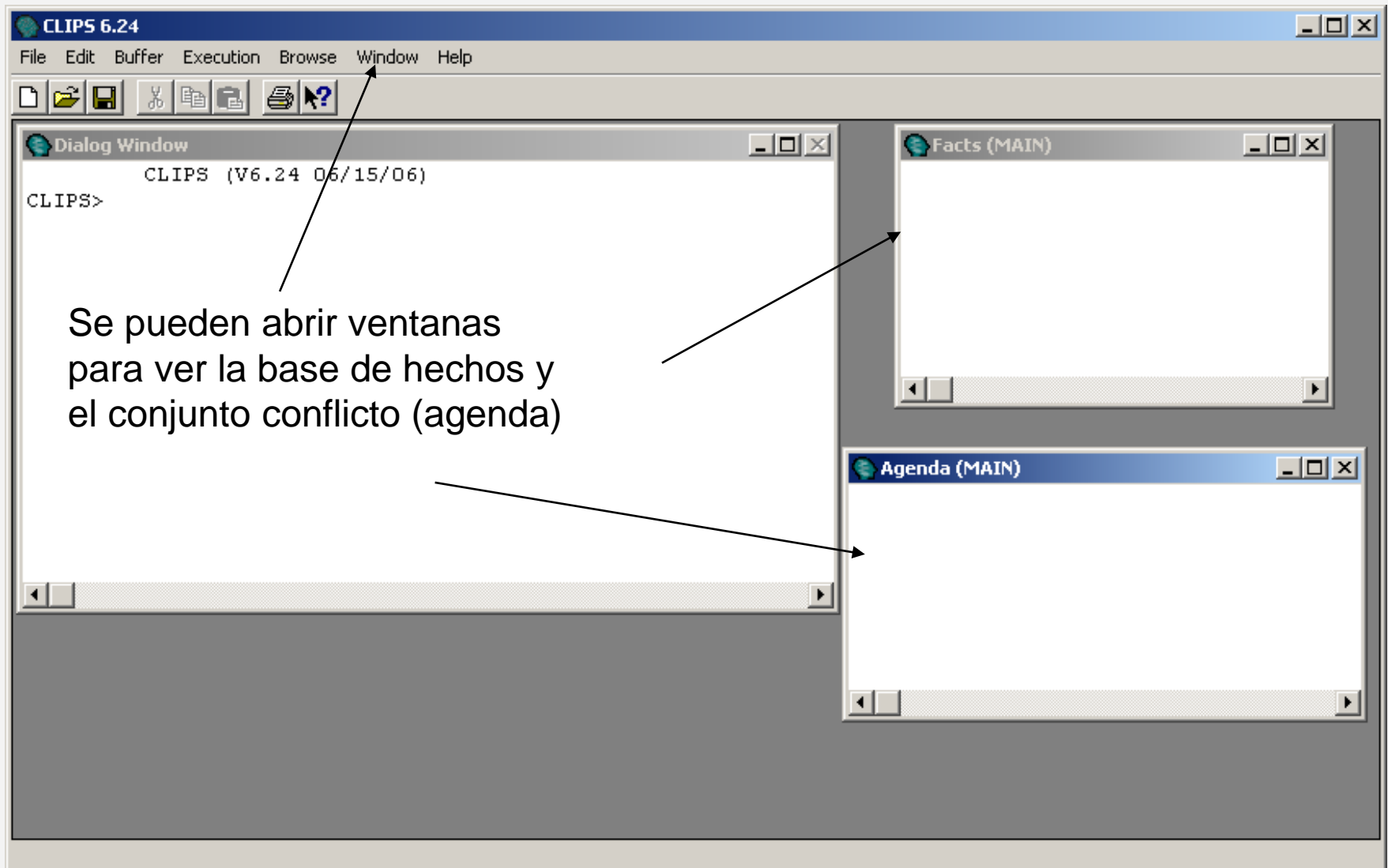
- Es una shell para la elaboración de Sistemas Expertos creada por la NASA en 1986 (v6.1)
- **C** Language Integrated **P**roduction **S**ystem
- Sistema de producción con encadenamiento hacia delante
- Disponible para diversas plataformas y fácilmente embebible en aplicaciones C, C++,...
- Sitio de descarga:

<http://www.clipsrules.net/>

Interfaz de entrada a CLIPS - I



Interfaz de entrada a CLIPS - II



Nomenclatura básica

- CLIPS llama a los hechos *facts*
- Las reglas las denomina *rules*
- El conjunto conflicto es la *agenda*
- Para razonar o inferir se utiliza *run*
- Las expresiones de CLIPS están basadas en el lenguaje LISP, esto es, se han de encerrar entre paréntesis ‘(’, ‘)’

Para salir de CLIPS se pondrá:

CLIPS> (exit) (o se cerrará la ventana como es habitual)

Base de Hechos – I

- Un HECHO es información relevante que describe un evento o característica del entorno donde ha de trabajar el S.E.
- Pueden describirse con un campo o varios. Si tienen varios campos, el primero suele representar una relación entre los restantes
- Todos los hechos se almacenan en la lista de hechos (fact-list), y se alojan en la memoria de trabajo (MT)
- A cada hecho se le asigna un identificador único que indica el orden con que fue introducido en la MT (timetag)

Base de Hechos - II

Comandos para manipular hechos:

- (**facts**): Lista los hechos de la MT
 - (**assert** <hecho>): Añade un hecho a la MT
 - (**retract** <id-hecho>): Elimina un hecho de la MT
 - (**clear**): Elimina todos los hechos y reglas de la MT
 - (**reset**): Elimina todos los hechos de la MT y restaura las condiciones iniciales (f-0)
 - (**deffacts** <nombre> <comentarios>
<hecho_1>
...
<hecho_n>): Define un conjunto de hechos de una vez (puede servir para definir los hechos iniciales)
- Los hechos de **deffacts** pasan a la MT tras hacer un **reset**

Base de Hechos - III

Ejemplos de manipulación de hechos:

```
CLIPS> (reset)
CLIPS> (facts)
f-0 (initial-fact)
For a total of 1 fact.
CLIPS> (assert (clase doctorado IA))
<Fact-1>
CLIPS> (facts)
f-0 (initial-fact)
f-1 (clase doctorado IA)
For a total of 2 facts.
CLIPS> (retract 1)
CLIPS> (assert (estudiar CLIPS))
<Fact-2>
CLIPS> (facts)
f-0 (initial-fact)
f-2 (estudiar CLIPS)
For a total of 2 facts.
CLIPS> (clear)
CLIPS> (facts)
CLIPS>
```


Base de Conocimientos - I

- La base de conocimientos en CLIPS está compuesta por reglas de producción. Estas reglas se definirán según la experiencia adquirida en el proceso de adquisición del conocimiento
- Las reglas se van escribiendo y almacenando en memoria.
- Se pueden cargar desde un fichero con la opción **load**. El fichero ha de tener extensión ***.clp**. La opción **save** sirve para guardar a fichero

Base de Conocimientos - II

- La estructura de una regla es como sigue:

```
(defrule <nombre-regla>
  [<documentación opcional>]
  [(declare (salience <num>))]
  (patrón_1)
  (patrón_2)
  ...
  (patrón_N)
=>
  (acción_1)
  (acción_2)
  ...
  (acción_N)
)
```

