INSTITUTO FEDERAL
Rio Grande do Sul

AgentSpeak(L)

Linguagem de Programação Orientada a Agentes

TÓPICOS AVANÇADOS I - INTELIGÊNCIA ARTIFICIAL (SSI071)

PROFESSOR: FABIO OKUYAMA

ALUNO: YONATHAN STEIN

PORTO ALEGRE - 30/11/2016

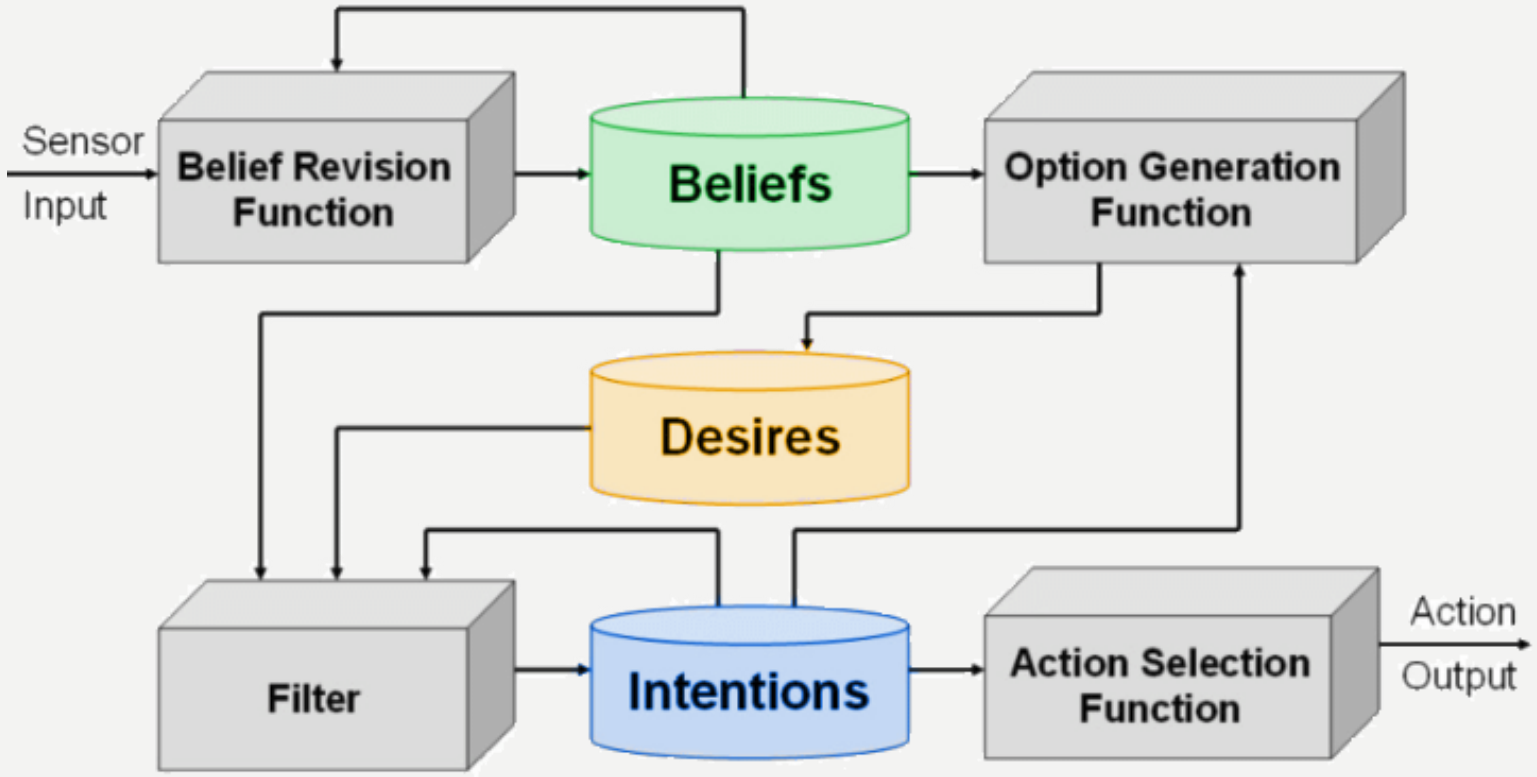
O QUE É AGENTSPEAK?

É uma linguagem de programação **orientada a agentes**. Baseia-se na programação lógica e na arquitetura **BDI** para os agentes autônomos (cognitivos). A linguagem foi originalmente chamado AgentSpeak (L) mas tornou-se mais popular como AgentSpeak.

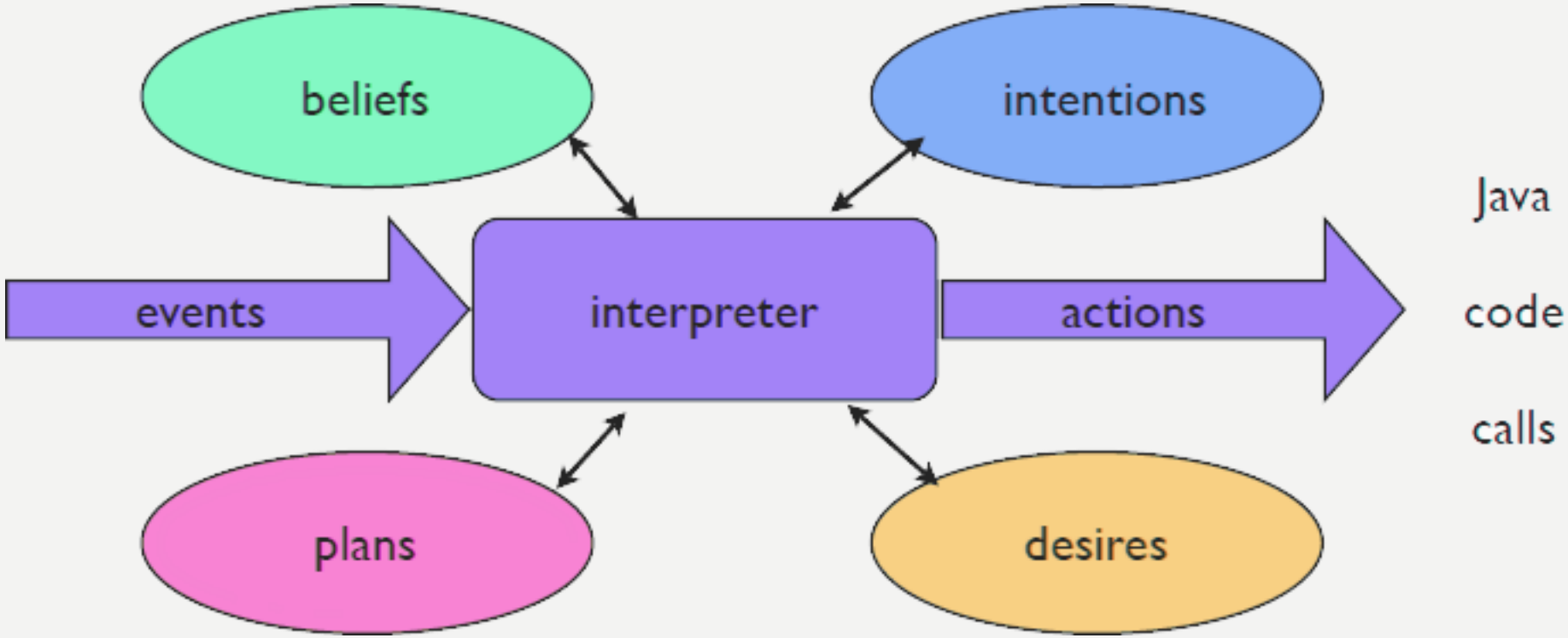
- Poderosa linguagem de programação para criar agentes inteligentes
- Baseado no paradigma: ***belief*** **–** ***desire*** **–** ***intention*** (**BDI**)
- Herança intelectual:
  - **The Procedural Reasoning System** (**PRS**)
    - *Developed at SRI in late 1980s*
  - *Logic Programming/Prolog*

ARQUITETURA (BDI)

CRENÇAS – DESEJOS – INTENÇÕES



ARQUITETURA (PRS)
SISTEMA DE RACIOCÍNIO PROCESSUAL



AGENTSPEAK: Control Loop

- Agente recebe **eventos** que podem ser:
  - Externos: a partir do ambiente ou a partir de dados perceptivos
  - Internamente gerados
- Tenta **tratar** os **eventos** procurando **planos** que **correspondam**
- Conjunto de **planos** que correspondem ao evento são **opções**/**desejos**
- Escolhe um **plano** dentre os **desejos** e executa: torna-se comprometido com isso – **uma intenção**
- Ao executar o **plano** pode gerar novos eventos que requeiram **tratamento**

AGENTSPEAK: Beliefs

Arquitetura:

- ***Beliefs*** (crenças) representam a informação que o agente possui sobre o seu ambiente
- São representadas **simbolicamente**
  - **Átomos de fundamento** lógico de primeira ordem

Exemplo:

- open(valve32)
- father(tom, michael)
- father(lily, michael)
- friend(michael, john)
- at_location(michael, gunne)
- on(blockA, blockB)

AGENTSPEAK: Plans

Arquitetura:

- Código desenvolvido off-line com antecedência
- Fornece informação ao agente sobre:
  - Como responder a **eventos**
  - Como atingir objetivos
- **Plans**:
  - ***event***
  - ***context***
  - ***body***

AGENTSPEAK: Plans

Estrutura:

- ***triggerCondition***
  - É um **evento** que o **plano** pode tratar
- **context**
  - Define as **condições** em que o **plano** pode ser utilizado
- **body**
  - Define as **ações** a serem realizadas se o **plano** for escolhido

triggerCondition :
context <-
body.

