JessTab: Protégé con Jess

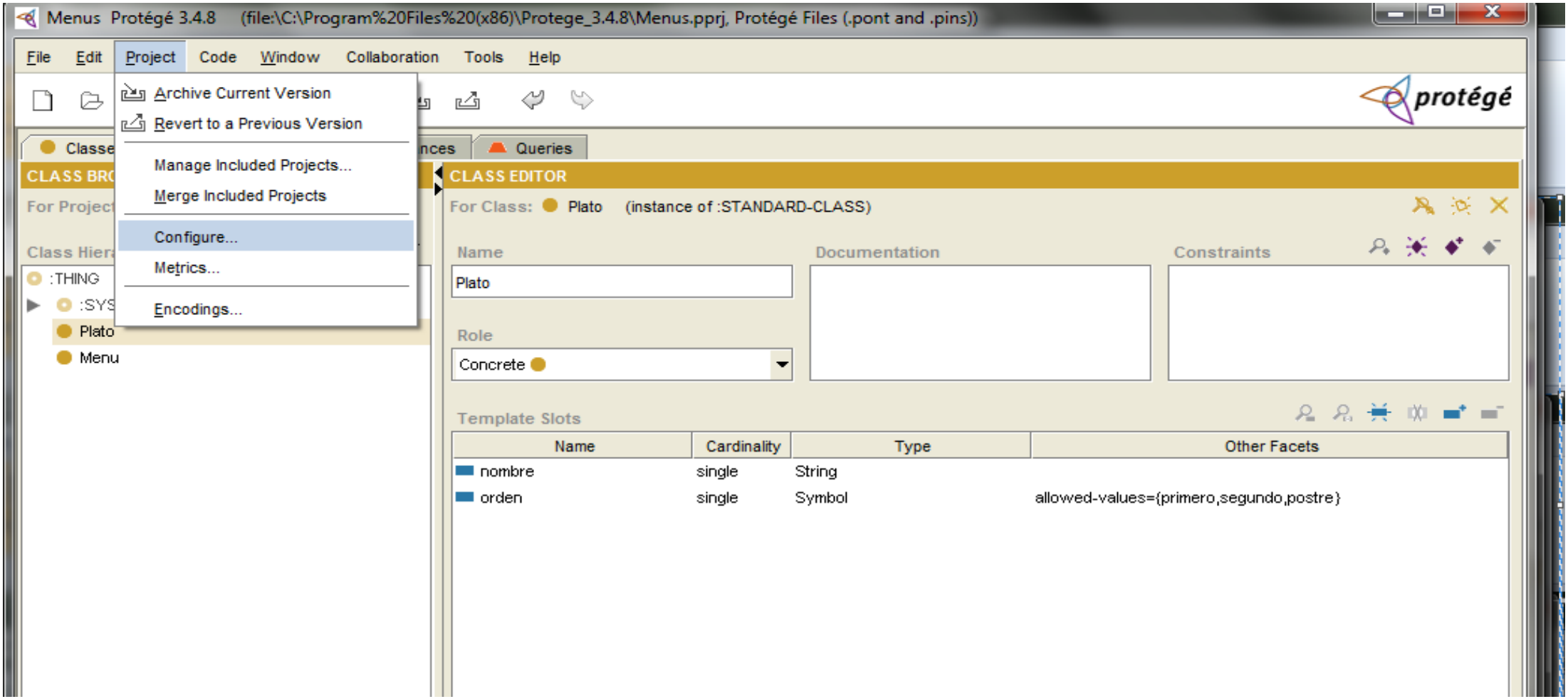
- JessTab es un puente entre Protégé y Jess
- Proporciona una consola de Jess en una pestaña de Protégé
- Permite incluir reglas Jess en las ontologías de Protégé
- Las reglas creadas pasan a formar parte de la ontología
- Se pueden crear correspondencias (mappings) entre instancias de la ontología y hechos en Jess
- Es posible manejar con Jess las ontologías y bases de conocimiento desarrolladas con Protégé
- Jess y Protégé pueden ser utilizados desde Java

Uso de JessTab en Protegé

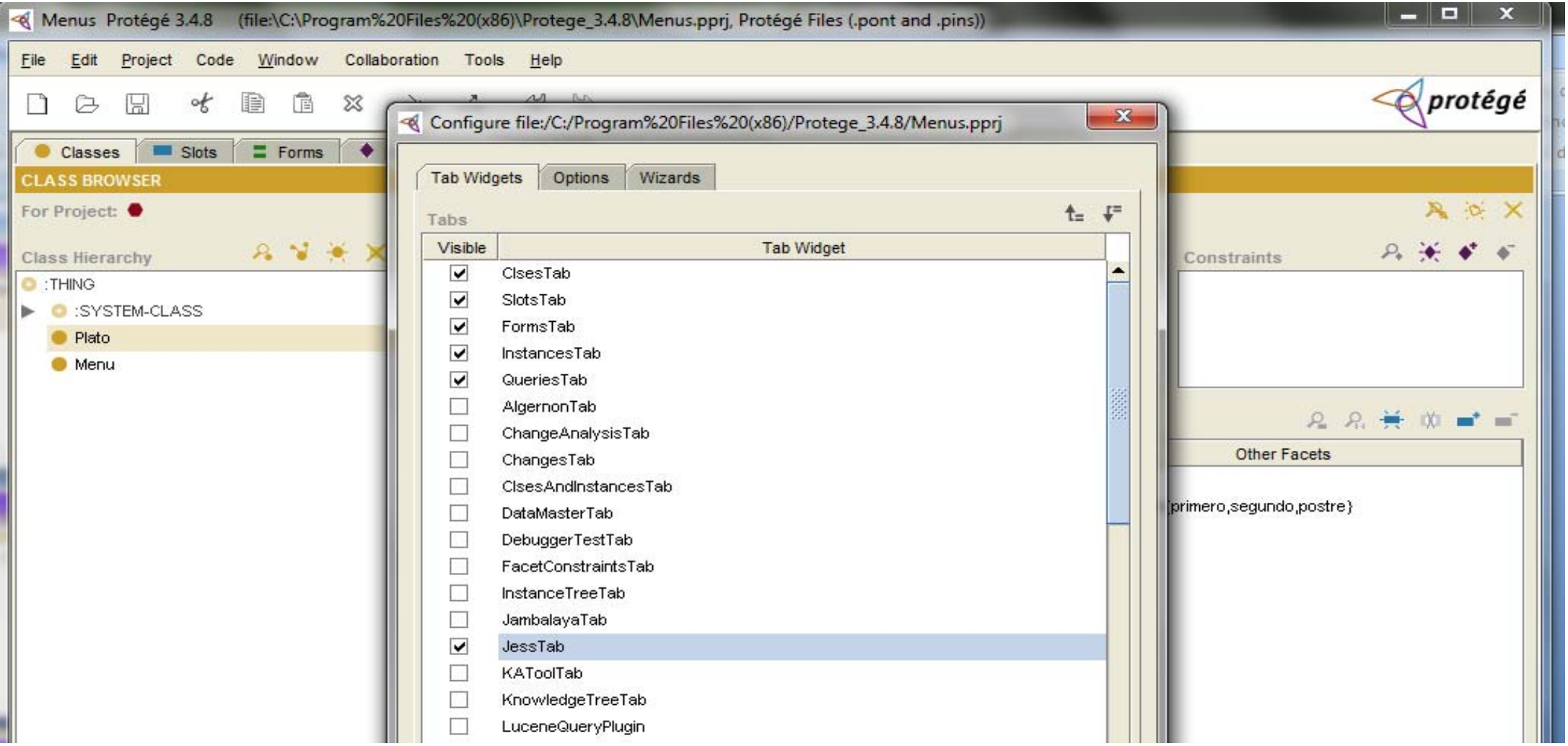
- Primero debe estar instalado Protégé 3.5 que lleva incluido JessTab
- Copiar el archivo jess.jar (incluido en la distribución de Jess)
- Pegarlo en el subdirectorio /plugins/se.liu.ida.JessTab de la instalación de Protégé 3.5. Este subdirectorio contiene ya otros dos archivos: JessTab17.jar y plugin.properties.