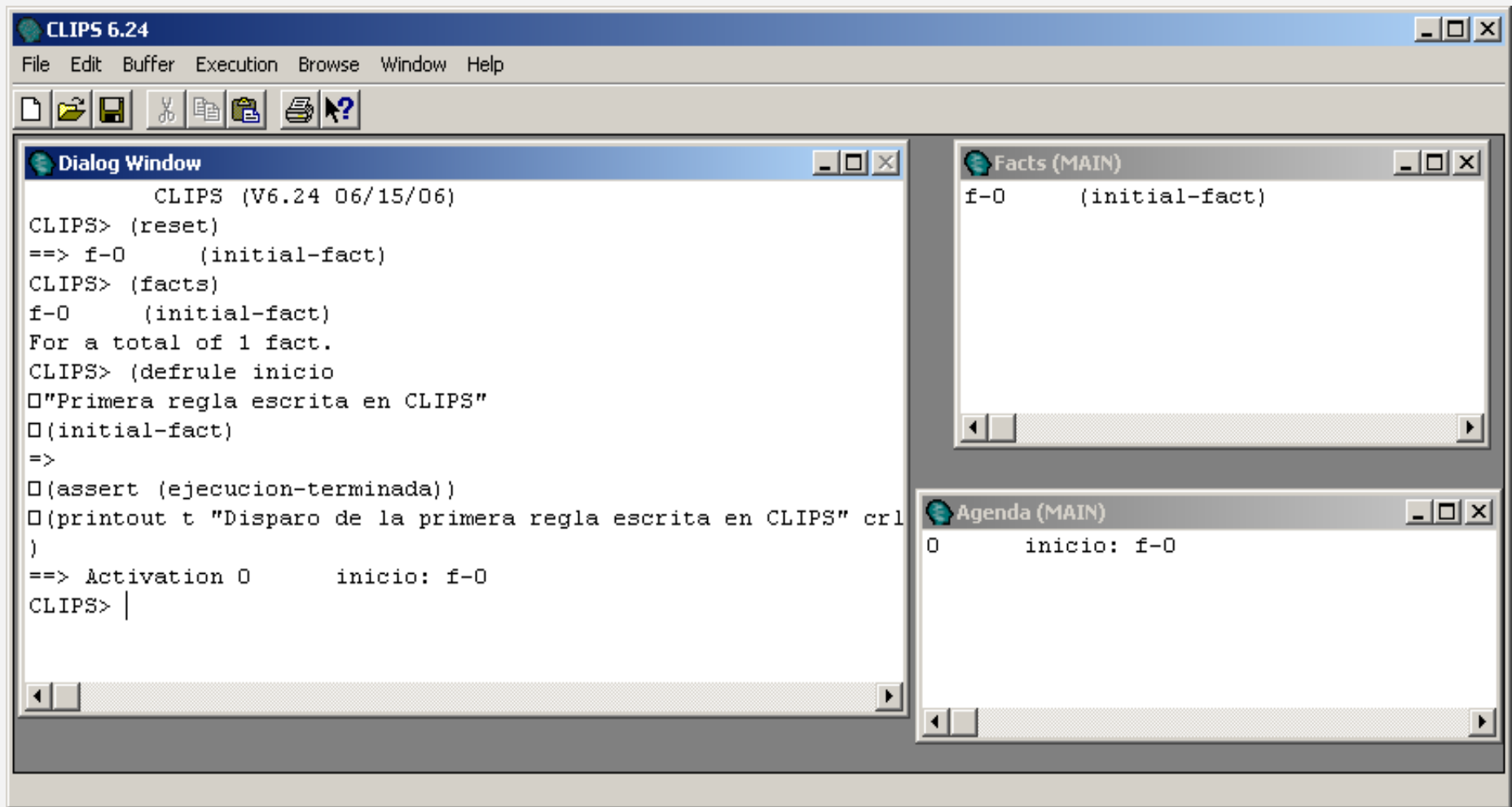
Base de Conocimientos - III

EJEMPLO

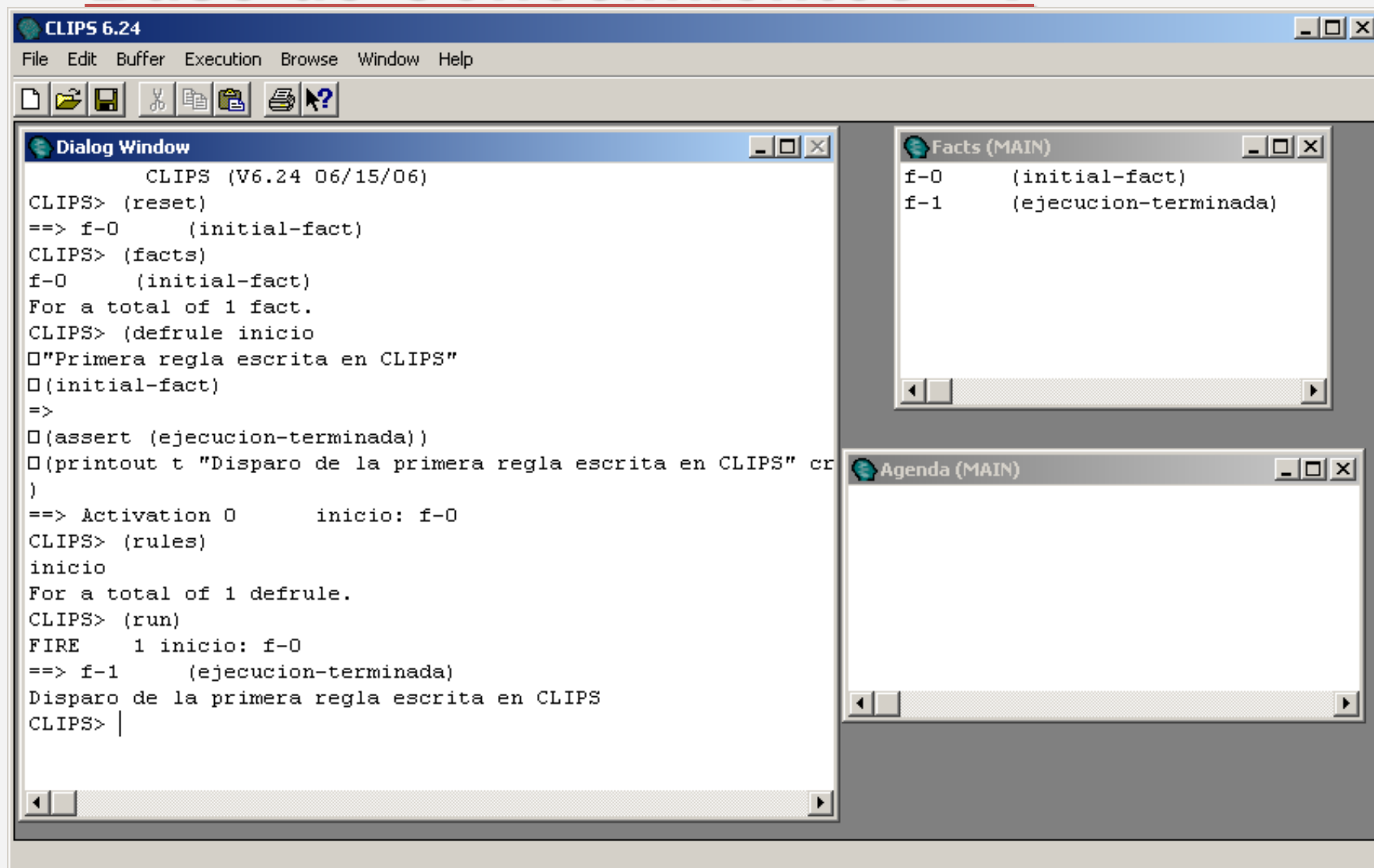
```
(defrule inicio
  "Primera regla escrita en CLIPS"
  (initial-fact)
=>
  (assert (ejecucion-terminada))
  (printout t "Disparo de la primera regla escrita en CLIPS" crlf)
)
```

```
CLIPS> (reset)
CLIPS> (facts)
f-0 (initial-fact)
For a total of 1 fact.
CLIPS> (agenda)
0 inicio: f-0
For a total of 1 activation.
CLIPS> (rules)
inicio
For a total of 1 defrule.
CLIPS> (run)
Disparo de la primera regla escrita en CLIPS
CLIPS>
```

Base de Conocimientos - IV



Base de Conocimientos - V



Uso de variables en CLIPS - I

- Se pueden usar variables y se han de poner en la parte izquierda de la regla. Para ello se especifica un comodín para sustituir un campo “?” o más de un campo “\$?”

```
(defrule inicio
  (initial-fact)
```

=>

```
(assert (alumno Juan Sanchez))
(assert (alumno Antonio))
```

)

```
(defrule variable-local-1
  (alumno ?x)
```

=>

```
(printout t "El alumno que no tiene apellidos es: " ?x crlf)
(assert (regla-variable-local-1 disparada))
```

)

```
(defrule variable-local-2
```

```
(alumno $?x)
(regla-variable-local-1 disparada)
```

=>

```
(printout t "El alumno: " $?x " esta en el sistema" crlf)
```

)

Uso de variables en CLIPS - II

```
CLIPS> (reset)
CLIPS> (facts)
f-0 (initial-fact)
For a total of 1 fact.
CLIPS> (agenda)
0      inicio: f-0
For a total of 1 activation.
CLIPS> (rules)
inicio
variable-local-1
variable-local-2
For a total of 3 defrules.
CLIPS> (run)
El alumno que no tiene apellidos es: Antonio
El alumno: (Juan Sanchez) esta en el sistema
El alumno: (Antonio) esta en el sistema
CLIPS>
```

Uso de variables en CLIPS - III

Variables globales

- Las variables globales permiten almacenar valores que son accesibles por todas las reglas y funciones de la base de conocimientos
- Pueden estar en la parte izquierda de las reglas si no son utilizadas para asignar un valor, y su cambio no activa otras reglas
- No pueden utilizarse como parámetros de funciones ni métodos

(defglobal

 ?*<nombre-variable_1>* = <valor-inicial_1>

 ?*<nombre-variable_2>* = <valor-inicial_2>

 ...