AGENTSPEAK: Events

Arquitetura:

- +! P
  - new goal acquired --“achieve P”
- -! P
  - goal P dropped
- + B
  - new belief B
- - B
  - belief B dropped

Jason is an interpreter for an extended version of AgentSpeak. It implements the operational semantics of that language, and provides a platform for the development of multi-agent systems, with many user-customisable features. Jason is available as Open Source, and is distributed under GNU LGPL.

Jason is developed by Jomi F. Hübner and Rafael H. Bordini, based on previous work done with many colleagues, in particular Michael Fisher, Joyce Martins, Álvaro Moreira, Renata Vieira, Willem Visser, Mike Wooldridge, but also many others, as acknowledged in the manual.

Jason
by Gustave Moreau (1865)

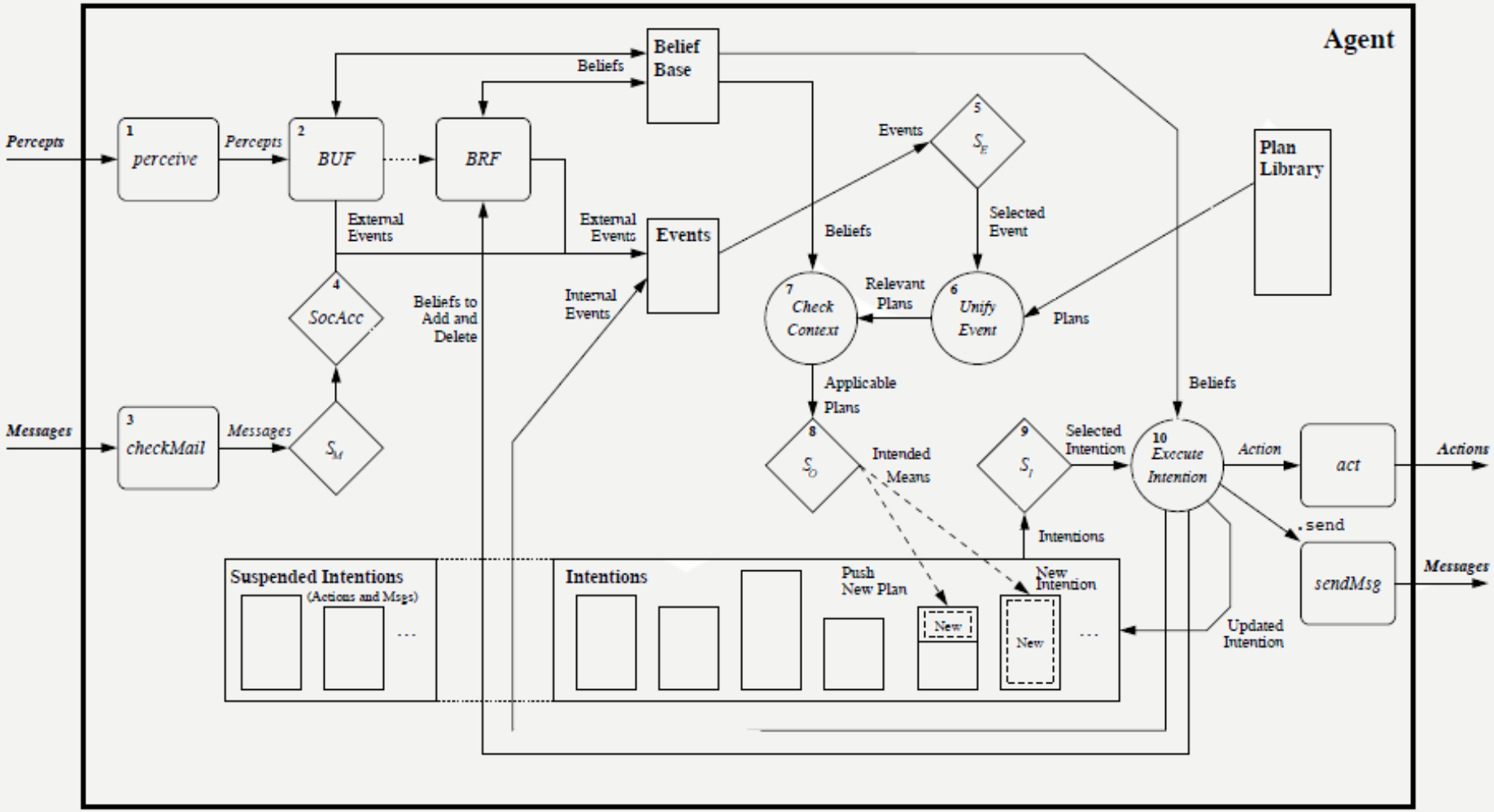
Oil on canvas, 204 x 115.5 cm
Musée d'Orsay, Paris.
© Photo RMN. Photograph by
Hervé Lewandowski

Jason

a Java-based interpreter for an extended version of AgentSpeak

- ✓ É uma implementação de AgentSpeak
- ✓ É uma biblioteca com ferramentas de depuração
- ✓ Possibilita o uso de multi-agentes
- ✓ Implementado em Java
- ✓ Cria um ambiente de desenvolvimento ágil
- ✓ Preparado para entrar em funcionamento rapidamente
- ✓ Software Livre (GNU LGPL)

JASON: Reasoning Cycle



JASON: Reasoning Cycle

Passos:

1. Perceber o Ambiente
2. Atualizando a Base de Conhecimento (Belief Base)
3. Recebendo Comunicação de Outros Agentes
4. Selecionando Mensagens "Aceitáveis Socialmente"
5. Selecionando um Evento
6. Recuperando Todos os Planos Relevantes
7. Determinação dos Planos Aplicáveis
8. Selecionando um Plano Aplicável
9. Selecionando uma Intenção para Execução Adicional
10. Executar um Passo de uma Intenção

http://jason.sourceforge.net/

https://github.com/jason-lang/jason

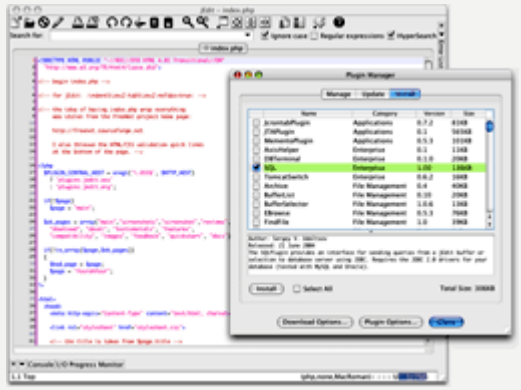
http://jason.sourceforge.net/Jason/Examples/Archive.html