Configurar Protégé para incluir Jess

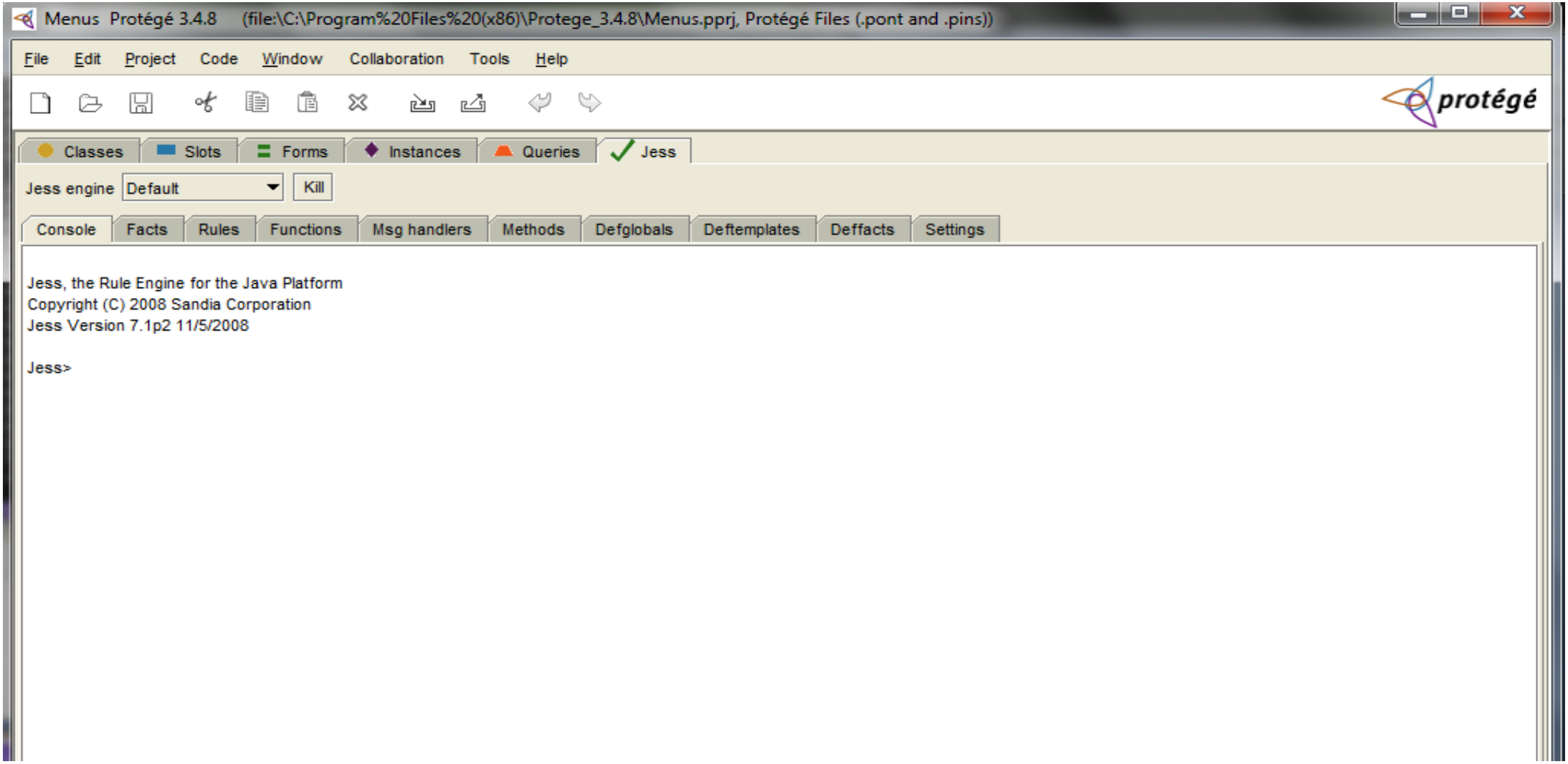
- Hay que modificar la configuración en el proyecto de Protégé donde queramos usar Jess