 ?*<nombre-variable_N>* = <valor-inicial_N>

)

Uso de variables en CLIPS - IV

Variables globales. Ejemplo

```
(defglobal
  ?*nombre* = Jose_Angel Olivas Varela
)

(deffacts profesores
  (profesor Jose_Angel Olivas Varela)
  (profesor Francisco_Pascual Romero Chicharro)
)

(defrule inicio
  (initial-fact)
  (profesor ?nombre ?ape1 ?ape2)
  (test (eq ?nombre ?*nombre*))
=>
  (printout t "El profesor de IA es:")
  (printout t ?nombre " " ?ape1 " " ?ape2 crlf)
)
```

Uso de variables en CLIPS - V

Variables globales. Ejemplo

```
CLIPS> (reset)
CLIPS> (facts)
f-0 (initial-fact)
f-1 (profesor Jose_Angel Olivas Varela)
f-2 (profesor Francisco_Pascual Romero Chicharro)
For a total of 3 facts.
CLIPS> (agenda)
0  inicio: f-0, f-1
For a total of 1 activation.
CLIPS> (rules)
inicio
For a total of 1 defrule.
CLIPS> (run)
El profesor de IA es: Jose_Angel Olivas Varela
CLIPS>
```

FUZZY CLIPS

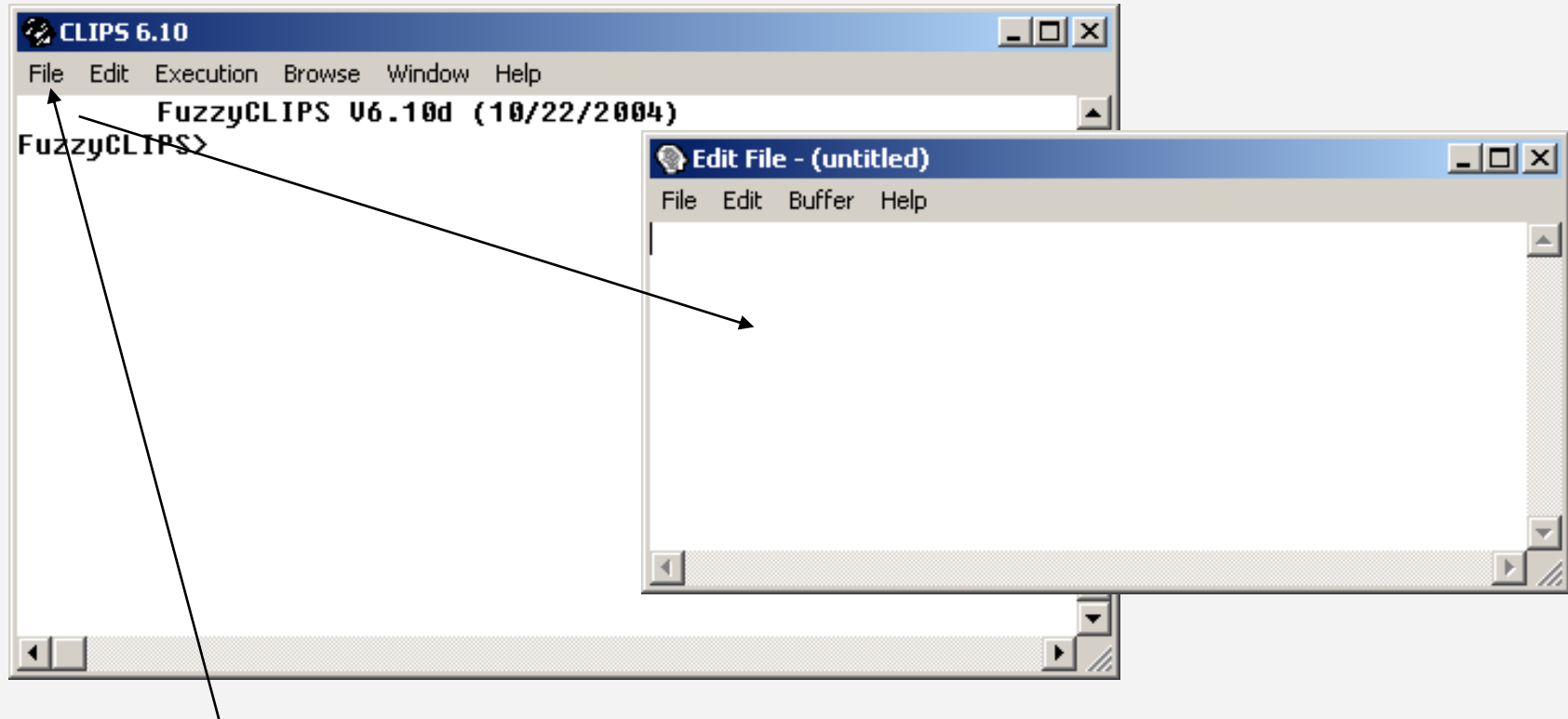
¿Qué es FuzzyCLIPS?

- Es una shell para la elaboración de Sistemas Expertos con incertidumbre que ha sido desarrollada a partir de CLIPS (6.10). Es también de libre distribución.
- Mantiene una sintaxis similar a la de CLIPS
- Contempla dos maneras de tratar la incertidumbre que a su vez se pueden combinar:
 - Incertidumbre a base de factores de certeza en reglas y hechos.
 - Incertidumbre a base de conjuntos borrosos.