Java SE Development Kit 8

http://www.oracle.com/technetwork/java/javase/downloads/

jEdit

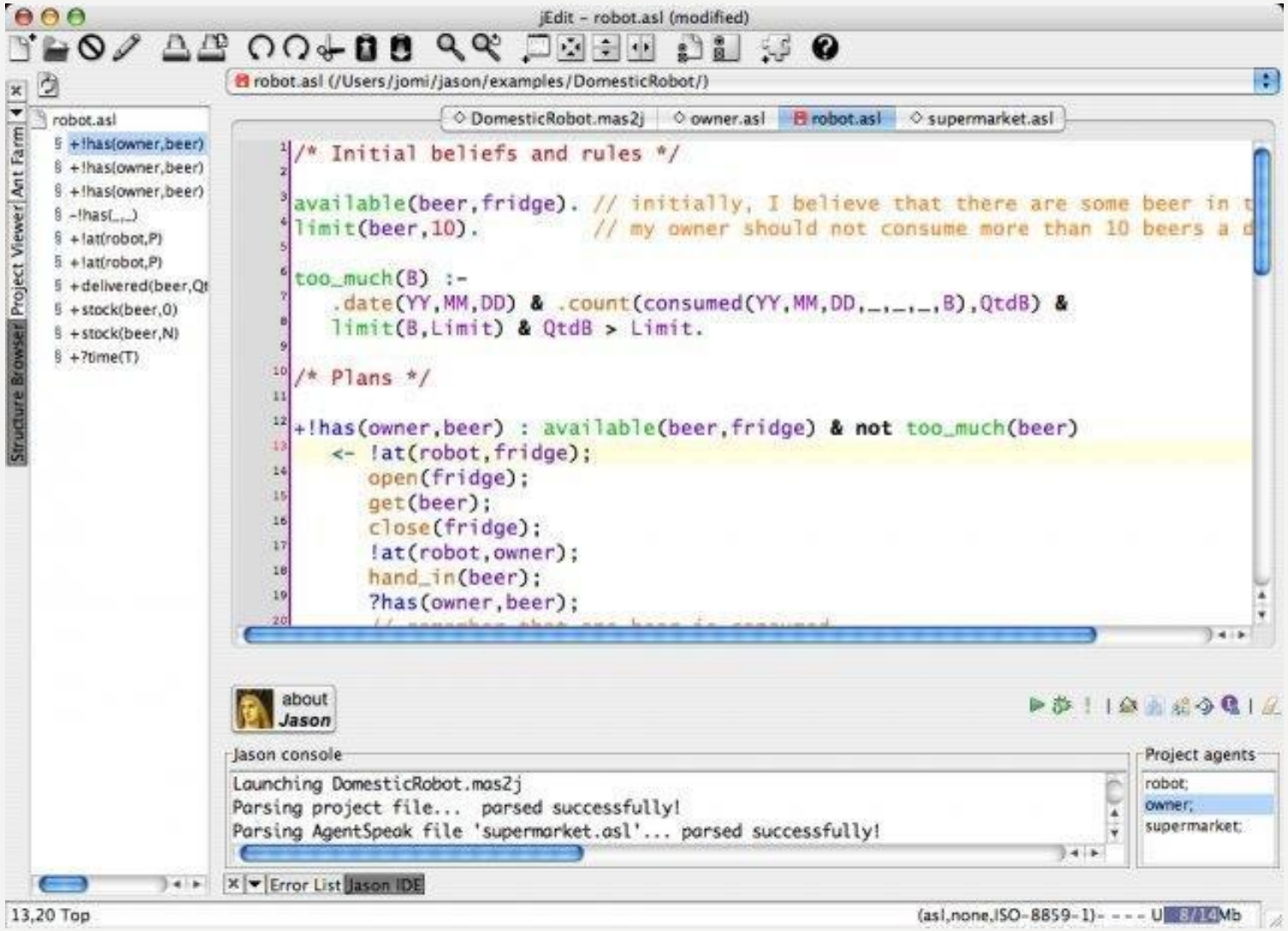
Programmer's Text Editor



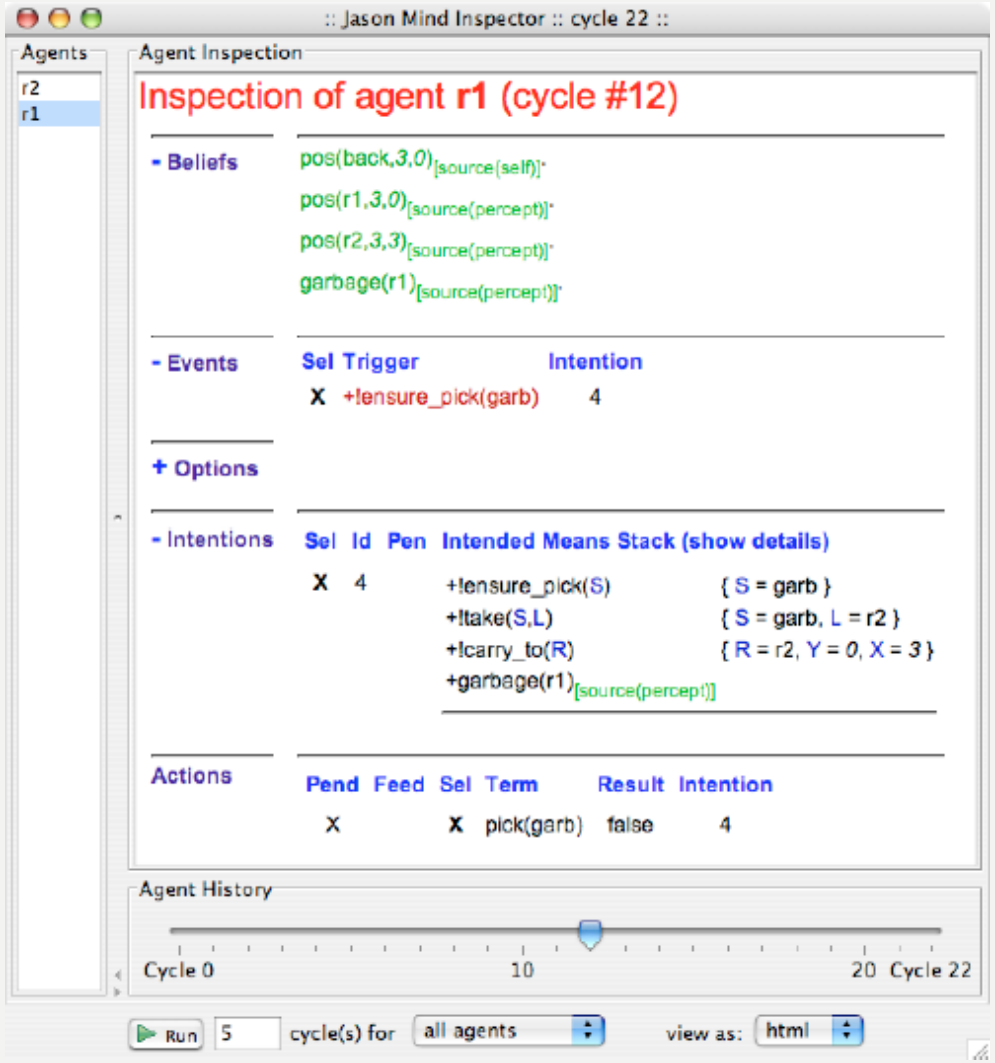
http://www.jedit.org/

https://sourceforge.net/projects/jedit/

JASON: jEdit

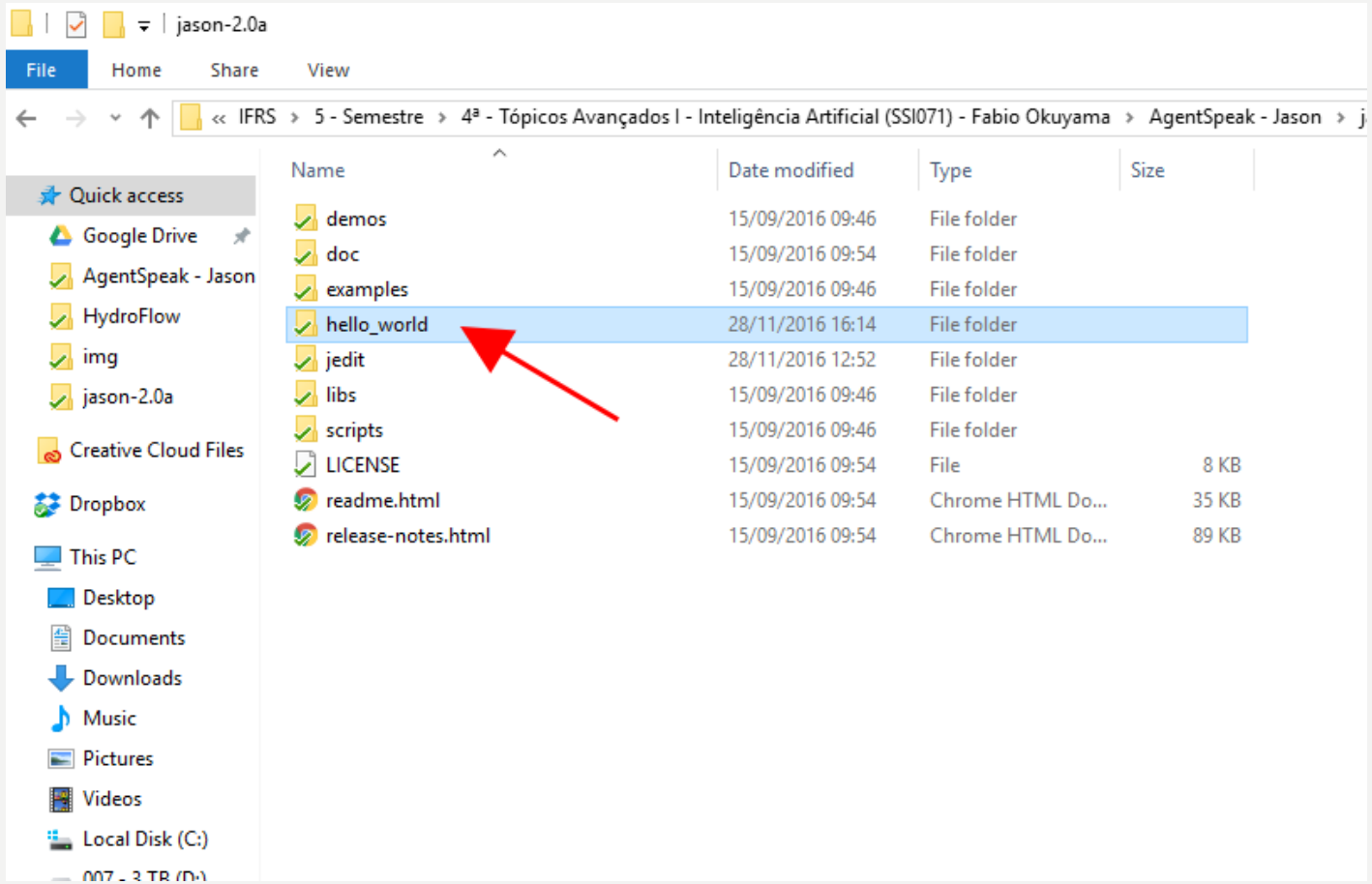

jEdit - robot.asl (modified)
robot.asl (/Users/jomi/jason/examples/DomesticRobot/)
DomesticRobot.mas2j
owner.asl
robot.asl
supermarket.asl
/* Initial beliefs and rules */
available(beer,fridge). // initially, I believe that there are some beer in t
limit(beer,10). // my owner should not consume more than 10 beers a d
too_much(B) :-