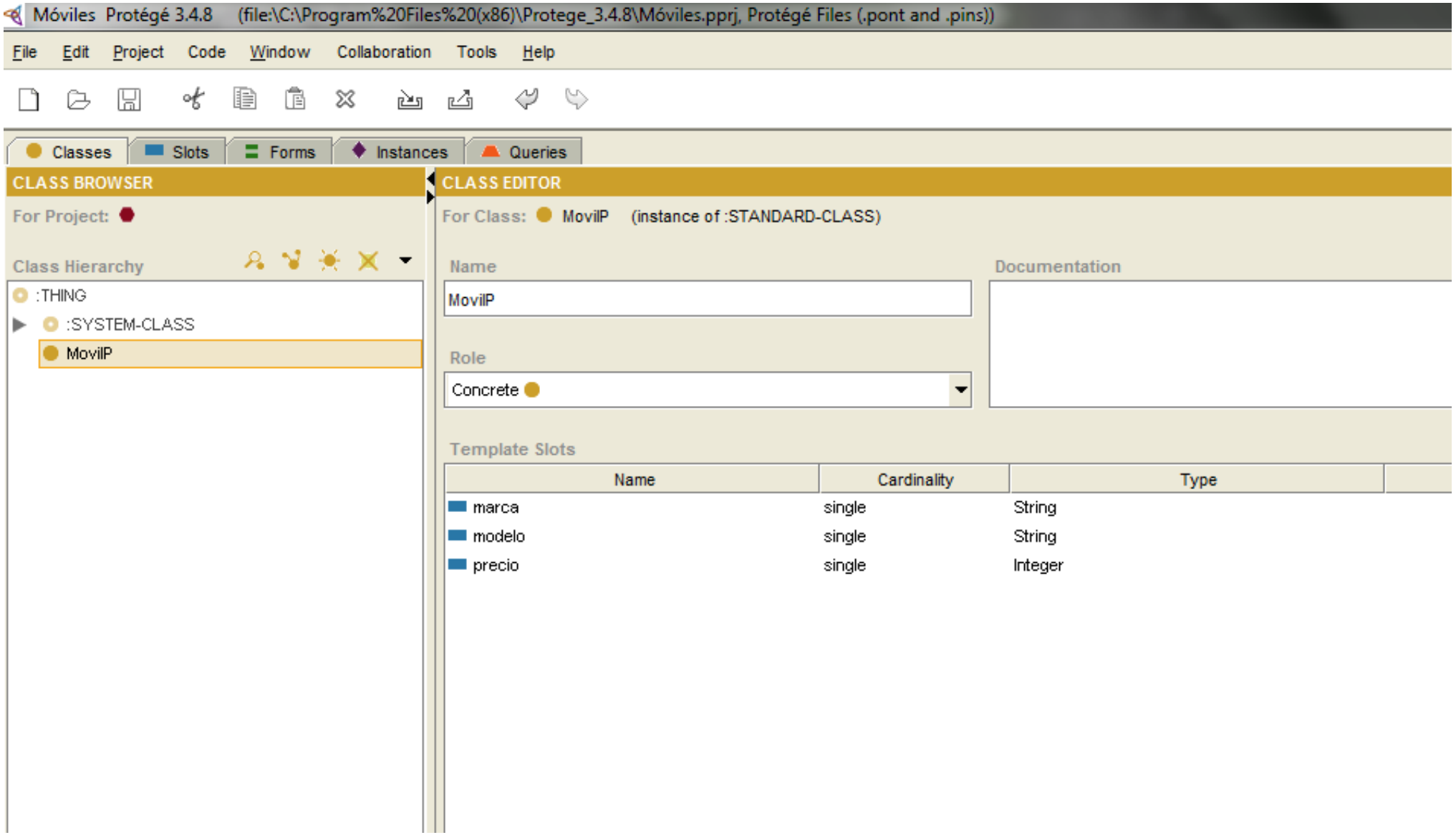
Configurar Protégé para incluir Jess



Consola de Jess en Protégé

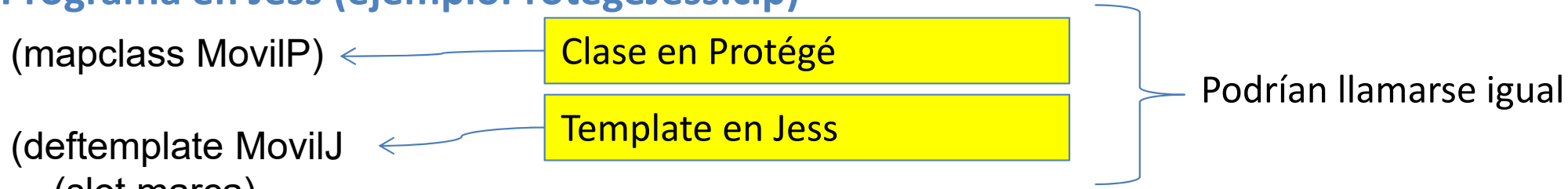


Ejemplo de carga de datos de móviles desde Jess



Ejemplo de carga de datos de móviles desde Jess

Programa en Jess (ejemploProtegeJess.clp)




```
    (slot marca)
    (slot modelo)
    (slot precio))

(deffacts inicio
    (MovilJ (marca Samsung)(modelo "Galaxy S4") (precio 699))
    (MovilJ (marca Apple) (modelo "iPhone 5") (precio 660)))

(defrule cargar
     (MovilJ (marca ?m)(modelo ?mo) (precio ?p))
=>
     (make-instance of MovilP (marca ?m)(modelo ?mo) (precio ?p)))

(reset)
(run)
(facts)
```

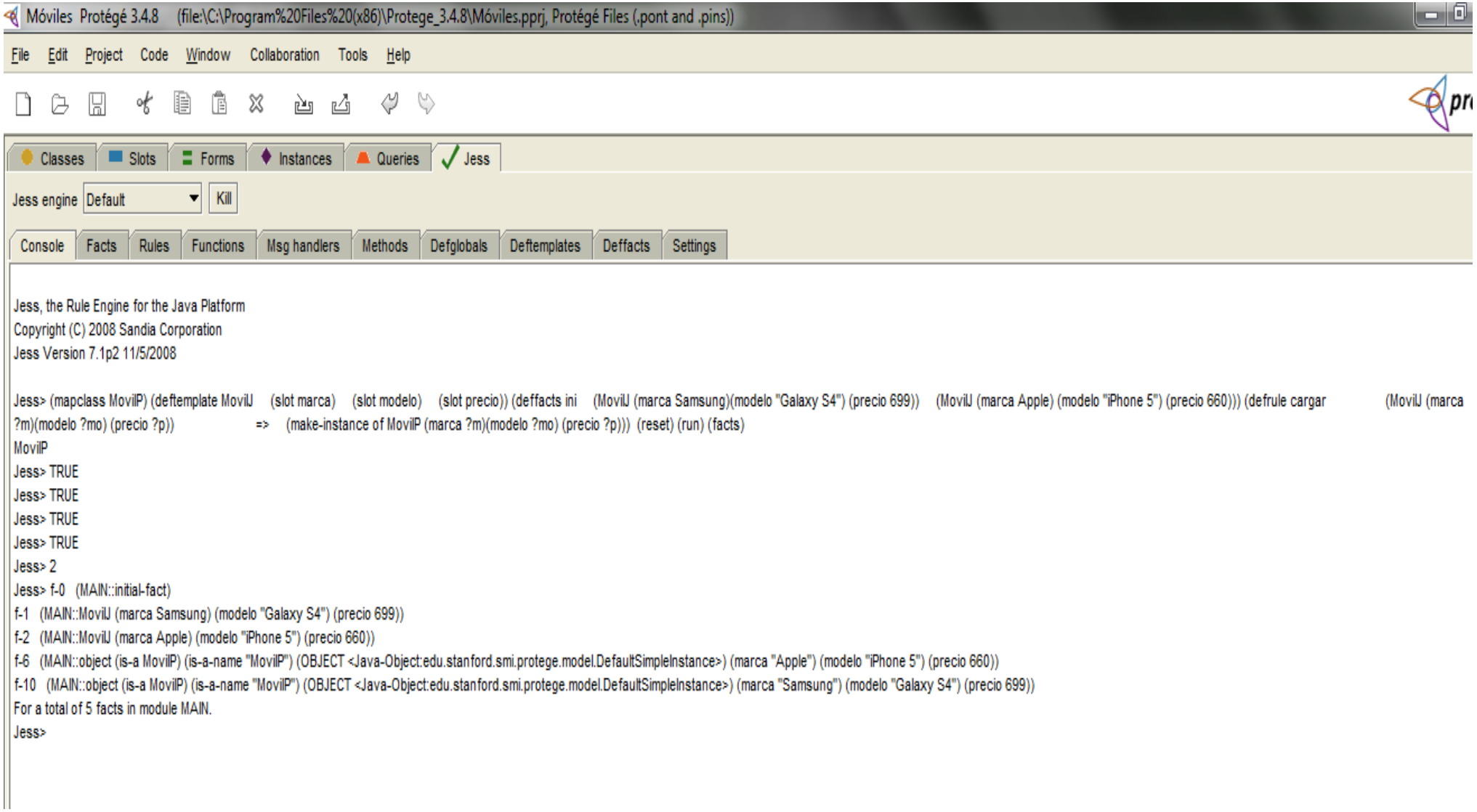
Ejecución de un programa Jess

El programa en Jess se puede ejecutar

- en *batch* con el nombre del fichero clp (rutas sin espacios ni acentos).
- pegando el contenido del clp en el panel inferior y pulsando *enter.*
- ejecutando instrucción a instrucción en el panel inferior.