<https://github.com/garydriley/FuzzyCLIPS631>

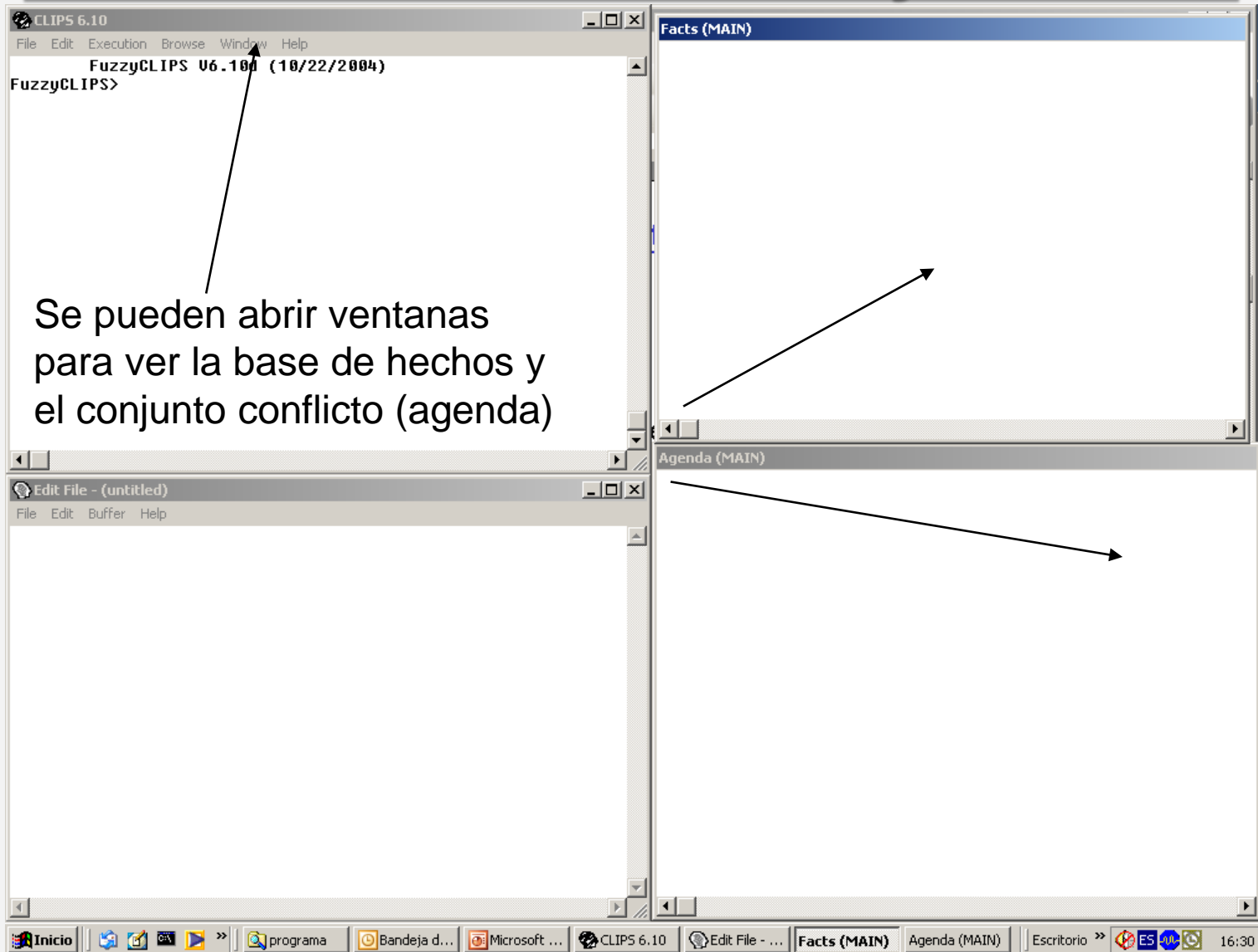
Interfaz de entrada a FuzzyCLIPS - I

Se arranca a través del programa fzclipswin.exe



Conveniente abrir desde File la opción de Editor para poder editar y cargar en memoria

Interfaz de entrada a FuzzyCLIPS - II



Incertidumbre con CF - I

- Los factores de certeza se definen en $[0, 1]$, 0 es total falsedad y 1 total verdad
- Definición de un factor de certeza para un hecho:

(fact) [CF certainty factor]

La definición del hecho es como en CLIPS

Hecho: (prediccion lluvia) (por defecto CF = 1)

Si queremos ponerle un CF:

(prediccion lluvia) CF 0.8

Incertidumbre con CF - II

- Definición de un factor de certeza para una regla:

```
(defrule nombre_regla  
  (declare (CF 0.95)) ; factor de certeza de la regla  
  (animal tipo pajaros))
```

=>

```
(assert (animal puede volar))
```

Si se tiene un hecho (animal tipo pajaros) se inferiría (animal puede volar) con CF 0.95

Incertidumbre con CF - III

- La combinación de factores de certezas entre hechos y reglas se realiza como en es habitual, mediante producto de certidumbre de antecedente y regla
- Si la regla tiene varios antecedentes y éstos diferentes factores de certeza, se toma el mínimo de ellos y éste se multiplica por el de la regla como es habitual.
- Como no se esperan CF negativos no hay combinaciones previstas para ello

Base de Hechos con CF

- Un caso ejemplo de definición de hechos con factores de certeza es el siguiente:

(deffacts hechos

 (motor no_arranca) CF 0.8

 (luces no_se_encienden) CF 0.2)

)

Base de Conocimientos con CF - I

- Un caso ejemplo de definición de base de conocimientos con factores de certeza es el siguiente:

```
(defrule regla_1
  (declare (CF 0.7))
  (or (motor no_arranca)
      (luces no_se_encienden)
  )
  =>
```

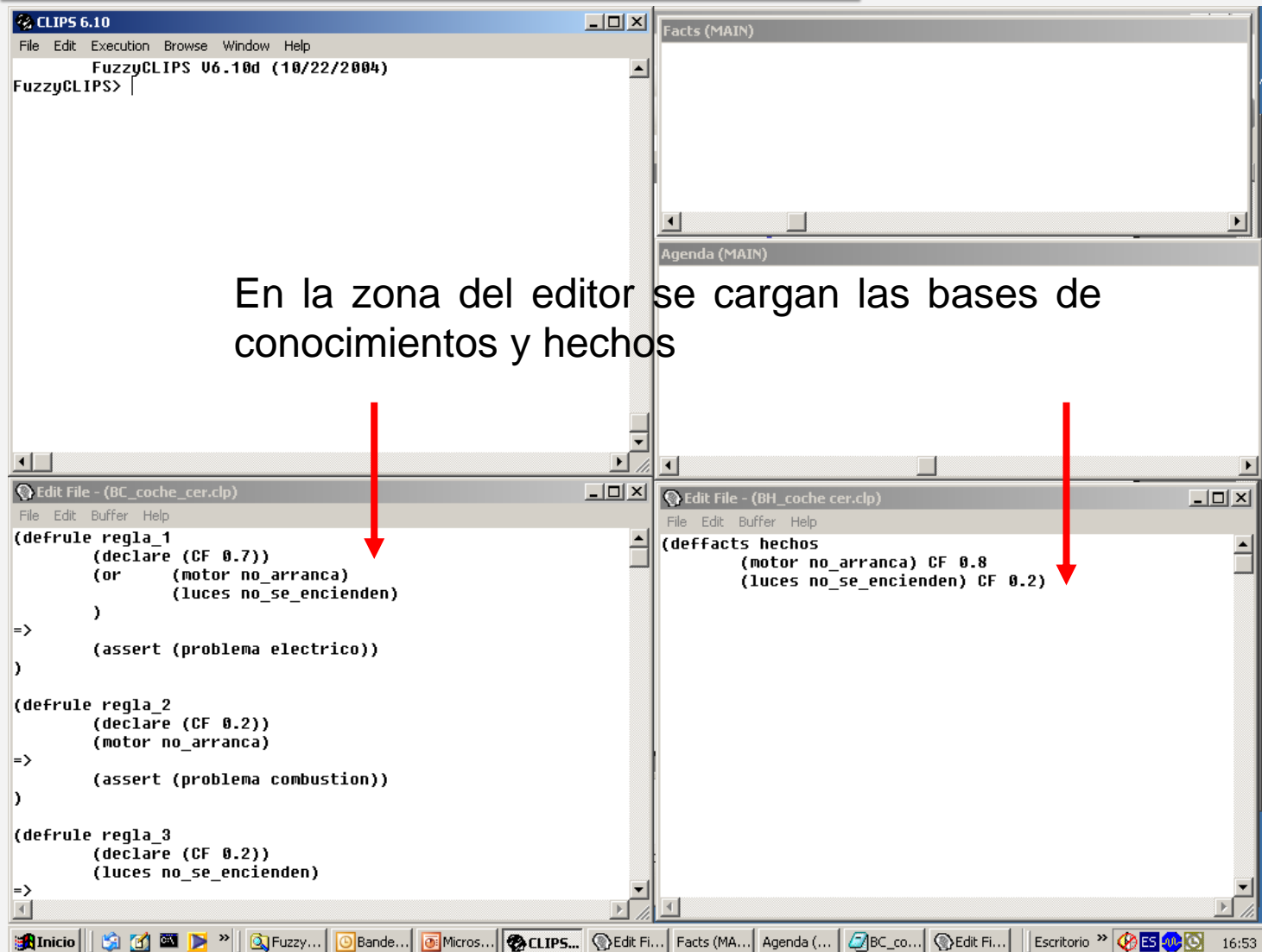
```
(assert (problema electrico))
)
```

```
(defrule regla_2
  (declare (CF 0.2))
  (motor no_arranca)
  =>
  (assert (problema combustion))
)
```