.date(YY,MM,DD) & .count(consumed(YY,MM,DD,_,_,_,B),QtdB) &
limit(B,Limit) & QtdB > Limit.
/* Plans */
+!has(owner,beer) : available(beer,fridge) & not too_much(beer)
<- !at(robot,fridge);
open(fridge);
get(beer);
close(fridge);
!at(robot,owner);
hand_in(beer);
?has(owner,beer);
about Jason
Jason console
Launching DomesticRobot.mas2j
Parsing project file... parsed successfully!
Parsing AgentSpeak file 'supermarket.asl'... parsed successfully!
Project agents
robot;
owner;
supermarket;
Error List
Jason IDE
13,20 Top
(asl,none,ISO-8859-1)

JASON: Mind Inspector



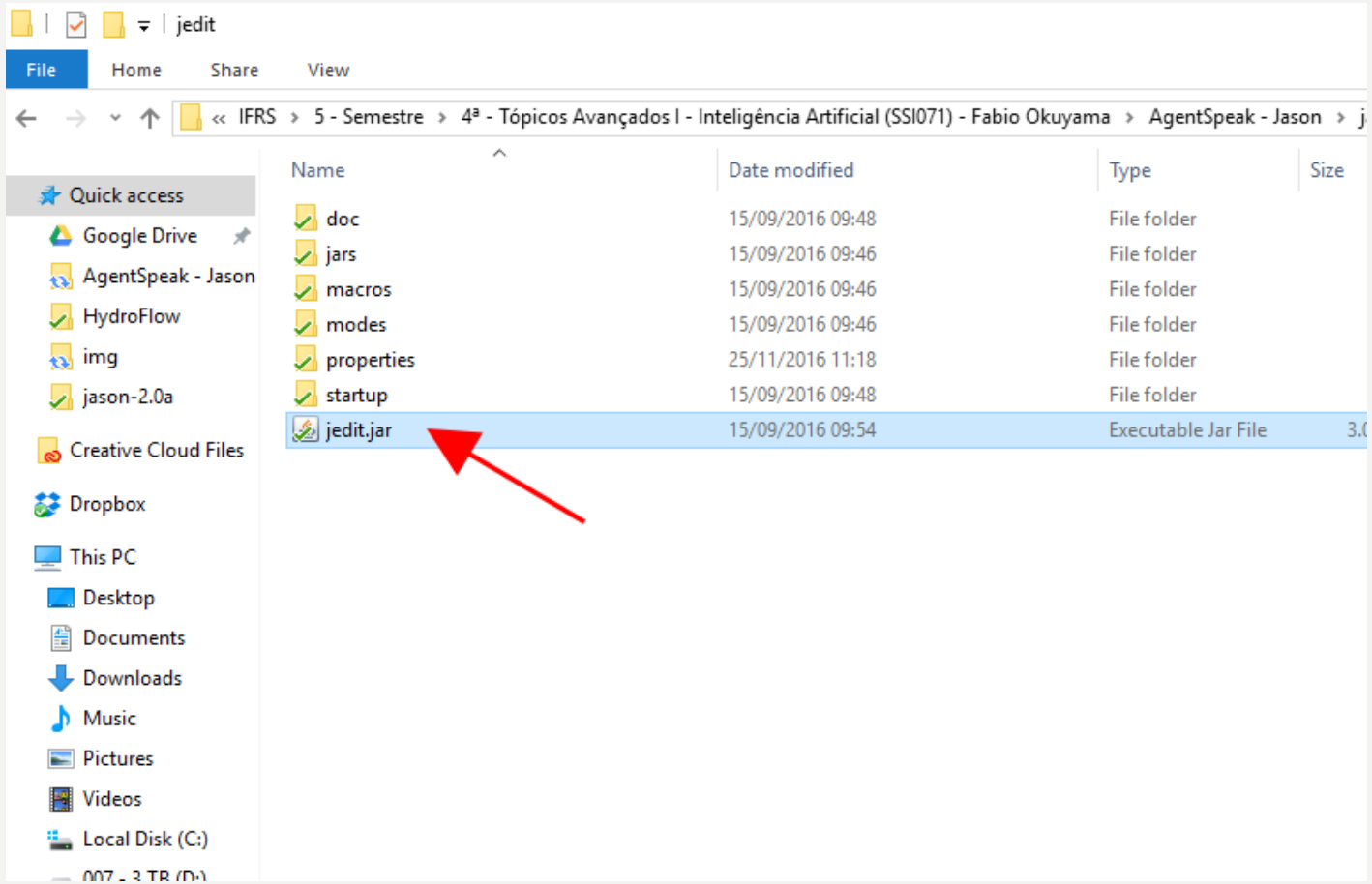
JASON: “Hello World”

Criar um diretório chamado “hello_world”



