

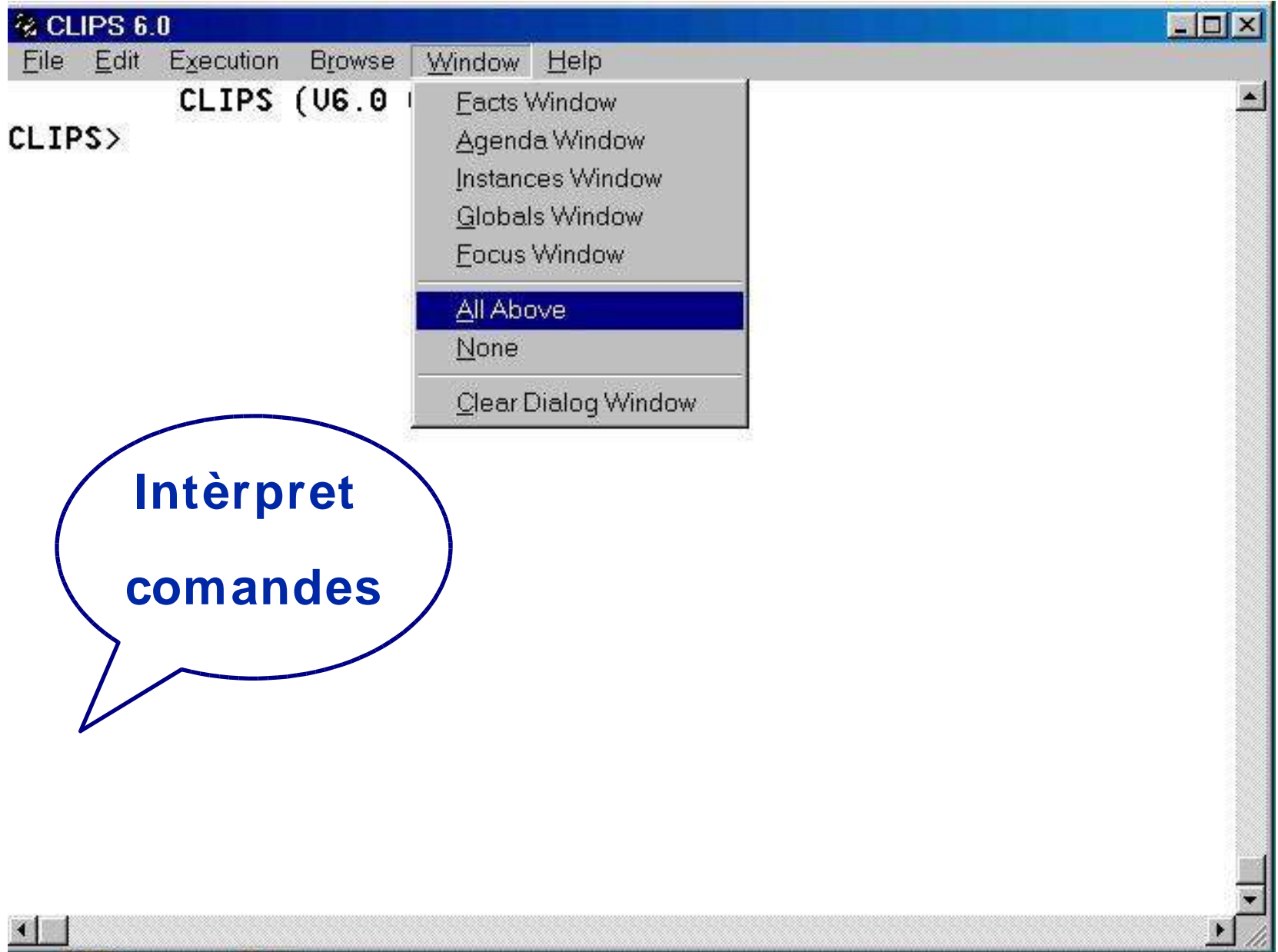
CLIPS

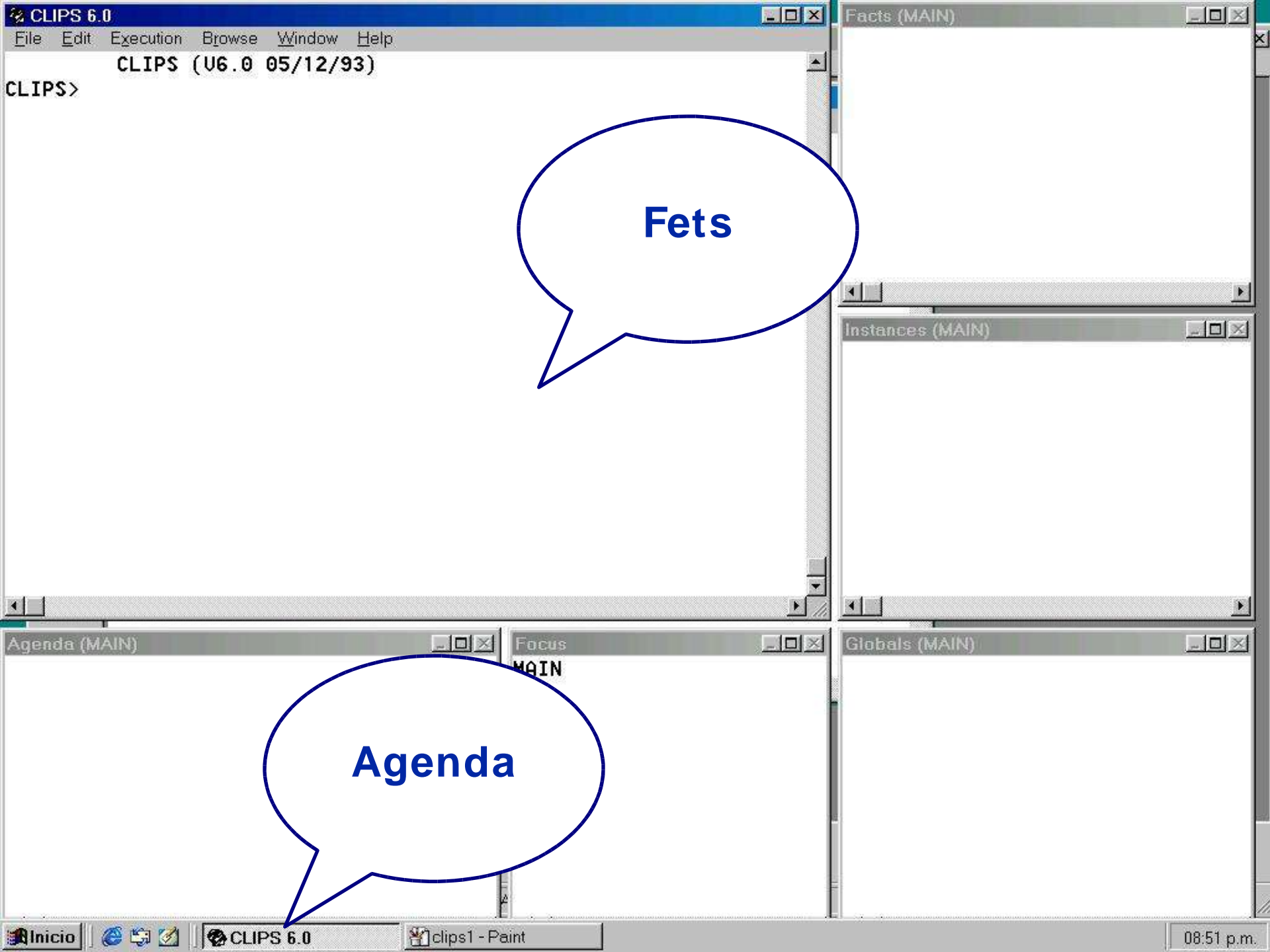
Sistema Expert de diagnosi d'avaries d'automòbils “auto.clp”