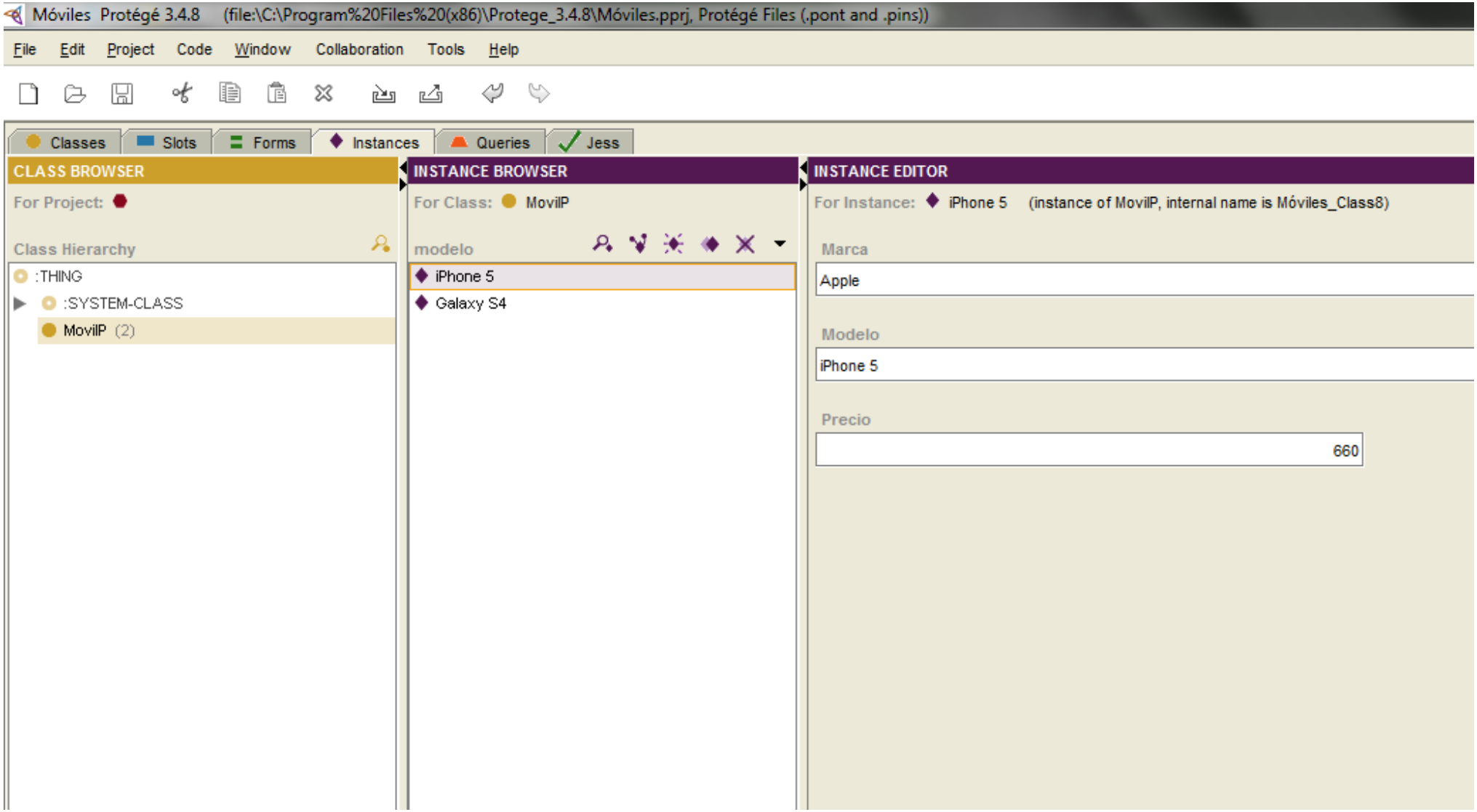
Ejemplo de carga de datos de móviles desde Jess