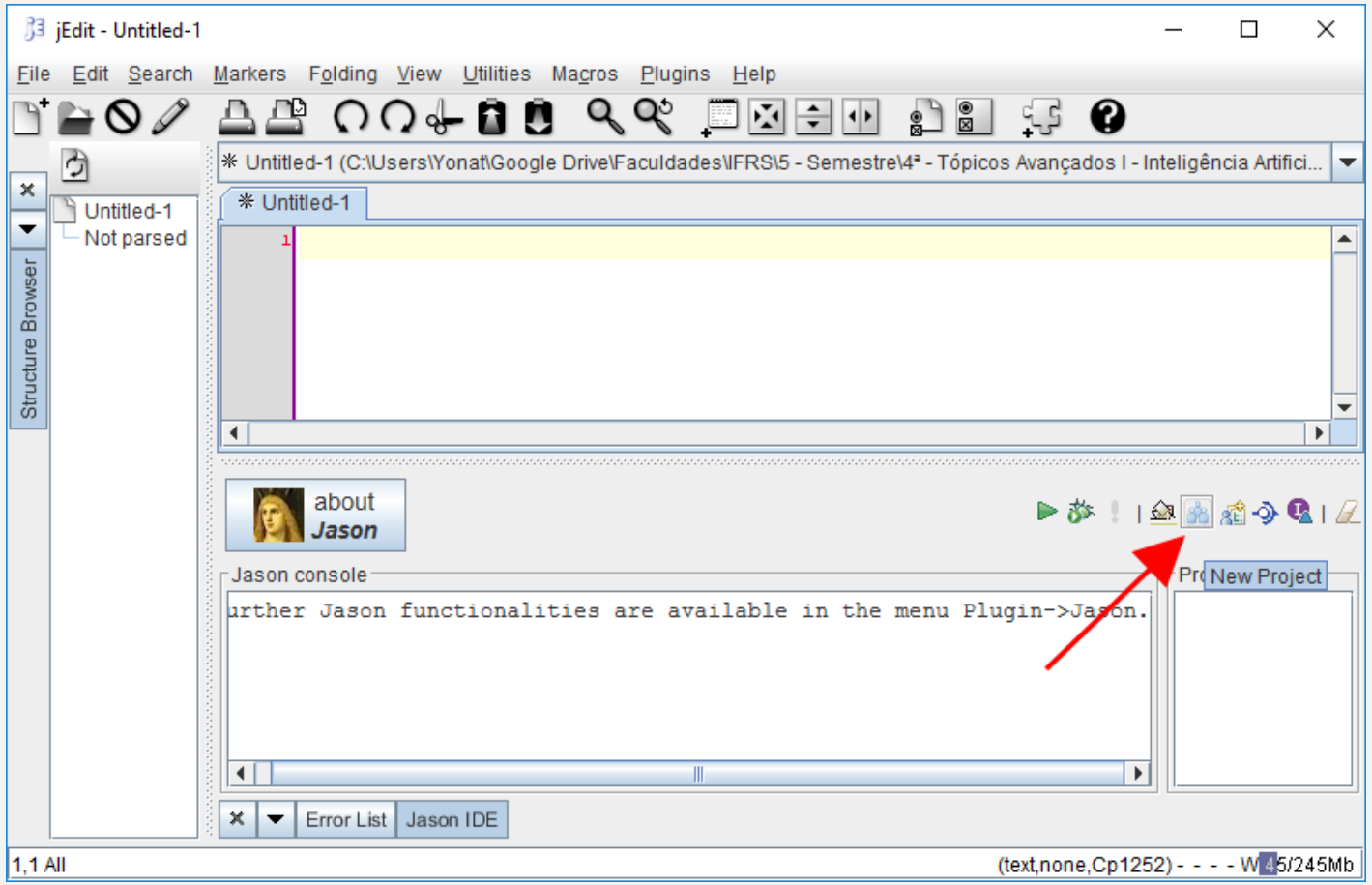
JASON: “Hello World”

Abrir o arquivo JAVA “jedit.jar” dentro do diretório “jedit”



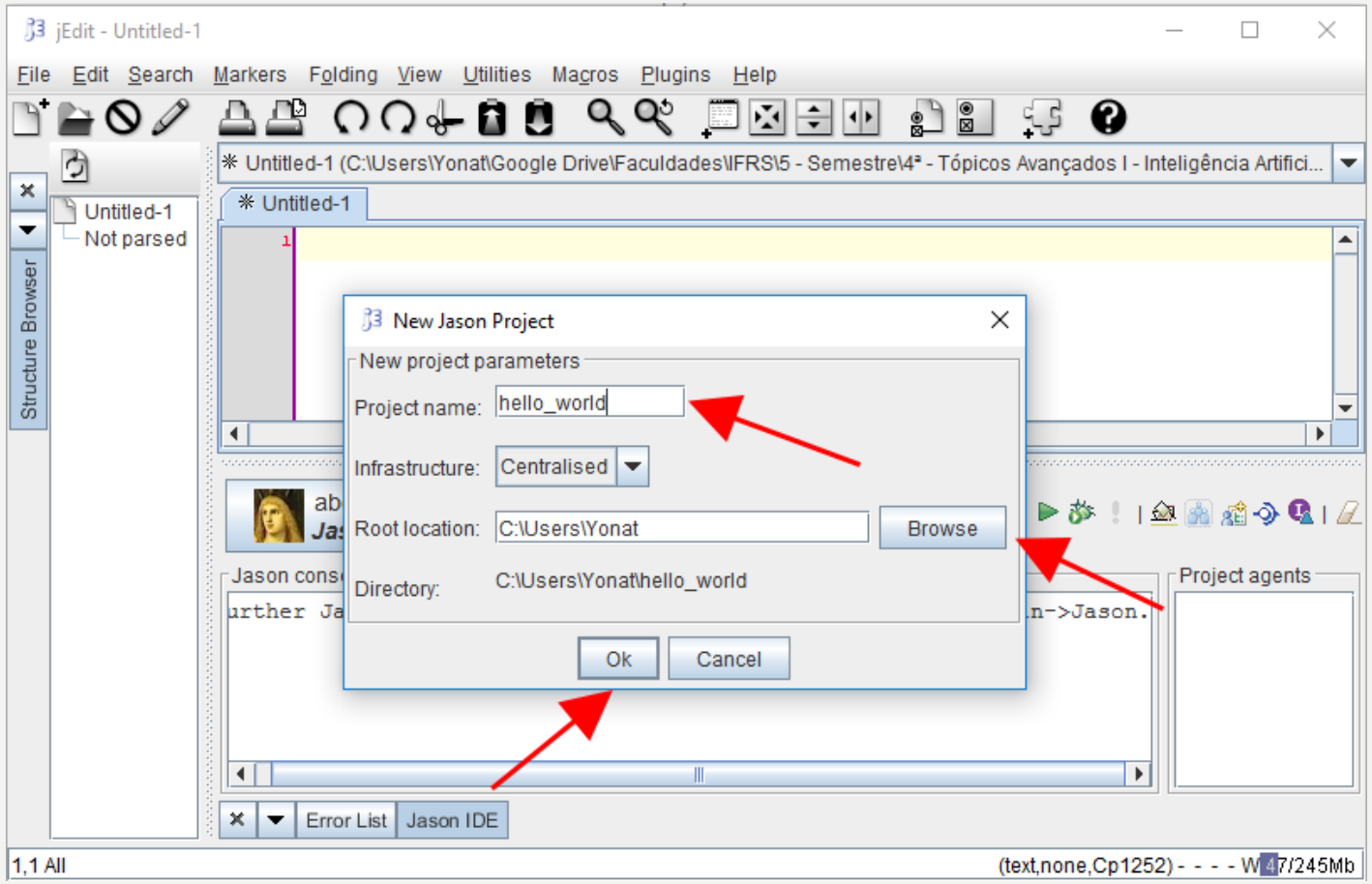
JASON: “Hello World”

Clicar em “New Project”