LLUÍS MÀRQUEZ

Octubre- 2001

CLIPS: Exemple

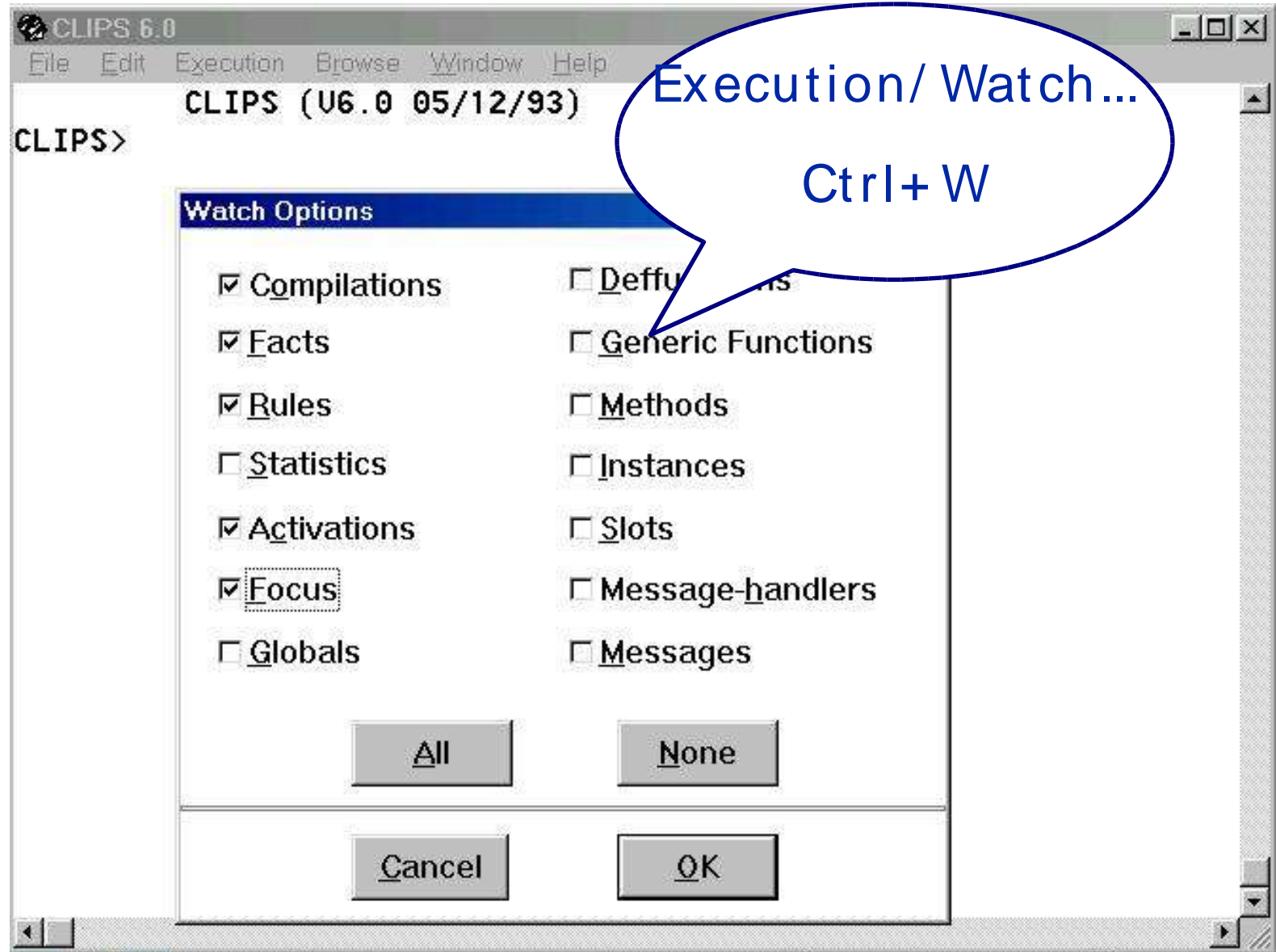




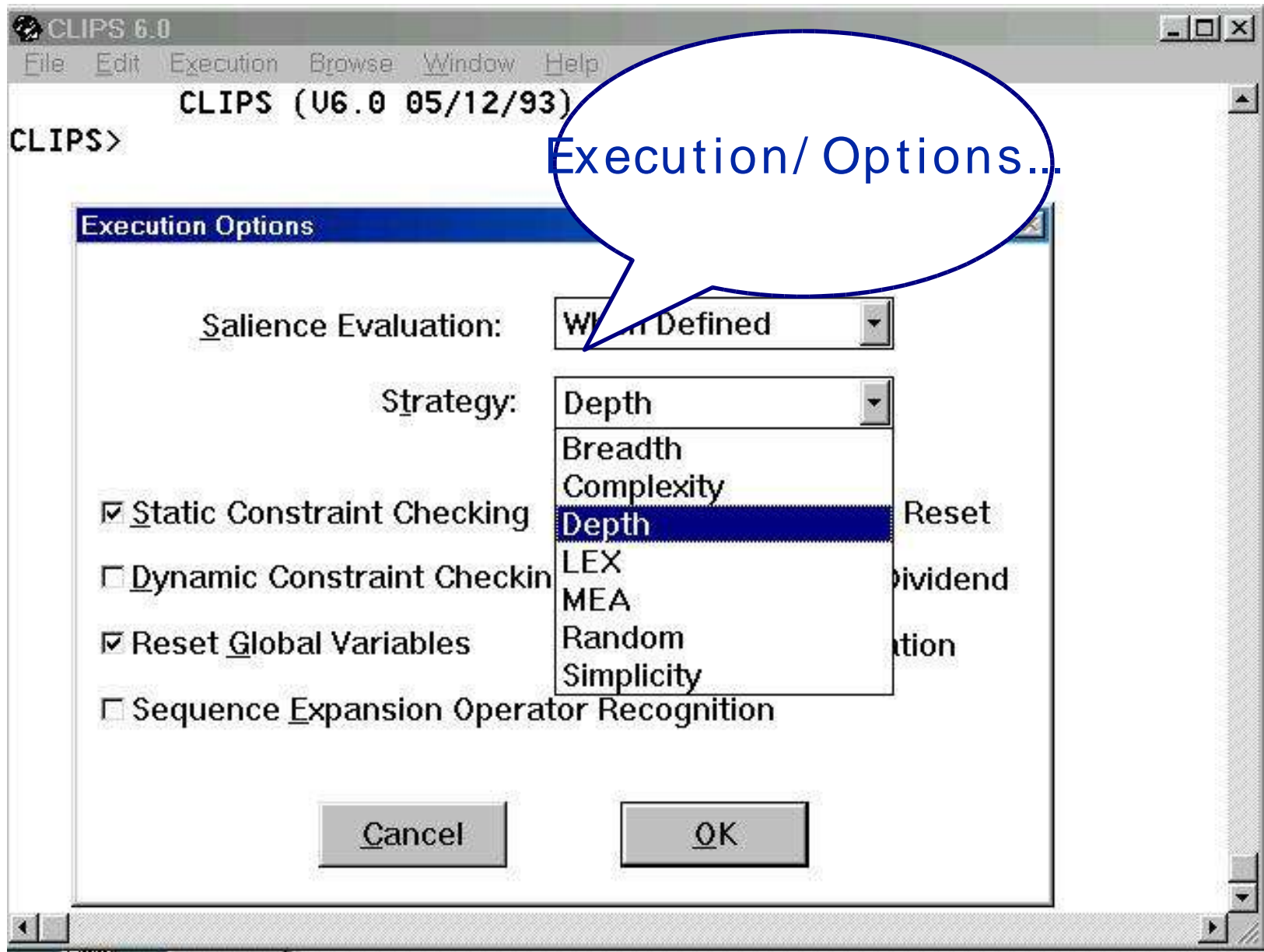
Fets

Agenda

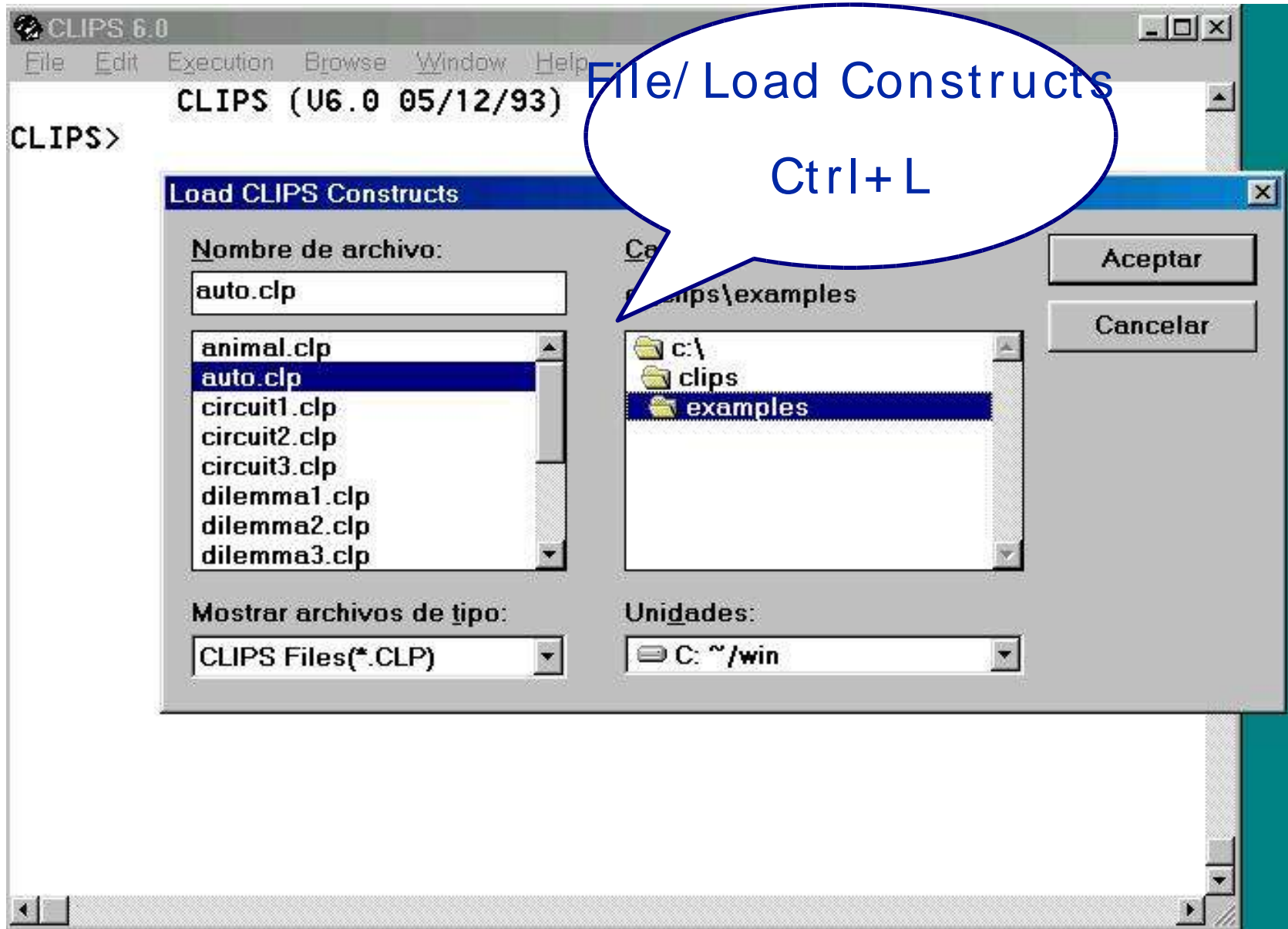
CLIPS: Exemple



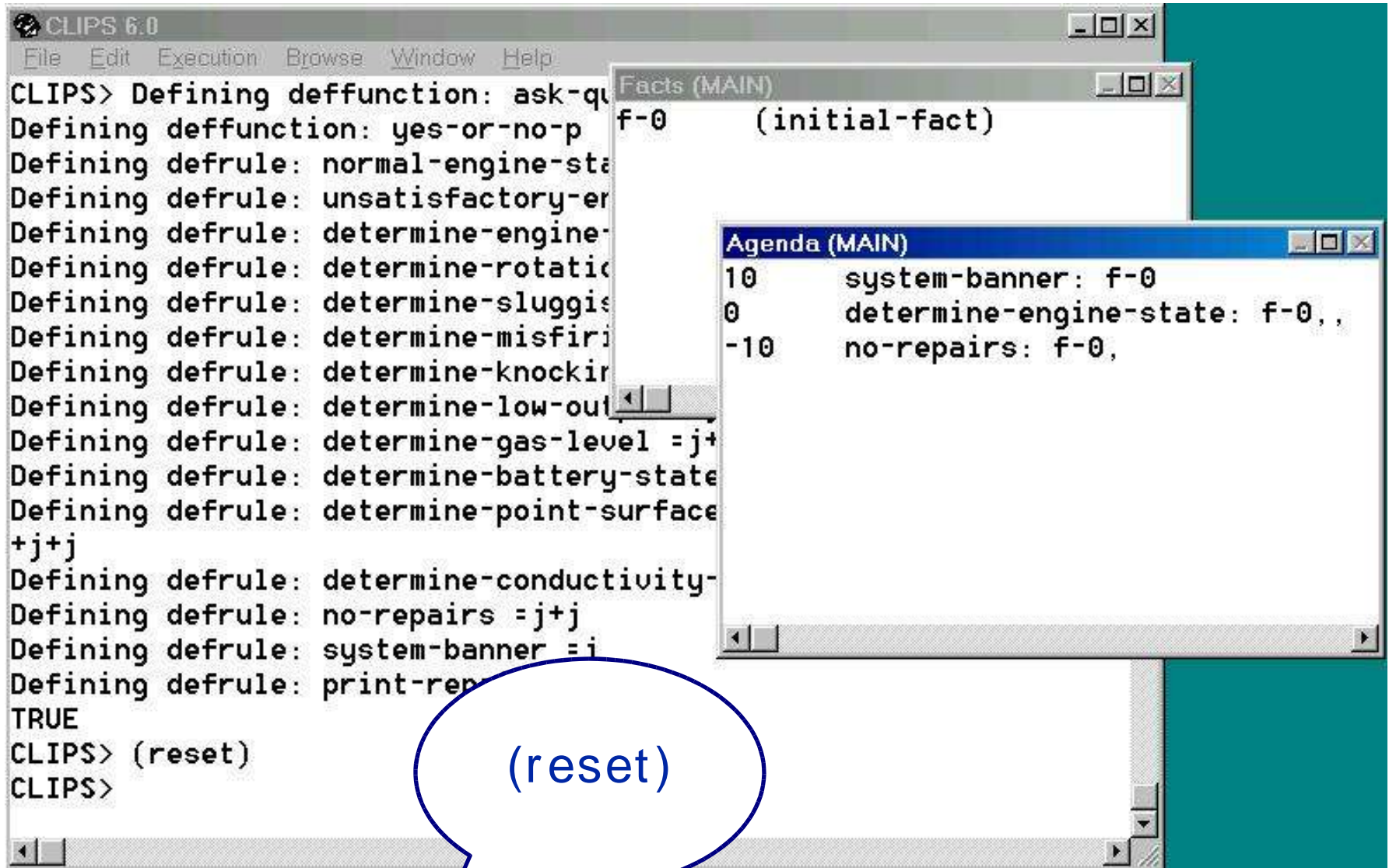
CLIPS: Example



CLIPS: Exemple



CLIPS: Example



```
CLIPS 6.0
File Edit Execution Browse Window Help
CLIPS> Defining deffunction: ask-qu
Defining deffunction: yes-or-no-p
Defining defrule: normal-engine-sta
Defining defrule: unsatisfactory-er
Defining defrule: determine-engine-
Defining defrule: determine-rotatic
Defining defrule: determine-sluggis
Defining defrule: determine-misfiri
Defining defrule: determine-knockir
Defining defrule: determine-low-out
Defining defrule: determine-gas-level = j+
Defining defrule: determine-battery-state
Defining defrule: determine-point-surface
+j+j
Defining defrule: determine-conductivity-
Defining defrule: no-repairs = j+j
Defining defrule: system-banner = i
Defining defrule: print-rep
TRUE
CLIPS> (reset)
CLIPS>
```

Facts (MAIN)

```
f-0 (initial-fact)
```

Agenda (MAIN)

```
10 system-banner: f-0
0 determine-engine-state: f-0,,
-10 no-repairs: f-0,
```