Base de Conocimientos con CF - II

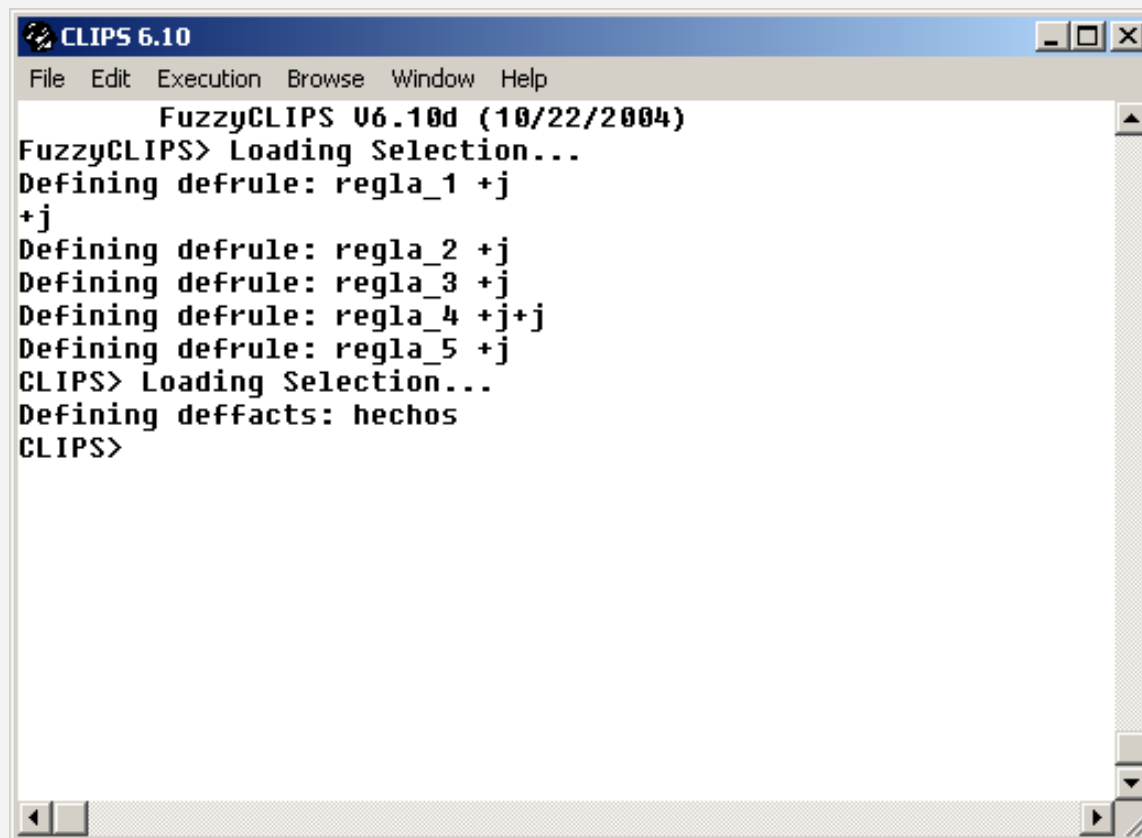
```
(defrule regla_3
  (declare (CF 0.2))
  (luces no_se_encienden)
=>
  (assert (bateria en_mal_estado))
)
(defrule regla_4
  (declare (CF 0.1))
  (bateria en_mal_estado)
  (problema combustion)
=>
  (assert (revisar en_taller))
)
(defrule regla_5
  (declare (CF 0.6))
  (problema electrico)
=>
  (assert (revisar en_taller))
)
```

Razonamiento con CF - I



Razonamiento con CF - II

Se expande la opción Buffer del editor y se elige Load Buffer para que se definan las bases de hechos y de conocimientos en el entorno:

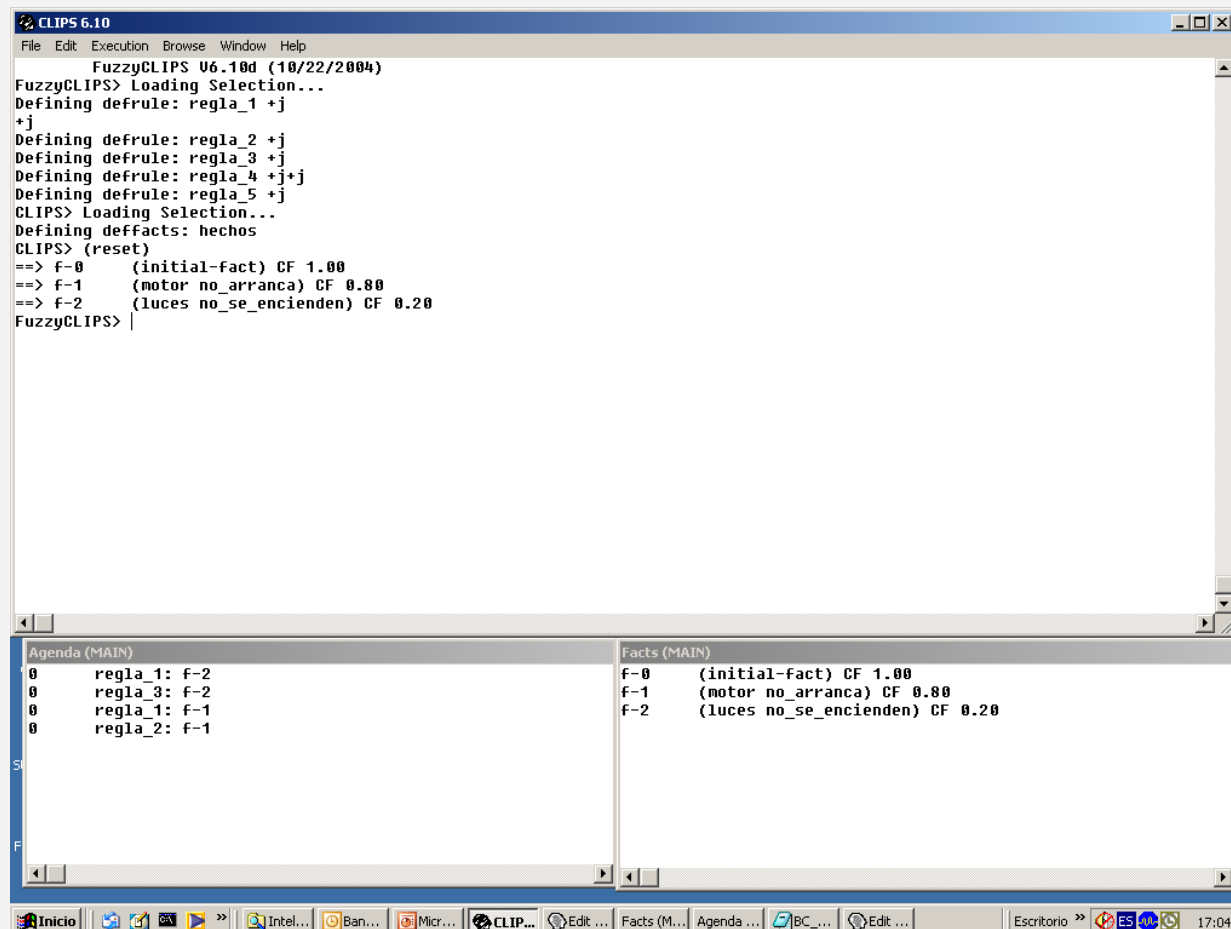


```
CLIPS 6.10
File Edit Execution Browse Window Help

      FuzzyCLIPS U6.10d (10/22/2004)
FuzzyCLIPS> Loading Selection...
Defining defrule: regla_1 +j
+j
Defining defrule: regla_2 +j
Defining defrule: regla_3 +j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 +j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS>
```

Razonamiento con CF - III

Se realiza un (reset) para inicializar los hechos y agenda.
En **Browse** de fuzzyclips es bueno escoger la opción **watch** y marcar reglas y hechos para ver lo que pasa



The screenshot shows the CLIPS 6.10 interface. The main window displays the following text:

```
CLIPS 6.10
File Edit Execution Browse Window Help
FuzzyCLIPS U6.10d (10/22/2004)
FuzzyCLIPS> Loading Selection...
Defining defrule: regla_1 +j
+j
Defining defrule: regla_2 +j
Defining defrule: regla_3 +j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 +j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS> (reset)
==> f-0 (initial-fact) CF 1.00
==> f-1 (motor no_arranca) CF 0.80
==> f-2 (luces no_se_encienden) CF 0.20
FuzzyCLIPS> |
```