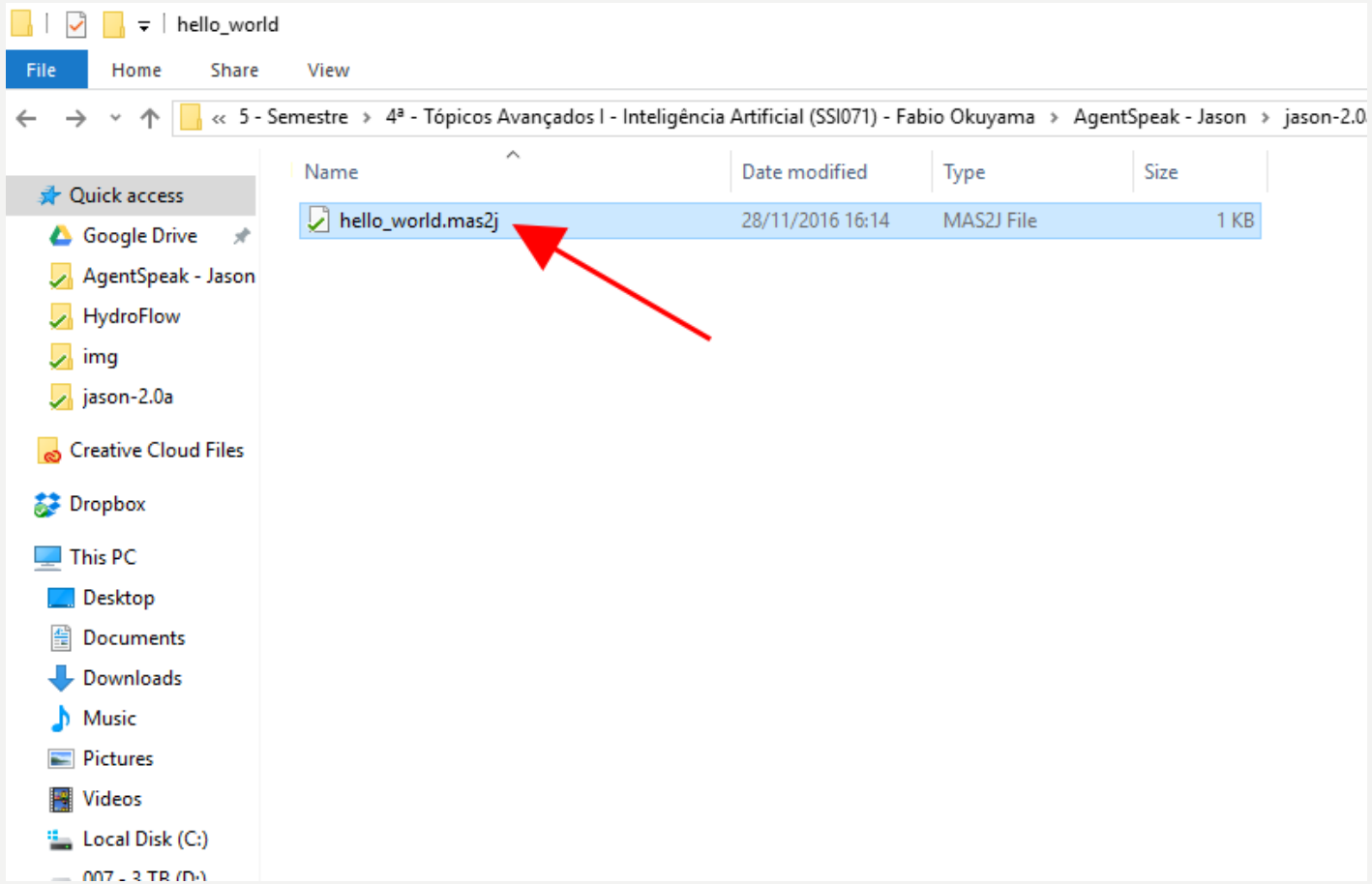
JASON: “Hello World”

Dar um nome ao projeto e selecionar o diretório raiz do diretório criado



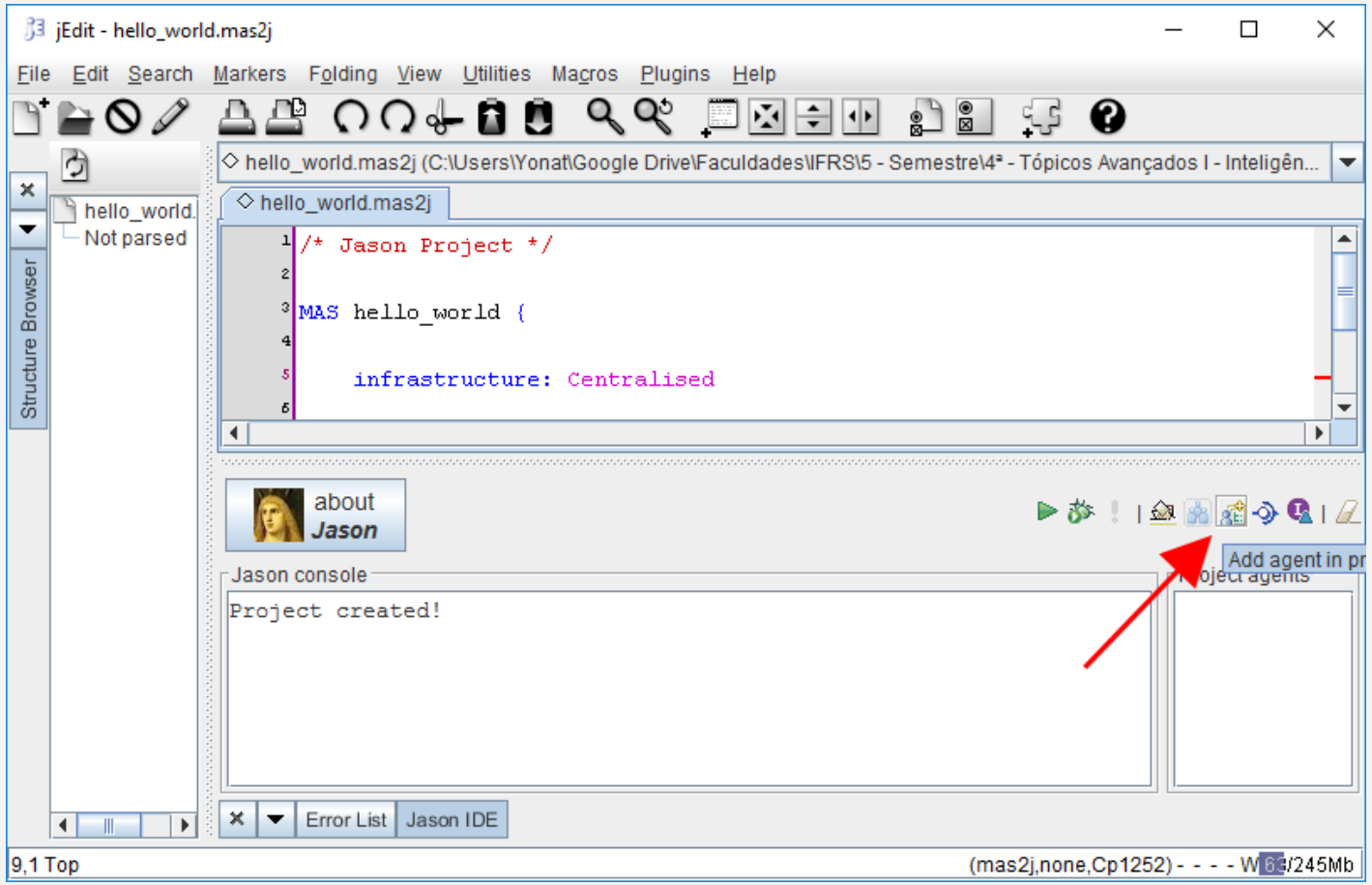
JASON: “Hello World”

Será criado um arquivo de mesmo nome no diretório “hello_world”



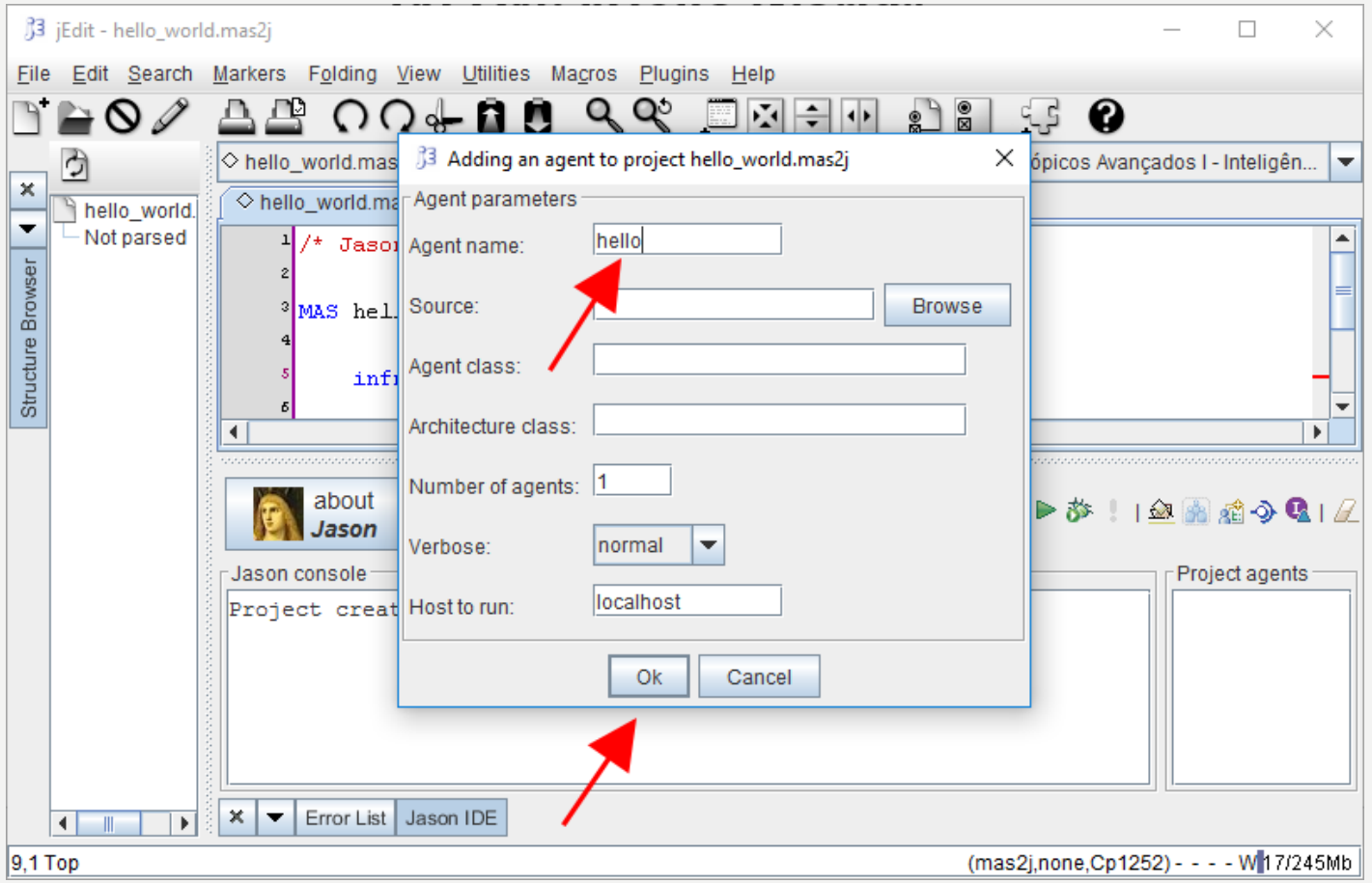
JASON: “Hello World”