(reset)

CLIPS: Example

```
;;*****  
;;* DEFFUNCTIONS *  
;;*****  
  
(deffunction ask-question (?question $?allowed-values)  
  (printout t ?question)  
  (bind ?answer (read))  
  (if (lexemep ?answer)  
      then (bind ?answer (lowercase ?answer)))  
  (while (not (member ?answer ?allowed-values)) do  
    (printout t ?question)  
    (bind ?answer (read))  
    (if (lexemep ?answer)  
        then (bind ?answer (lowercase ?answer))))  
  ?answer)  
  
(deffunction yes-or-no-p (?question)  
  (bind ?response (ask-question ?question yes no y n))  
  (if (or (eq ?response yes) (eq ?response y))  
      then TRUE  
      else FALSE))
```

CLIPS: Example

```
(defrule no-repairs ""  
  (declare (saliency -10))  
  (not (repair ?))  
  =>  
  (assert (repair "Take your car to a mechanic.")))
```

```
;;;*****  
;;;*  STARTUP AND REPAIR RULES *  
;;;*****
```

```
(defrule system-banner ""  
  (declare (saliency 10))  
  =>  
  (printout t crlf crlf)  
  (printout t "The Engine Diagnosis Expert System")  
  (printout t crlf crlf))
```

```
(defrule print-repair ""  
  (declare (saliency 10))  
  (repair ?item)  
  =>  
  (printout t crlf crlf)  
  (printout t "Suggested Repair:")  
  (printout t crlf crlf)  
  (format t " %s%n%n%n" ?item))
```

CLIPS: Example

The screenshot displays the CLIPS 6.0 environment. The main window shows a list of rules being defined, such as 'determine-rotation-s', 'determine-sluggishness', etc. Below the list, the user has entered '(reset)' and '(run 1)'. A blue speech bubble with the text '(run 1)' is overlaid on the main window. Two smaller windows are open: 'Facts (MAIN)' showing 'f-0 (initial-fact)' and 'Agenda (MAIN)' showing a list of goals with priorities, including 'determine-engine-state: f-0,,' and 'no-repairs: f-0,,'.

```
CLIPS 6.0
File Edit Execution Browse Window Help
Defining defrule: determine-rotation-s
Defining defrule: determine-sluggishness
Defining defrule: determine-misfiring
Defining defrule: determine-knocking =
Defining defrule: determine-low-output
Defining defrule: determine-gas-level
Defining defrule: determine-battery-sta
Defining defrule: determine-point-surfa
+j+j
Defining defrule: determine-conductivi
Defining defrule: no-re
Defining defrule: sys
Defining defrule: pri
TRUE
CLIPS> (reset)
CLIPS> (run 1)

The Engine Diagnosis Expert System

CLIPS>
```

Facts (MAIN)

```
f-0 (initial-fact)
```

Agenda (MAIN)

```
0 determine-engine-state: f-0,,
-10 no-repairs: f-0,
```

CLIPS: Example

```
;;;*****  
;;;* QUERY RULES *  
;;;*****  
  
(defrule determine-engine-state ""  
  (not (working-state engine ?))  
  (not (repair ?))  
  =>  
  (if (yes-or-no-p "Does the engine start (yes/no)? ")  
      then  
      (if (yes-or-no-p "Does the engine run normally (yes/no)? ")  
          then (assert (working-state engine normal))  
          else (assert (working-state engine unsatisfactory)))  
      else |  
      (assert (working-state engine does-not-start))))
```

CLIPS: Example

The screenshot displays the CLIPS 6.0 environment with three windows:

- CLIPS 6.0 (Main Window):** Contains a menu bar (File, Edit, Execution, Browse, Window, Help) and a text area with the following content:

```
Defining defrule: determine-sluggishness = j+j
Defining defrule: determine-misfiring = j+j
Defining defrule: determine-knocking = j+j
Defining defrule: determine-low-output = j+j
Defining defrule: determine-gas-level = j+j
Defining defrule: determine-battery-status = j+j
Defining defrule: determine-point-surface = j+j
Defining defrule: determine-conductivity = j+j
Defining defrule: no-repairs = j+j
Defining defrule: system-banner = j
Defining defrule: print-repair +j
TRUE
CLIPS> (reset)
CLIPS> (run 1)

The Engine Diagnosis Expert System

CLIPS> (run 1)
Does the engine start (yes/no)? no
```
- Facts (MAIN):** A window showing the current facts in memory:

```
f-0      (initial-fact)
```
- Agenda (MAIN):** A window showing the current agenda items:

```
-10      no-repairs: f-0,      : f-0,,
```