Programa Jess cargado y ejecutado



Ejemplo de carga de datos de móviles desde Jess

Se han añadido 2 instancias en Protégé

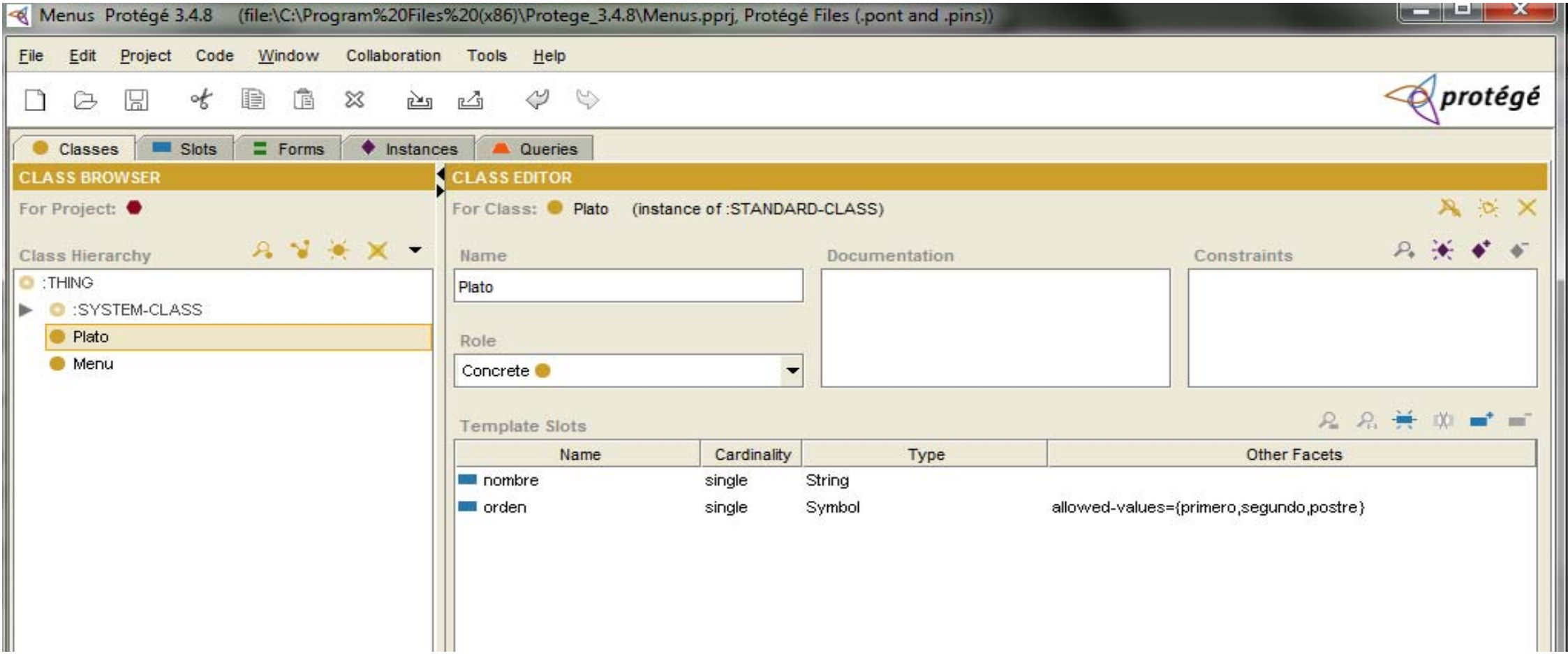


Ejemplo de ejecución de reglas con instancias de Protégé

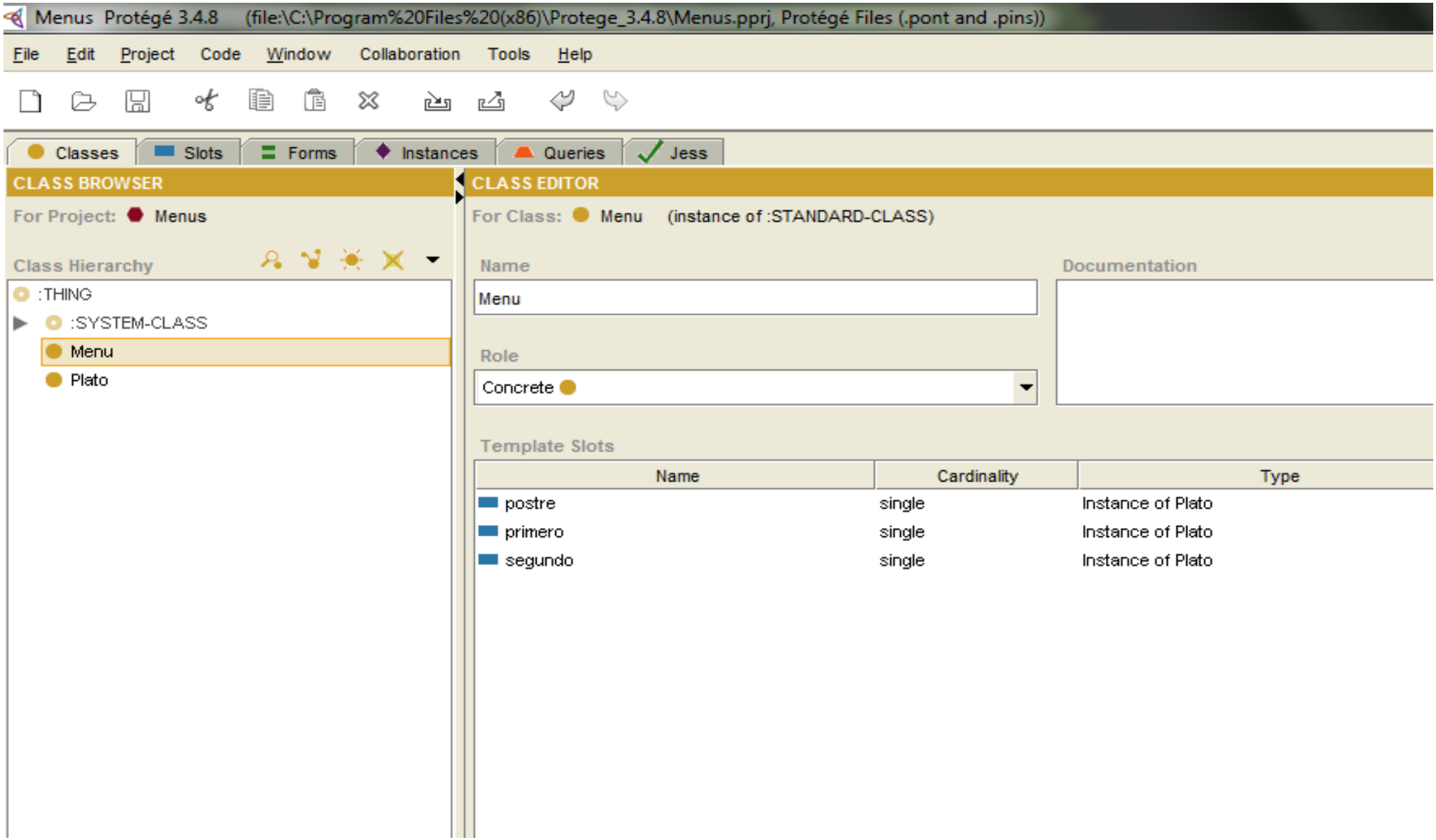
- A partir de una sencilla base de conocimiento de platos (creada en Protégé), con dos instancias de primer plato, dos de segundo plato y dos de postre, generar con Jess instancias de la clase Menú también definida en la misma ontología Protégé.

- En este ejemplo, todos los datos se han metido en Protégé y usamos una regla Jess para generar resultados en Protégé.