Adicionar um agente clicando em “Add agent in project”



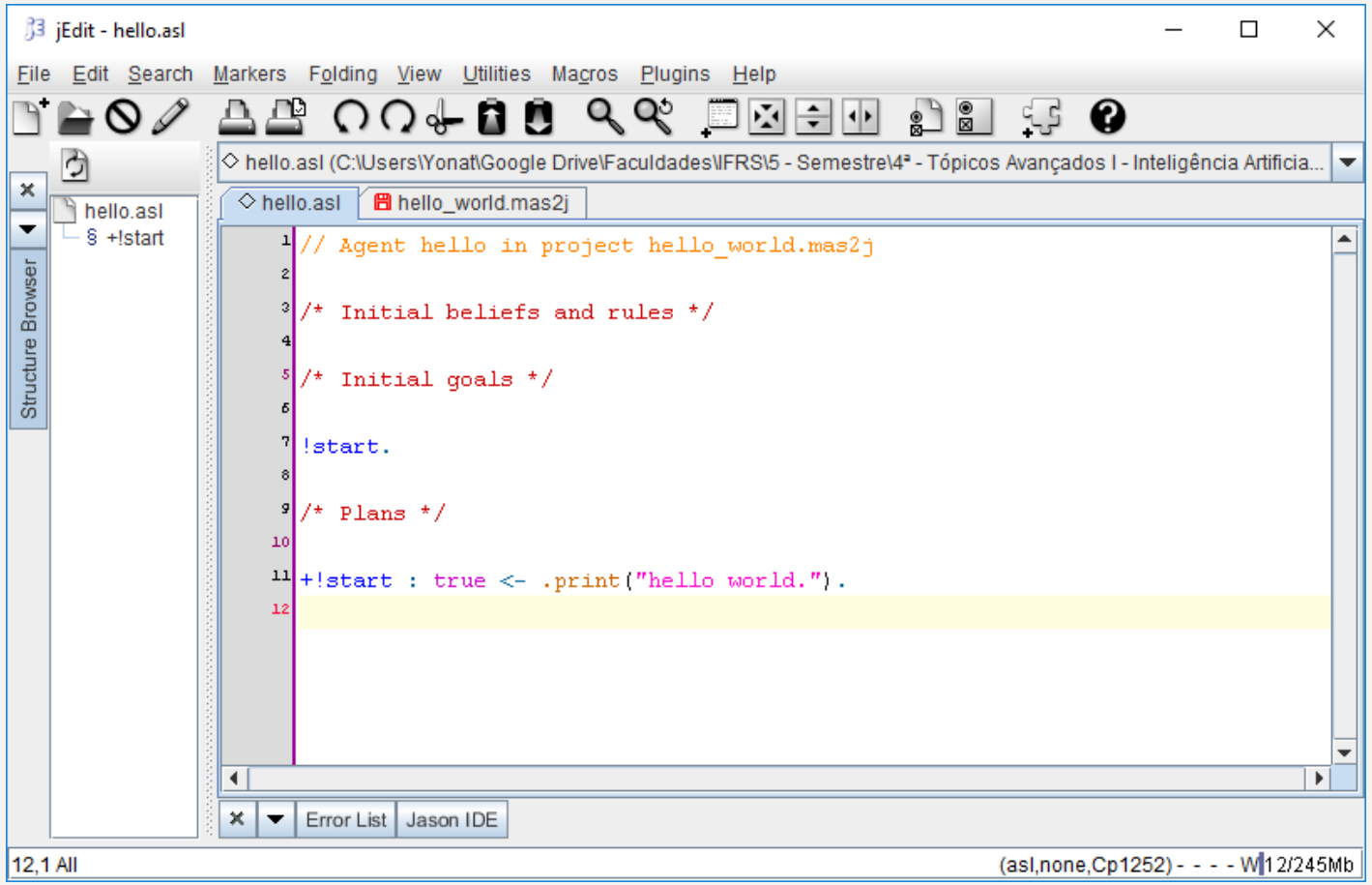
JASON: “Hello World”

Dar o nome “hello” e confirmar



JASON: “Hello World”

Será criado um agente com a definição padrão



JASON: “Hello World”

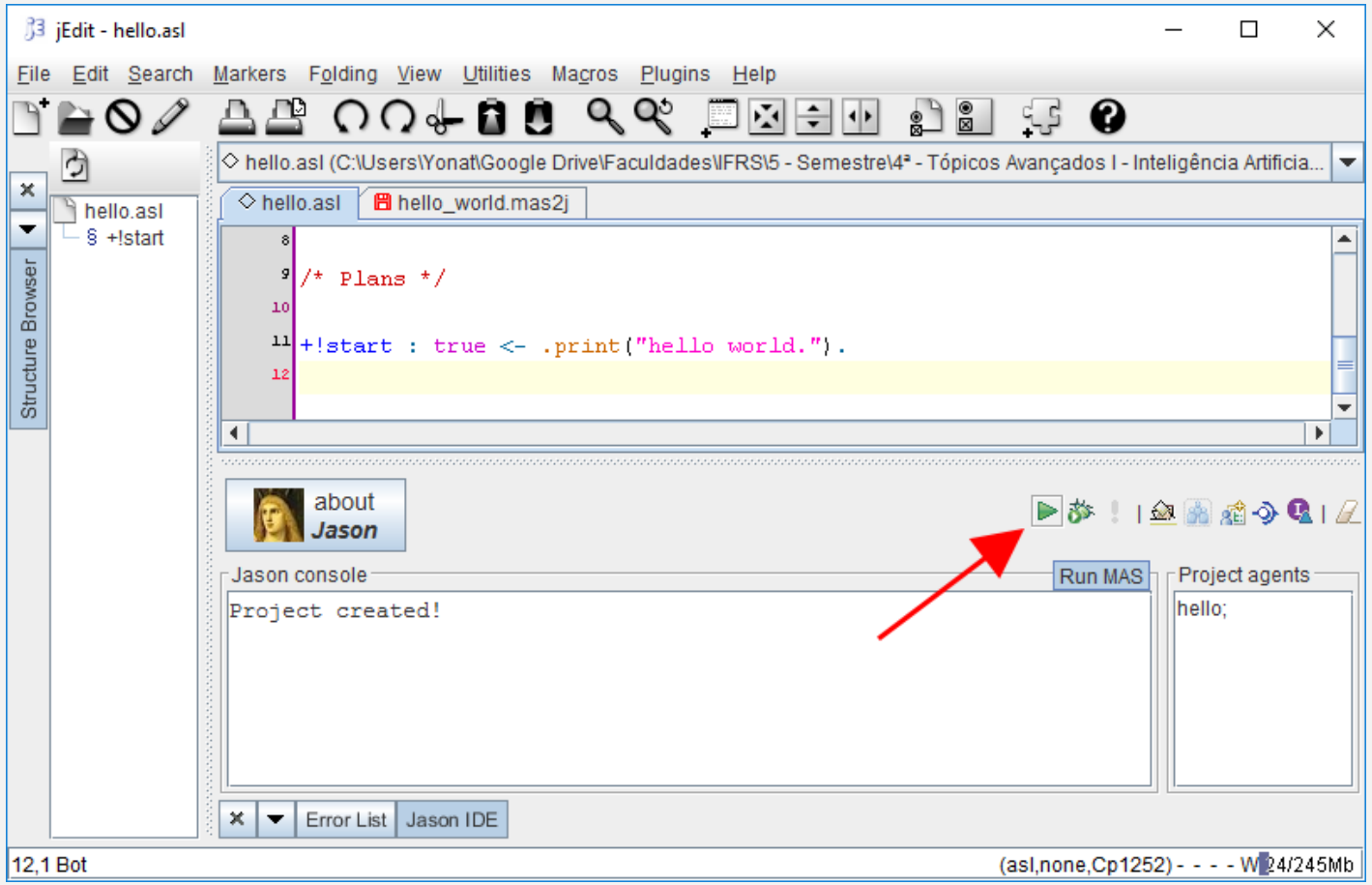
Explicação:

```
1.  // Agent hello in project hello_world.mas2j
2.
3.  /* Initial beliefs and rules */
4.
5.  /* Initial goals */
6.
7.  !start.
8.
9.  /* Plans */
10.
11. +!start : true <- .print("hello world.").
```