Below the main window, there are two panels: 'Agenda (MAIN)' and 'Facts (MAIN)'. The 'Agenda (MAIN)' panel shows the following items:

Priority	Rule	Left Side	Right Side
0	regla_1	f-2	
0	regla_3	f-2	
0	regla_1	f-1	
0	regla_2	f-1	

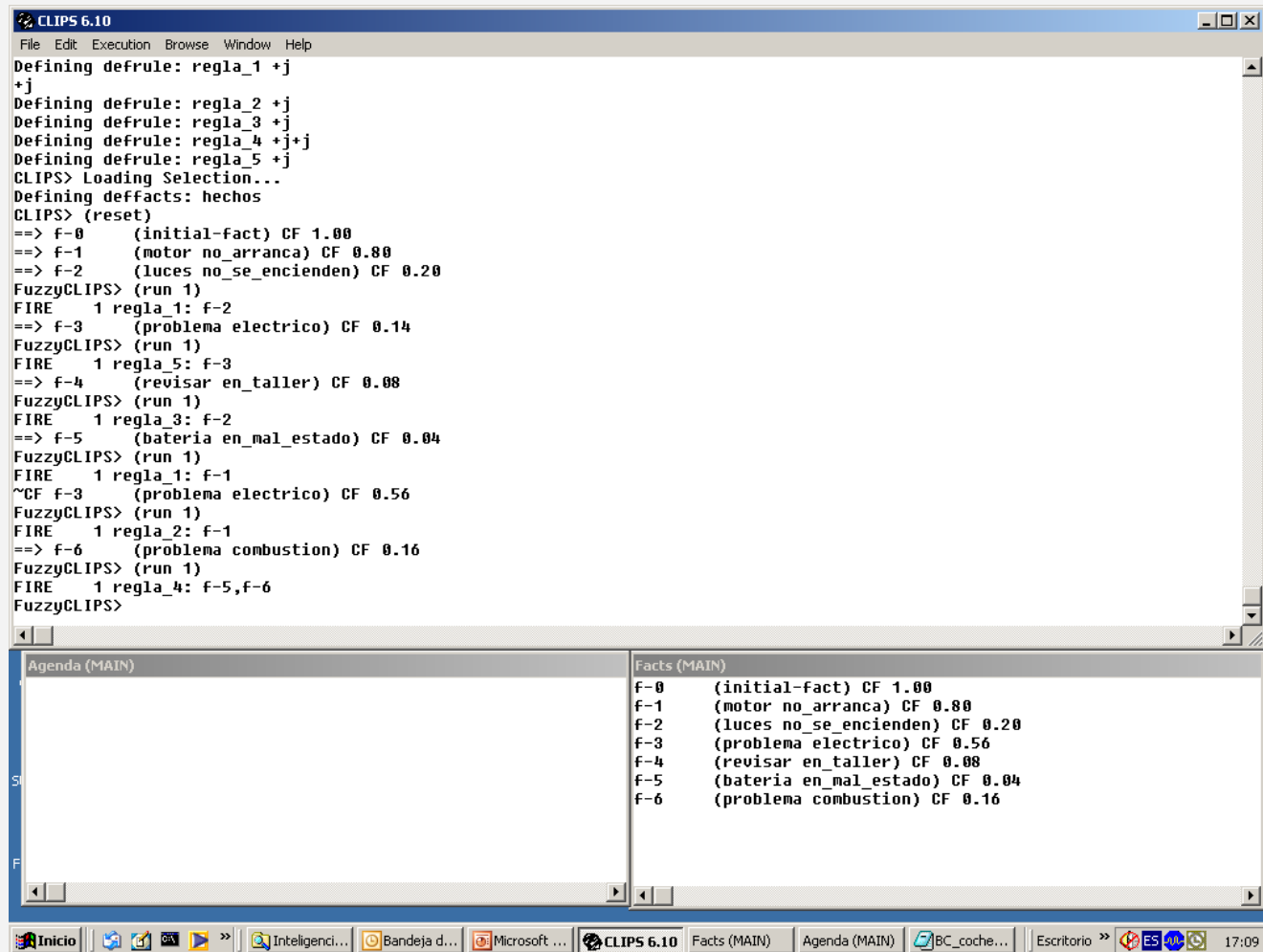
The 'Facts (MAIN)' panel shows the following facts:

Fact	Value
f-0	(initial-fact) CF 1.00
f-1	(motor no_arranca) CF 0.80
f-2	(luces no_se_encienden) CF 0.20

The taskbar at the bottom shows various application icons and the system clock indicating 17:04.

Razonamiento con CF - IV

Se hace (run 1) para ver uno a uno los disparos de reglas



```
CLIPS 6.10
File Edit Execution Browse Window Help
Defining defrule: regla_1 +j
+j
Defining defrule: regla_2 +j
Defining defrule: regla_3 +j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 +j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS> (reset)
==> f-0      (initial-fact) CF 1.00
==> f-1      (motor no_arranca) CF 0.80
==> f-2      (luces no_se_encienden) CF 0.20
FuzzyCLIPS> (run 1)
FIRE 1 regla_1: f-2
==> f-3      (problema electrico) CF 0.14
FuzzyCLIPS> (run 1)
FIRE 1 regla_5: f-3
==> f-4      (revisar en_taller) CF 0.08
FuzzyCLIPS> (run 1)
FIRE 1 regla_3: f-2
==> f-5      (bateria en_mal_estado) CF 0.04
FuzzyCLIPS> (run 1)
FIRE 1 regla_1: f-1
~CF f-3      (problema electrico) CF 0.56
FuzzyCLIPS> (run 1)
FIRE 1 regla_2: f-1
==> f-6      (problema combustion) CF 0.16
FuzzyCLIPS> (run 1)
FIRE 1 regla_4: f-5,f-6
FuzzyCLIPS>
```

Agenda (MAIN)

Facts (MAIN)

Fact	CF
f-0	(initial-fact) CF 1.00
f-1	(motor no_arranca) CF 0.80
f-2	(luces no_se_encienden) CF 0.20
f-3	(problema electrico) CF 0.56
f-4	(revisar en_taller) CF 0.08
f-5	(bateria en_mal_estado) CF 0.04
f-6	(problema combustion) CF 0.16

Inicio | Inteligenci... | Bandeja d... | Microsoft ... | CLIPS 6.10 | Facts (MAIN) | Agenda (MAIN) | BC_coche... | Escritorio » | 17:09

Incertidumbre con FL - I

- Se pueden definir conjuntos borrosos de cualquier manera, bien a base de definición de pares, bien con formas predefinidas (mirar manual).
- La forma de proceder es definir universo, etiquetas y conjuntos.
- Con estas definiciones se pueden usar en hechos como es habitual en CLIPS y lo mismo en reglas.