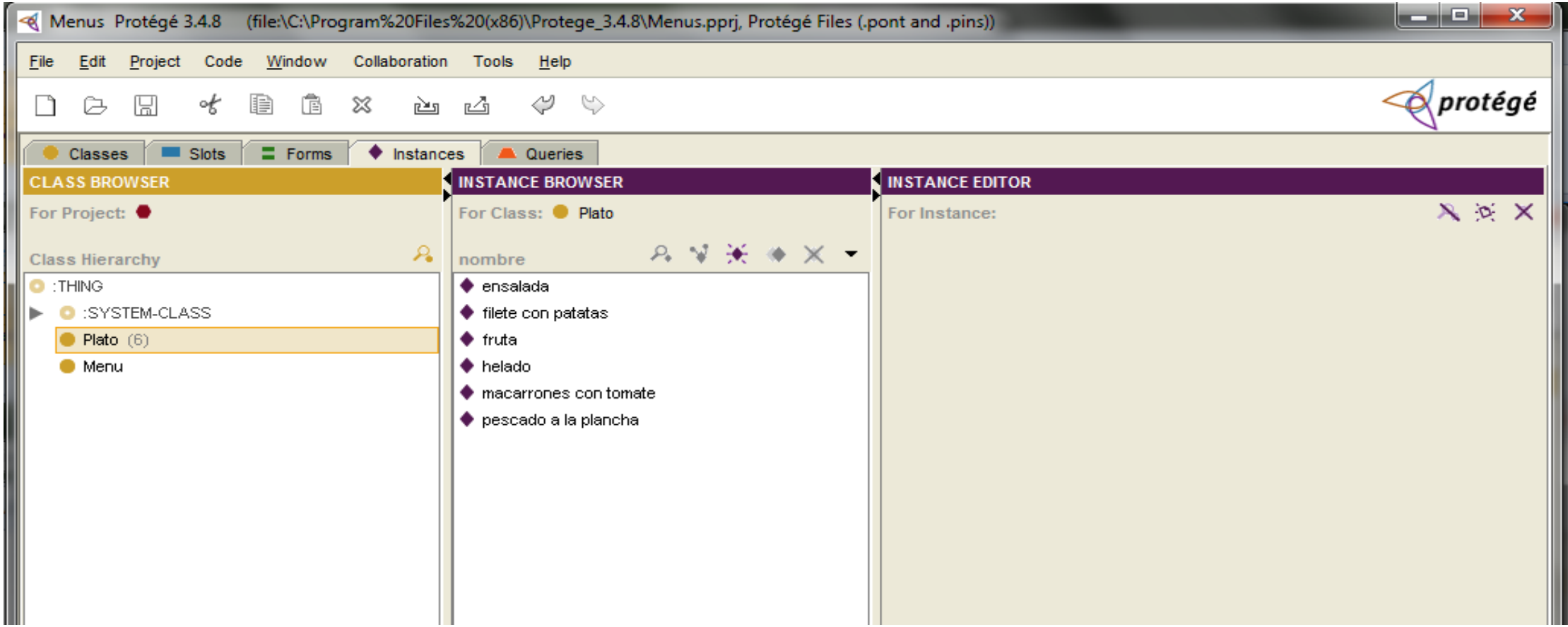
Clases en la ontología Protégé



Clase Menú

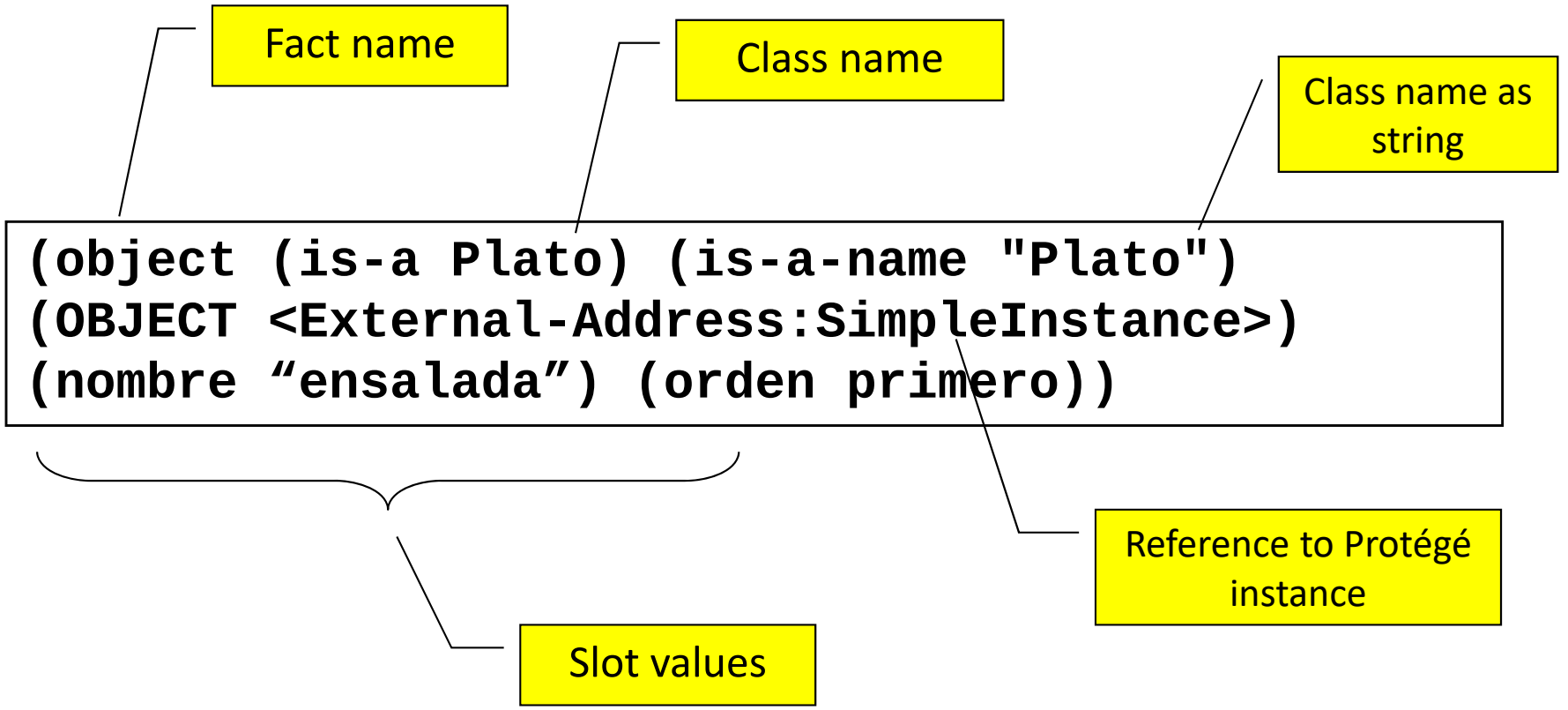

Menus Protégé 3.4.8 (file:\C:\Program%20Files%20(x86)\Protege_3.4.8\Menus.pprj, Protégé Files (.pont and .pins))
File Edit Project Code Window Collaboration Tools Help
Classes Slots Forms Instances Queries Jess
CLASS BROWSER
For Project: Menus
Class Hierarchy
:THING
:SYSTEM-CLASS
Menu
Plato
CLASS EDITOR
For Class: Menu (instance of :STANDARD-CLASS)
Name
Menu
Documentation
Role
Concrete
Template Slots
Name
Cardinality
Type
postre
single
Instance of Plato
primero
single
Instance of Plato
segundo
single
Instance of Plato

Instancias de Plato



Correspondencia entre Jess y Protégé

- Hecho Jess correspondiente a una instancia de la clase Plato creada en Protégé



Programa en Jess para el ejemplo de Menús

```
; programa en Jess con Protege
(mapclass Plato)
(mapclass Menu)
(defrule genera "Coge instancias de la clase Plato, genera instancias de Menú y
    las guarda en la ontología"
 ?h1 <- (object (is-a Plato)(orden primero))
 ?h2 <- (object (is-a Plato)(orden segundo))
 ?h3 <- (object (is-a Plato)(orden postre))
   =>
   (make-instance of Menu (primero ?h1) (segundo ?h2) (postre ?h3))
    )
(reset)
(run)
(facts)
```

Establece la correspondencia entre Protégé y Jess

Coge una instancia de cada plato en Protégé

Construye instancias de Menú en Protégé, con todas las combinaciones posibles de instancias de cada plato

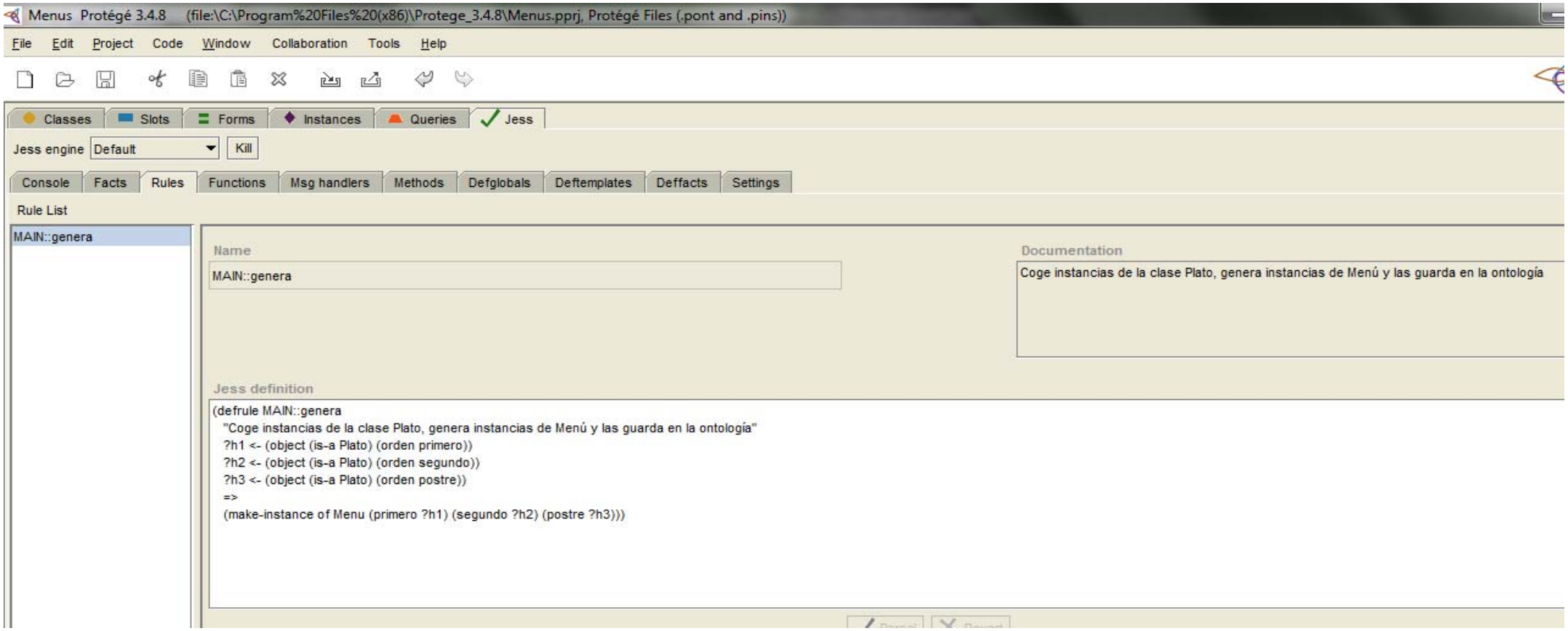
Programa en Jess para el ejemplo de Menús