CLIPS: Example

The screenshot displays the CLIPS 6.0 environment. The main window shows a list of rules being defined, including 'determine-mis', 'determine-kno', 'determine-low', 'determine-gas', 'determine-bat', 'determine-poi', 'determine-con', 'no-repairs = j', 'system-banner', and 'print-repair'. The user has entered '(reset)' and '(run 1)'. Below the rules, the text 'The Engine Diagnosis Expert System' is displayed, followed by the prompt 'Does the engine start (yes/no)? no' and the user's input 'no'. Two floating windows are visible: 'Facts (MAIN)' and 'Agenda (MAIN)'. The 'Facts (MAIN)' window shows two facts: 'f-0 (initial-fact)' and 'f-1 (working-state engine does-not-start)'. The 'Agenda (MAIN)' window shows a single agenda item: '0 determine-rotation-state: f-1,,'. The 'Facts (MAIN)' window has a blue border and a blue highlight around the second fact. The 'Agenda (MAIN)' window also has a blue border and a blue highlight around the first agenda item.

```
CLIPS 6.0
File Edit Execution Browse Window Help
Defining defrule: determine-mis
Defining defrule: determine-kno
Defining defrule: determine-low
Defining defrule: determine-gas
Defining defrule: determine-bat
Defining defrule: determine-poi
+j+j
Defining defrule: determine-con
Defining defrule: no-repairs = j
Defining defrule: system-banner
Defining defrule: print-repair
TRUE
CLIPS> (reset)
CLIPS> (run 1)

The Engine Diagnosis Expert System

CLIPS> (run 1)
Does the engine start (yes/no)? no
CLIPS>
```

Facts (MAIN)

```
f-0 (initial-fact)
f-1 (working-state engine does-not-start)
```

Agenda (MAIN)

```
0 determine-rotation-state: f-1,,
-10 no-repairs: f-0,
```

CLIPS: Example

```
(defrule determine-rotation-state ""
  (working-state engine does-not-start)
  (not (rotation-state engine ?))
  (not (repair ?))
  =>
  (if (yes-or-no-p "Does the engine rotate (yes/no)? ")
      then
      (assert (rotation-state engine rotates))
      (assert (spark-state engine irregular-spark))
      else
      (assert (rotation-state engine does-not-rotate))
      (assert (spark-state engine does-not-spark))))
```

CLIPS: Example

```
CLIPS 6.0
File Edit Execution Browse Window Help
Defining defrule: determine-low
Defining defrule: determine-gas
Defining defrule: determine-bat
Defining defrule: determine-po
+j+j
Defining defrule: determine-con
Defining defrule: no-repairs = j
Defining defrule: system-banner
Defining defrule: print-repair
TRUE
CLIPS> (reset)
CLIPS> (run 1)

Facts (MAIN)
f-0 (initial-fact)
f-1 (working-state engine does-not-start)
f-2 (rotation-state engine rotates)
f-3 (spark-state engine irregular-spark)

Agenda (MAIN)
0 determine-point-surface-state: f-
0 determine-gas-level: f-1,f-2,
10 no-repairs: f-0,

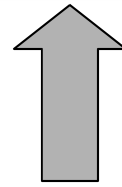
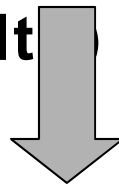
The Engine Diagnosis Expert System

CLIPS> (run 1)
Does the engine start (yes/no)? no
CLIPS> (run 1)
Does the engine rotate (yes/no)? yes
CLIPS>
```


CLIPS: Example

```
Agenda (MAIN)
0    determine-point-surface-state: f-1,f-3,
0    determine-gas-level: f-1,f-2,
-10  no-repairs: f-0,
```

(set- strategy **breadth**)



(set- strategy **depth**)

```
Agenda (MAIN)
0    determine-gas-level: f-1,f-2,
0    determine-point-surface-state: f-1,f-3,
-10  no-repairs: f-0,
```

CLIPS: Example

```
(defrule determine-point-surface-state ""
  (or (and (working-state engine does-not-start)
           (spark-state engine irregular-spark))
       (symptom engine low-output))
  (not (repair ?))
  =>
  (bind ?response
        (ask-question "What is the surface state of the points (normal/burned/cor
                       normal burned contaminated))
  (if (eq ?response burned)
      then
      (assert (repair "Replace the points.))
    else (if (eq ?response contaminated)
             then (assert (repair "Clean the points.))))))
```

CLIPS: Example

```
CLIPS 6.0
File Edit Execution Browse Window Help
Defining defrule: determine-bat
Defining defrule: determine-poi
+j+j
Defining defrule: determine-con
Defining defrule: no-repairs =j
Defining defrule: system-banner
Defining defrule: print-repair
TRUE
CLIPS> (reset)
CLIPS> (run 1)

Facts (MAIN)
f-0      (initial-fact)
f-1      (working-state engine does-not-start)
f-2      (rotation-state engine rotates)
f-3      (spark-state engine irregular-spark)

Agenda (MAIN)
0        determine-gas-level: f-1,f-2,
-10      no-repairs: f-0,

The Engine Diagnosis Expert System

CLIPS> (run 1)
Does the engine start (yes/no)? no
CLIPS> (run 1)
Does the engine rotate (yes/no)? yes
CLIPS> (run 1)
What is the surface state of the points (normal/burned/contaminated)? normal
CLIPS>
```