Incertidumbre con FL - II

- Definición de conjuntos borrosos. Ejemplo:

```
(deftemplate X1
```

```
  0 7 ;define el universo de discurso V_min y V_max
```

```
  (
```

```
    (baja (0 1) (2 0)) ; define etiquetas y conjunto
```

```
    (media (1 0) (3 1) (5 0))
```

```
    (alta (3 0) (7 1))
```

```
  )
```

```
)
```

```
(deftemplate X2
```

```
  0 7
```

```
  (
```

```
    (baja (0 1) (2 0))
```

```
    (media (1 0) (3 1) (5 0))
```

```
    (alta (3 0) (7 1))
```

```
  )
```

```
)
```

Incertidumbre con FL - III

- Definición de conjuntos borrosos. Ejemplo:

```
(deftemplate Y
  0 100
  (
    (poca (0 1) (25 1) (40 0))
    (media (30 0) (50 1) (70 0))
    (muchacha (60 0) (80 1) (100 0))
  )
)
```

Base de conocimientos con FL - I

- En este ejemplo se muestra una BC con conjuntos borrosos:

```
(defrule regla_1
  (X1 baja)
  (X2 baja)
=>
  (assert (Y poca))
)
```

```
(defrule regla_2
  (X1 baja)
  (X2 media)
=>
  (assert (Y poca))
)
```

Base de conocimientos con FL - II

- En este ejemplo se muestra una BC con conjuntos borrosos:

```
(defrule regla_3  
  (X1 media)  
  (X2 baja)
```

=>

```
  (assert (Y media)))
```

```
(defrule regla_4  
  (X1 alta)  
  (X2 media)
```

=>

```
  (assert (Y mucha)))
```

```
(defrule regla_5  
  (X1 alta)  
  (X2 alta)
```

=>

```
  (assert (Y mucha)))
```

Base de hechos con FL

- En este ejemplo se muestra una BH con conjuntos borrosos:

```
(deffacts hechos
  (X1 (4 0) (4 1) (4 0))
  (X2 (1.5 0) (1.5 1) (1.5 0))
)
```

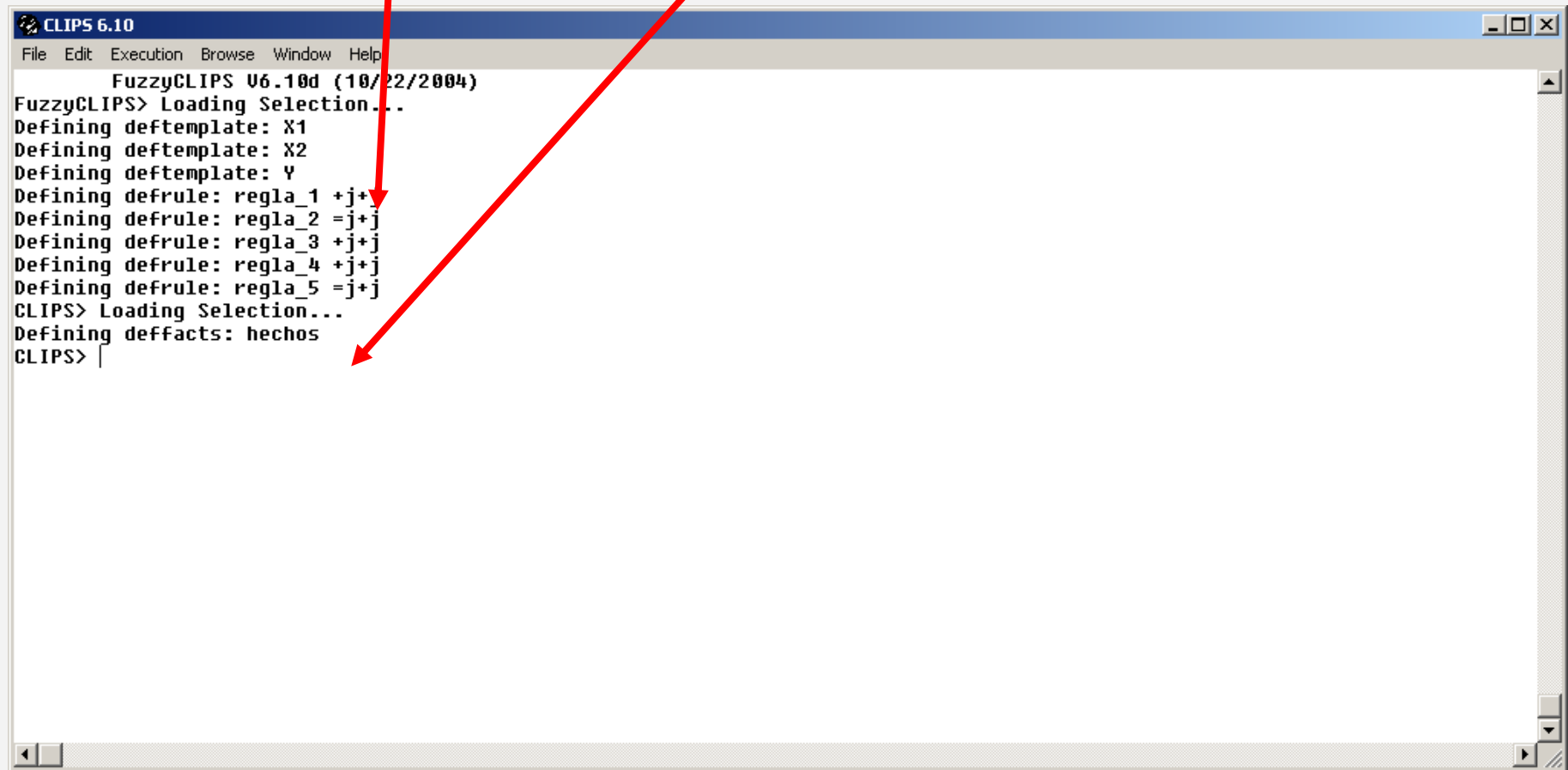
Se ha hecho una definición con singleton, sería $X1=4$ y $X2 =1.5$ lo que se observa

Razonamiento con FL - I

- Se utiliza la regla de composición de inferencias con max-min. Se utiliza por defecto. También se puede usar max-producto.
- La carga de reglas y de hechos se hace igual que como se ha comentado el caso de factores de certeza.

Razonamiento con FL - II

Cargadas reglas y hechos



```
CLIPS 6.10
File Edit Execution Browse Window Help
FuzzyCLIPS V6.10d (10/22/2004)
FuzzyCLIPS> Loading Selection...
Defining deftemplate: X1
Defining deftemplate: X2
Defining deftemplate: Y
Defining defrule: regla_1 +j+j
Defining defrule: regla_2 =j+j
Defining defrule: regla_3 +j+j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 =j+j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS> |
```


Razonamiento con FL - III

Tras un (reset) se inicializan hechos y agenda

The screenshot shows the CLIPS 6.10 interface with the following content:

```
CLIPS 6.10
File Edit Execution Browse Window Help
FuzzyCLIPS V6.10d (10/22/2004)
FuzzyCLIPS> Loading Selection...
Defining deftemplate: X1
Defining deftemplate: X2
Defining deftemplate: Y
Defining defrule: regla_1 +j+j
Defining defrule: regla_2 =j+j
Defining defrule: regla_3 +j+j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 =j+j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS> (reset)
FuzzyCLIPS>
```

The interface is divided into two panes at the bottom:

- Facts (MAIN):**

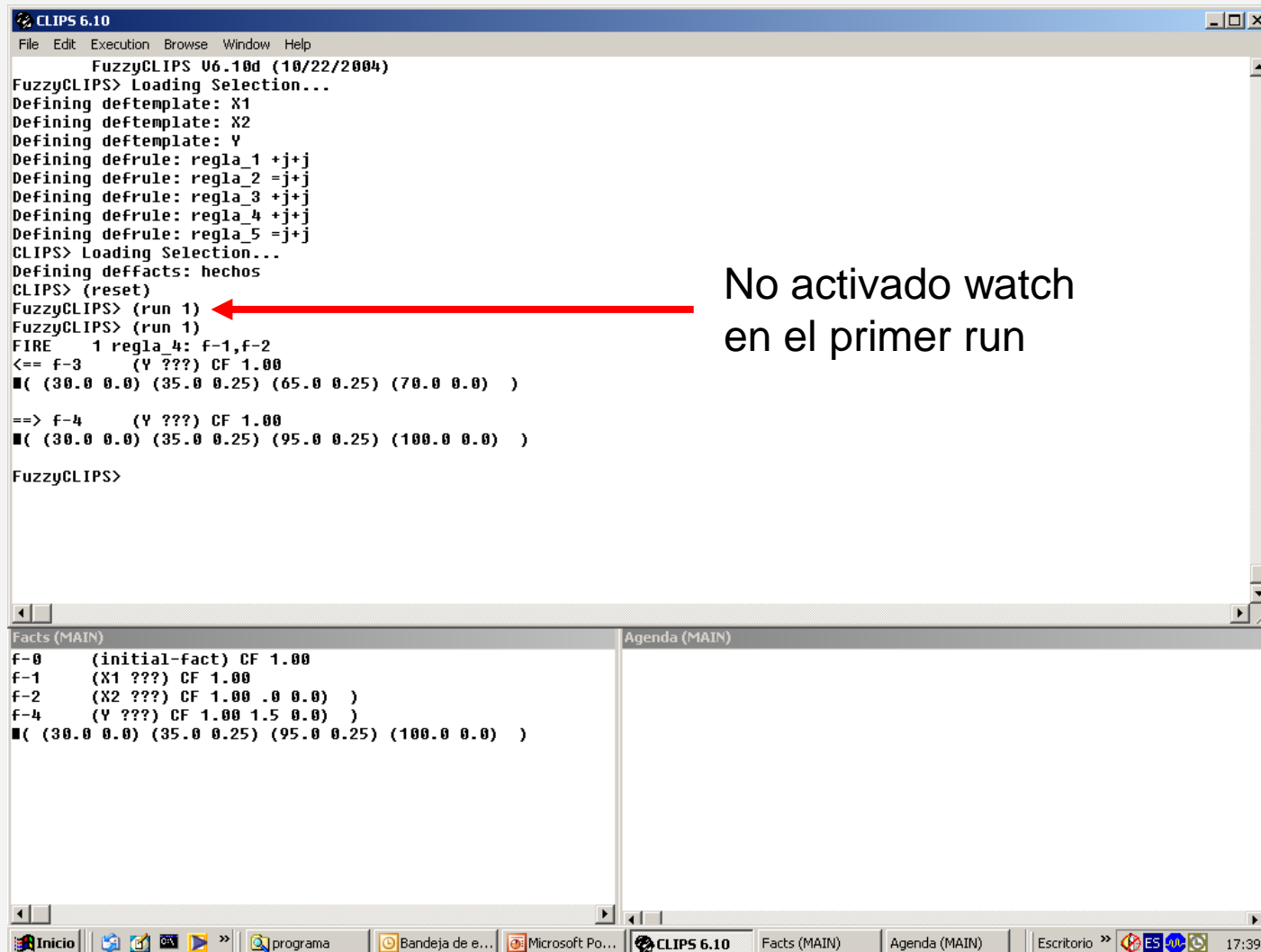
```
f-0 (initial-fact) CF 1.00
f-1 (X1 ???) CF 1.00
f-2 (X2 ???) CF 1.00 .0 0.0 )
■( (1.5 0.0) (1.5 1.0) (1.5 0.0) )
```
- Agenda (MAIN):**

```
0 regla_3: f-1,f-2
0 regla_4: f-1,f-2
```

Two red arrows point from the text "Tras un (reset) se inicializan hechos y agenda" to the "Facts (MAIN)" and "Agenda (MAIN)" panes respectively.

Razonamiento con FL - IV

Se hacen (run 1) hasta vaciar la agenda



```
CLIPS 6.10
File Edit Execution Browse Window Help
FuzzyCLIPS U6.10d (10/22/2004)
FuzzyCLIPS> Loading Selection...
Defining deftemplate: X1
Defining deftemplate: X2
Defining deftemplate: Y
Defining defrule: regla_1 +j+j
Defining defrule: regla_2 =j+j
Defining defrule: regla_3 +j+j
Defining defrule: regla_4 +j+j
Defining defrule: regla_5 =j+j
CLIPS> Loading Selection...
Defining deffacts: hechos
CLIPS> (reset)
FuzzyCLIPS> (run 1)
FuzzyCLIPS> (run 1)
FIRE 1 regla_4: f-1,f-2
<== f-3 (Y ???) CF 1.00
■( (30.0 0.0) (35.0 0.25) (65.0 0.25) (70.0 0.0) )

==> f-4 (Y ???) CF 1.00
■( (30.0 0.0) (35.0 0.25) (95.0 0.25) (100.0 0.0) )

FuzzyCLIPS>
```

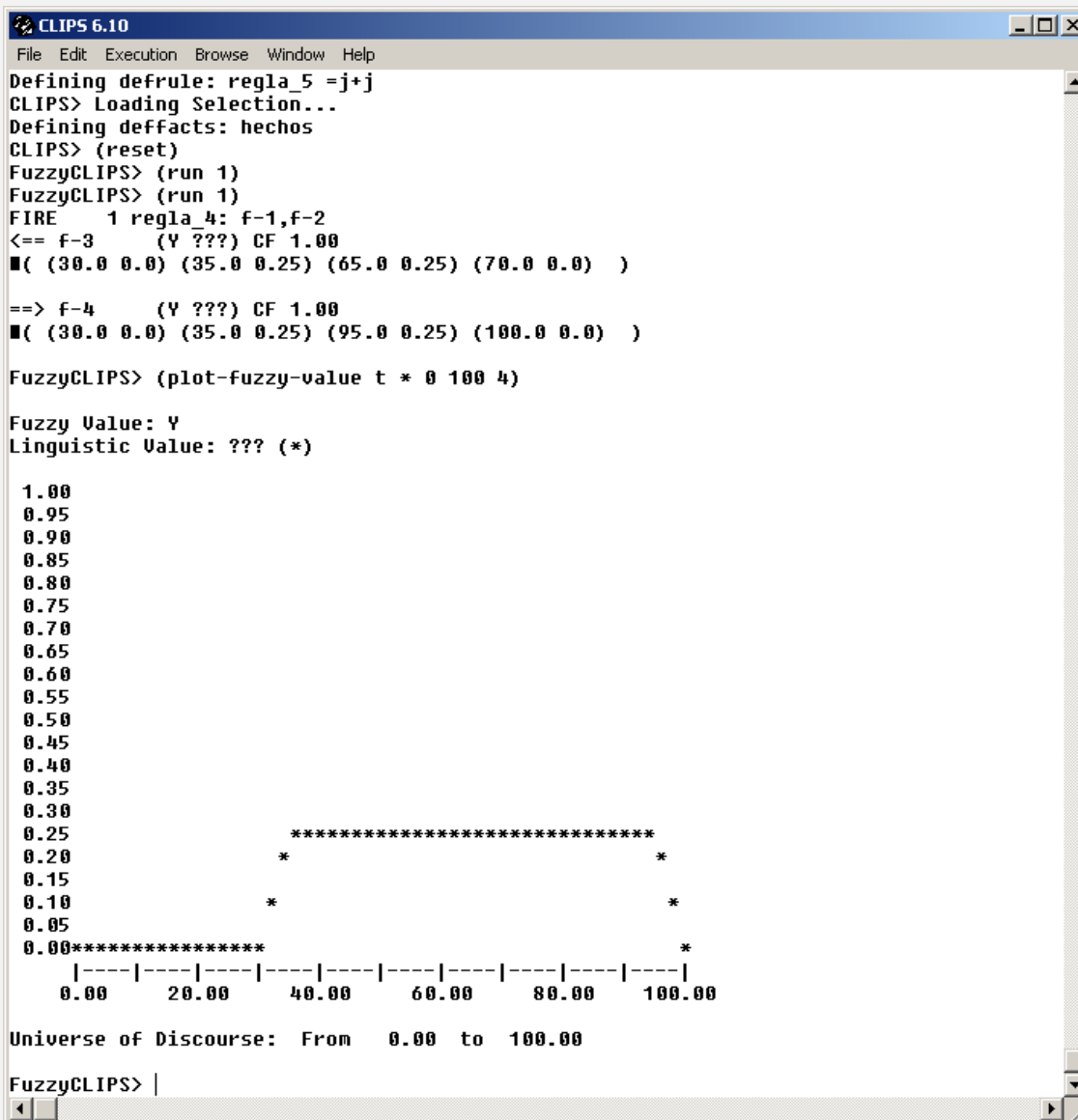
No activado watch en el primer run

Facts (MAIN)

```
f-0 (initial-fact) CF 1.00
f-1 (X1 ???) CF 1.00
f-2 (X2 ???) CF 1.00 0.0 0.0 )
f-4 (Y ???) CF 1.00 1.5 0.0 )
■( (30.0 0.0) (35.0 0.25) (95.0 0.25) (100.0 0.0) )
```

Agenda (MAIN)

Razonamiento con FL - V



Si se quiere
dibujar el conjunto
Y resultante se
llama al hecho 4

Razonamiento con FL - VI

Obtención de la desborrosificación con el máximo:

(maximum-defuzzify 4) y da 65

Con el CDG:

(moment-defuzzify 4) y da también 65

GRACIAS

viu | **Universidad**
Internacional
de Valencia