Alternativamente, en vez de punteros a los hechos instancia, se pueden usar las referencias externas

```
; programa en Jess con Protege
(mapclass Plato)
(mapclass Menu)
(defrule genera "Coge instancias de la clase Plato, genera instancias de Menú y
    las guarda en la ontología"
 (object (is-a Plato) (OBJECT ?h1) (orden primero))
 (object (is-a Plato) (OBJECT ?h2) (orden segundo))
 (object (is-a Plato) (OBJECT ?h3) (orden postre))
   =>
   (make-instance of Menu (primero ?h1) (segundo ?h2) (postre ?h3))
    )
(reset)
(run)
(facts)
```

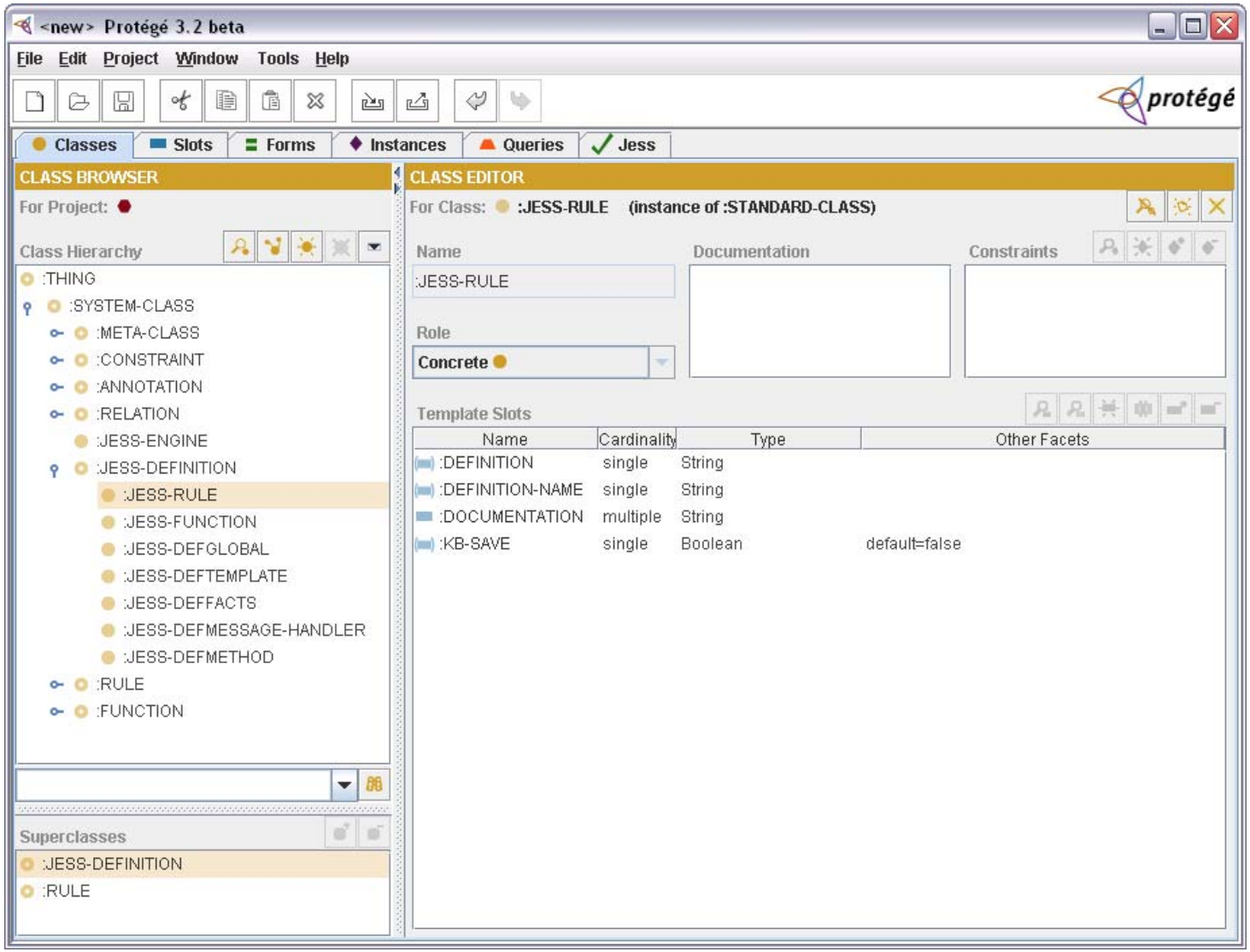
Regla incorporada a la ontología

Las reglas introducidas en el programa Jess se pueden visualizar desde la pestaña Rules y guardarlas junto con la ontología.