CLIPS: Example

```
(defrule determine-gas-level ""  
  (working-state engine does-not-start)  
  (rotation-state engine rotates)  
  (not (repair ?))  
  =>  
  (if (not (yes-or-no-p "Does the tank have any gas in it (yes/no)? "))  
      then  
      (assert (repair "Add gas."))))
```

CLIPS: Example

The screenshot shows the CLIPS 6.0 interface. The main window contains the following text:

```
+j+j
Defining defrule: determine-con
Defining defrule: no-repairs = j
Defining defrule: system-banner
Defining defrule: print-repair
TRUE
CLIPS> (reset)
CLIPS> (run 1)

The Engine Diagnosis Expert Sys

CLIPS> (run 1)
Does the engine start (yes/no)? no
CLIPS> (run 1)
Does the engine rotate (yes/no)? yes
CLIPS> (run 1)
What is the surface state of the points (normal/burned/contaminated)? normal
CLIPS> (run 1)
Does the tank have any gas in it (yes/no)? no
CLIPS>
```

The 'Facts (MAIN)' window displays the following facts:

```
f-0 (initial-fact)
f-1 (working-state engine does-not-start)
f-2 (rotation-state engine rotates)
f-3 (spark-state engine irregular-spark)
f-4 (repair "Add gas.")
```

The 'Agenda (MAIN)' window displays the following agenda item:

```
10 print-repair: f-4
```

CLIPS: Example

The screenshot displays the CLIPS 6.0 expert system interface. The main window contains a text area with the following text:

```
The Engine Diagnosis Expert Sys  
CLIPS> (run 1)  
Does the engine start (yes/no)?  
CLIPS> (run 1)  
Does the engine rotate (yes/no)  
CLIPS> (run 1)  
What is the surface state of th  
CLIPS> (run 1)  
Does the tank have any gas in i  
CLIPS> (run 1)  
  
Suggested Repair:  
  
Add gas.  
  
CLIPS>
```

Two smaller windows are overlaid on the main window:

- Facts (MAIN)**: A window listing facts:

```
f-0 (initial-fact)  
f-1 (working-state engine does-not-start)  
f-2 (rotation-state engine rotates)  
f-3 (spark-state engine irregular-spark)  
f-4 (repair "Add gas.")
```
- Agenda (MAIN)**: A window that is currently empty.

CLIPS: Example

```
(defrule no-repairs ""  
  (declare (salience -10))  
  (not (repair ?))  
  =>  
  (assert (repair "Take your car to a mechanic.")))
```

```
;;;*****  
;;;*  STARTUP AND REPAIR RULES  *  
;;;*****
```

```
(defrule system-banner ""  
  (declare (salience 10))  
  =>  
  (printout t crlf crlf)  
  (printout t "The Engine Diagnosis Expert System")  
  (printout t crlf crlf))
```

```
(defrule print-repair ""  
  (declare (salience 10))  
  (repair ?item)  
  =>  
  | (printout t crlf crlf)  
    (printout t "Suggested Repair:")  
    (printout t crlf crlf)  
    (format t " %s%n%n%n" ?item))
```

CLIPS: Example

The screenshot displays the CLIPS 6.0 expert system interface. The main window shows a suggested repair and a series of user interactions. Two smaller windows, 'Facts (MAIN)' and 'Agenda (MAIN)', provide a detailed view of the system's internal state.

```
CLIPS 6.0
File Edit Execution Browse Window Help

Suggested Repair:

Add gas.

CLIPS> (reset)
CLIPS> (run 1)

The Engine Diagnosis Expert Sys

CLIPS> (run 1)
Does the engine start (yes/no)? no
CLIPS> (run 1)
Does the engine rotate (yes/no)? yes
CLIPS> (run 1)
What is the surface state of the points (normal/burned/contaminated)? normal
CLIPS> (run 1)
Does the tank have any gas in it (yes/no)? yes
CLIPS>
```

Facts (MAIN)

f-0	(initial-fact)
f-1	(working-state engine does-not-start)
f-2	(rotation-state engine rotates)
f-3	(spark-state engine irregular-spark)

Agenda (MAIN)

```
-10 no-repairs: f-0,
```


CLIPS: Example

The Engine Diagnosis Expert Sys

```
CLIPS> (run 1)
Does the engine start (yes/no)?
CLIPS> (run 1)
Does the engine rotate (yes/no)
CLIPS> (run 1)
What is the surface state of th
CLIPS> (run 1)
Does the tank have any gas in i
CLIPS> (run 1)
CLIPS> (run 1)
```

Suggested Repair:

Take your car to a mechanic.

```
CLIPS>
```

Facts (MAIN)

- f-0 (initial-fact)
- f-1 (working-state engine does-not-start)
- f-2 (rotation-state engine rotates)
- f-3 (spark-state engine irregular-spark)
- f-4 (repair "Take your car to a mechanic.")

Agenda (MAIN)