- O agente tem um único objetivo inicial: !start
- Este objetivo está lá quando o agente é iniciado
- O ponto de exclamação diz "este é um objetivo"
- Existe um único plano, que diz “se você tiver adquirido o objetivo start" deve: imprimir “hello world"

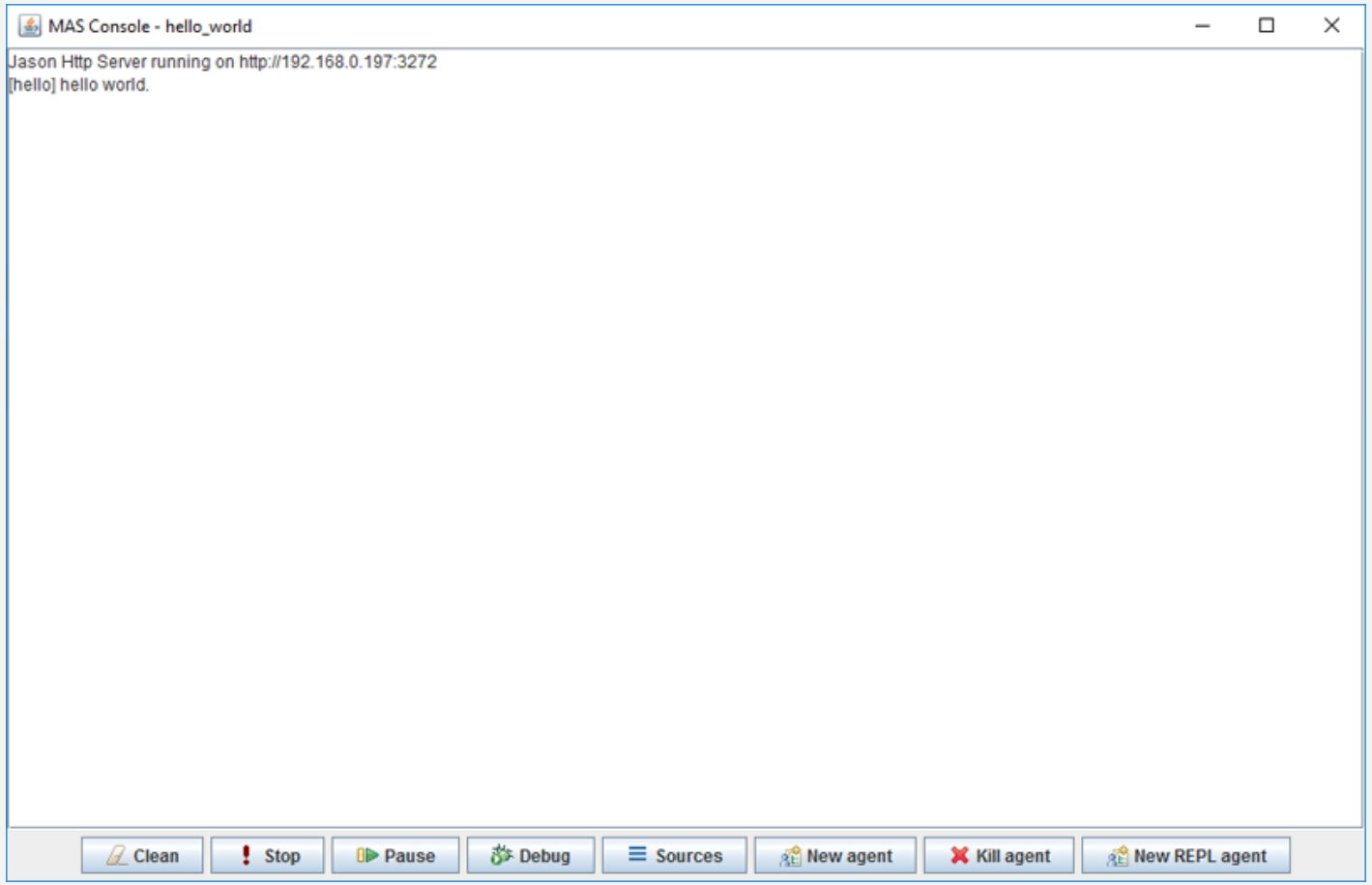
JASON: “Hello World”

Clicar em “Run MAS” para executar o programa

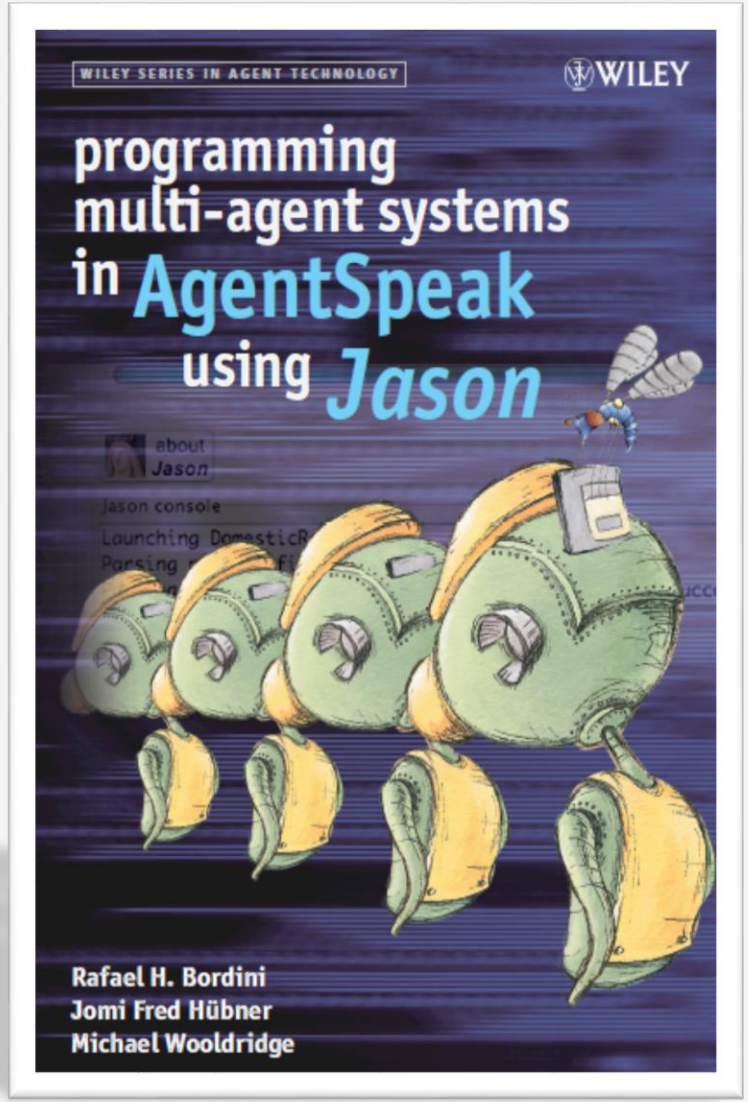


JASON: “Hello World”

Programa executado com sucesso!



JASON: Book


WILEY SERIES IN AGENT TECHNOLOGY
WILEY
programming
multi-agent systems
in AgentSpeak
using Jason
about Jason
Jason console
Rafael H. Bordini
Jomi Fred Hübner
Michael Wooldridge