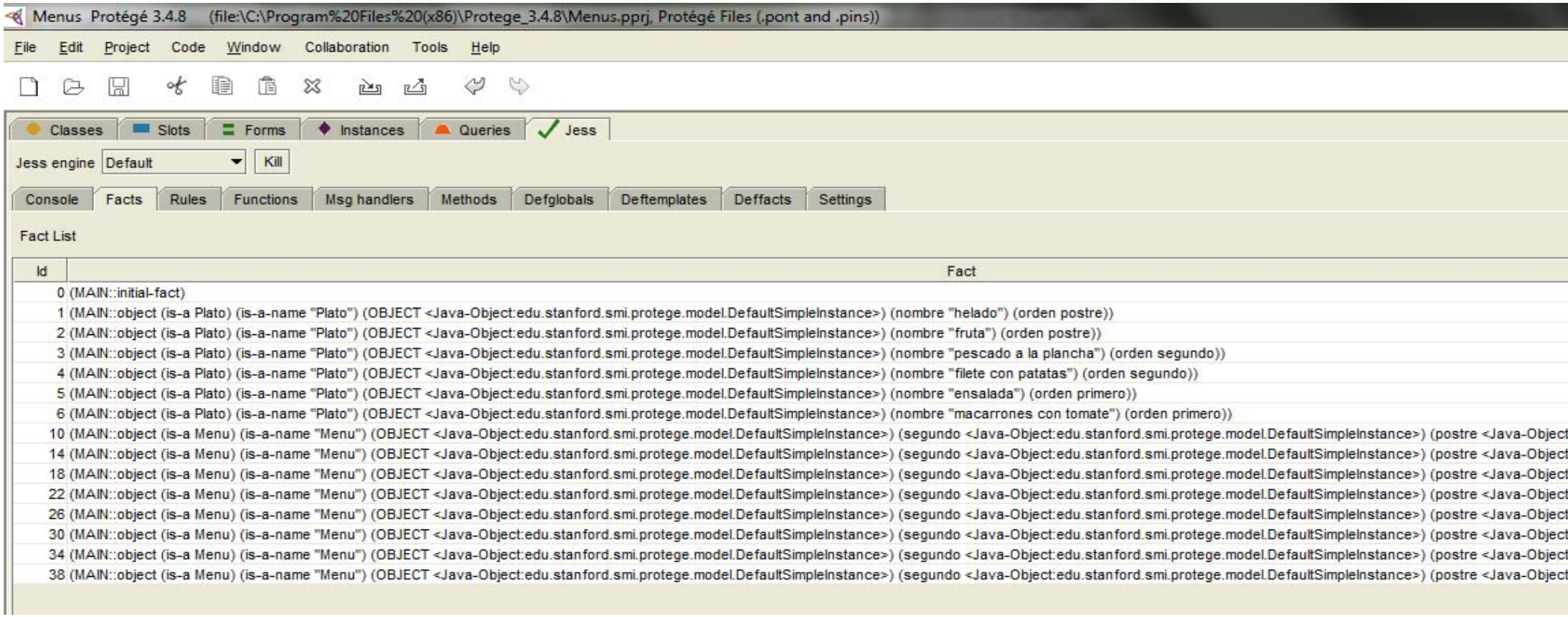


Regla incorporada a la ontología

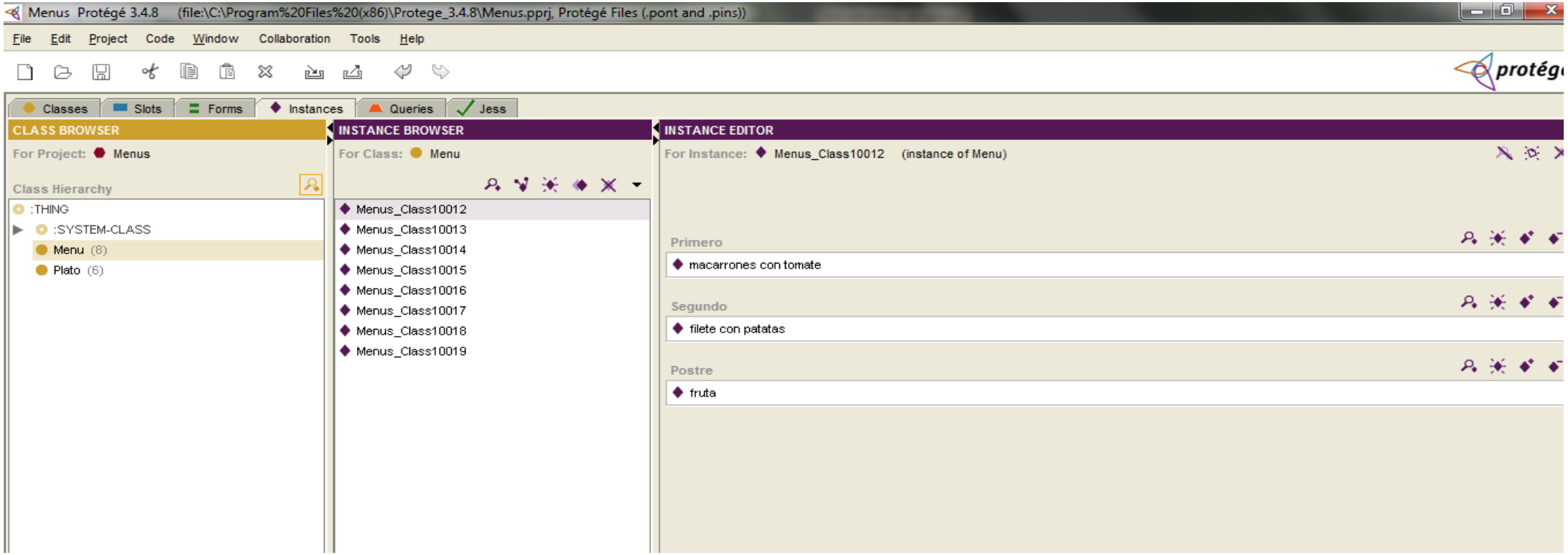
Las reglas introducidas en el programa Jess se pueden visualizar desde la pestaña Rules y guardarlas junto con la ontología.



Hechos Jess generados correspondientes a la ontología



Menús generados



Accediendo al contenido de las instancias desde las reglas

```
(defrule print_age_2
  (object (is-a Person)
          (OBJECT ?obj)
          (name ?n))
=>
  (printout t "The person " ?n
              " is " (slot-get ?obj age) crlf))
```

```
(defrule print_age_3
  ?f <- (object (is-a Person)
                (name ?n))
=>
  (printout t "The person " ?n
              " is " (slot-get ?f age) crlf))
```

Accediendo al contenido de las instancias desde las reglas

```
(defrule upgrade_my_age
  ?f <- (object (is-a Person)
                (name ”Joe Hacker”)
                (age 20))
=>
  (slot-set ?f age 21))
```

Herencia múltiple

- Un plato concreto (ensalada) es una instancia de Plato
- para hacer que también lo sea de la clase Primero (subclase de Plato), la añadimos a la lista de instancias directas de la clase Primero
- Hay que hacer los mapclass después de crear las instancias y antes de ejecutar la regla.

```
(mapclass :THING)
(mapclass Primero)

(defrule MAIN::CategorizarPrimerPlato
(object (is-a Plato) (OBJECT ?plato) (orden primero))
(object (is-a :STANDARD-CLASS) (:NAME "Primero")
                                        (:DIRECT-INSTANCES $?lista))
=>
(slot-set "Primero" :DIRECT-INSTANCES
                    (insert$ ?lista (+ 1 (length$ ?lista)) ?plato)))
```

Recomendaciones

- Recordar que JessTab no se lleva bien con la herencia múltiple
  - Las instancias para él pertenecen a la clase dentro de la que se crearon (aunque también sean hijas de otras clases)
  - Por ello, es conveniente usar una clase "almacén" que herede de la clase madre y de la que sean instancias todos nuestros objetos
    - Nuestras reglas trabajarán con instancias de esa clase y así no será necesario hacer reglas diferenciadas para cada subclase.
- Usar **`(mapclass NombreClase)`** para establecer correspondencias con las clases definidas en la ontología
  - Lo haremos con la clase madre ya que el "mapeo" se extiende a todas sus clases hijas
- Usar **`(facts)`** para ver los hechos Jess correspondientes a la ontología
- Empezar con ejemplos sencillos
- Desarrollar incrementalmente las reglas, con patrones que encajen en los hechos
- Ejecutar las reglas con (run) y ver los hechos resultantes

Funciones Jess para manejar bases de conocimiento

mapclass
mapinstance
unmapinstance
defclass
make-instance
initialize-instance
modify-instance
duplicate-instance
definstances
unmake-instance
slot-get
slot-set
slot-replace$
slot-insert$
slot-delete$
slot-facets
slot-types
slot-cardinality

slot-range
slot-allowed-values
slot-allowed-classes
slot-allowed-parents
slot-documentation
slot-sources
facet-get
facet-set
class
class-existp
class-abstractp
class-reactivep
superclassp
subclassp
class-superclasses
class-subclasses
get-defclass-list
class-slots

instancep
instance-existp
instance-name
instance-address
instance-addressp
instance-namep
slot-existp
slot-default-value
set-kb-save
get-kb-save
load-kb-definitions
load-project
include-project
save-project
jesstab-version-number
jesstab-version-string
get-knowledge-base
get-tabs

Referencias

- Manual de JessTab:
  http://www.ida.liu.se/~her/JessTab/JessTab.pdf
- Tutorial de JessTab:
  http://www.ida.liu.se/~her/JessTab/tutorial06/
- Ejemplo de programa en Jess que usa una ontología Protégé:
  http://www.ida.liu.se/~her/JessTab/tutorial06/newspaper.jess
- Transparencias del tutorial:
  http://www.ida.liu.se/~her/JessTab/tutorial06/JessTabTutorial.ppt