

Instruções Laboratório LINUX

Curso de Introdução a Sistemas Multiagentes Embarcados

chon@grupo.cefet-rj.br

29 de setembro de 2025

Conteúdo

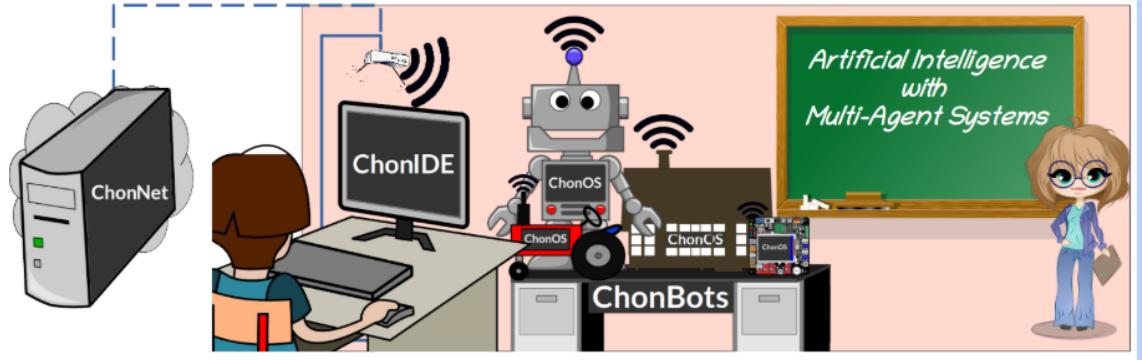
1	Introdução	2
2	Software	3
2.1	Instalação no Debian, Ubuntu ou derivados	3
2.1.1	Dependências	3
2.1.2	Instruções de Instalação da ChonIDE	3
2.1.3	Virtualizando no Windows	3
2.1.4	Teste de Funcionamento da ChonIDE	4
2.2	Testes dos simuladores	7
2.2.1	Webots	7
3	Rede	10
3.1	Wireless Local Area Network (WLAN)	10
3.2	Middleware de comunicação IoT	10
3.2.1	Liberação de portas no firewall para uso do Gateway público	10
3.2.2	Servidor ContextNet local	10
3.2.3	Teste de conectividade com o Middleware Context-net	11
3.3	Autodescoberta	12
3.3.1	Teste do serviço de autodescoberta	13
4	Bigchain DB	14
4.1	Liberação de portas no firewall para uso da TestChain pública	14
4.2	Servidor BigchainDB Local	14
4.3	Teste de Conectividade com o Servidor BigchainDB	14

1 Introdução

Este manual busca auxiliar a equipe de TI das instituições onde o curso de SMA Embarcado será ofertado. É necessário:

- instalar uma IDE nos computadores do laboratório;
- providenciar uma rede sem fio para que os alunos possam programar os protótipos de forma remota;
- permitir acesso ao Gateway IoT do grupo de pesquisa, ou prover um gateway local.

Este tutorial está dividido em 3 partes: software; rede e hardware. é necessário permitir acesso ao Gateway IoT do grupo de pesquisa, ou prover um gateway local.



2 Software

O desenvolvimento de SMA embarcados requer conhecimento em diferentes áreas, como programação orientada a agentes, programação orientada a objetos, programação de baixo nível e conceitos básicos de eletrônica. A literatura apresenta uma arquitetura de desenvolvimento dividida em quatro camadas: raciocínio, serial, firmware e hardware. No entanto, uma das principais dificuldades que os designers enfrentam é a necessidade de usar e configurar diferentes Ambientes de Desenvolvimento Integrado (IDE) e fazer várias integrações para embarcar o SMA. Mesmo utilizando todas essas tecnologias, a embarcação e monitoramento do é feita por meio de conexões físicas cabeadas, tornando-as limitadas e pouco práticas dependendo da aplicação.

Pensando nisso, o Chon Group (<https://github.com/chon-group>) desenvolveu a ChonIDE (<https://ide.chon.group/>) para facilitar a construção de SMA Embarcado, centralizando todo o processo de desenvolvimento, permitindo a embarcação e o monitoramento de forma fácil e remota, sem conexões físicas com fio.

2.1 Instalação no Debian, Ubuntu ou derivados

2.1.1 Dependências

Antes de começar, é necessário instalar as dependências. Primeiro, abra uma instância do terminal. Na distribuição Ubuntu e derivados, isso é normalmente feito utilizando a combinação de teclas CTRL + ALT + T. Faça uma atualização do computador, reiniciando-o logo em seguida:

```
sudo apt -y update;
sudo apt -y upgrade;
```

Ao reiniciar, abra uma instância do terminal e execute o seguinte comando:

```
sudo apt install git gcc g++ binutils make
```

2.1.2 Instruções de Instalação da ChonIDE

Inicialmente é necessário criar um novo arquivo de repositório do APT (/etc/apt/sources.list.d/chonos.list). Neste arquivo é necessário incluir uma linha que aponta para o repositório Main do ChonOS (deb [trusted=yes] http://packages.chon.group/ chonos main). Para isso, pode-se utilizar o comando abaixo:

```
echo "deb [trusted=yes] http://packages.chon.group/ chonos main" |
sudo tee /etc/apt/sources.list.d/chonos.list
```

Após a configuração do repositório é necessário atualizar a lista de pacotes (apt update) e instalar o Java e a ChonIDE (chonide) no computador. Para isso, utilize o comando abaixo:

```
sudo apt update ;
sudo apt install chonide chonos-simulide webots chonos-serial-port-emulator ;
```

2.1.3 Virtualizando no Windows

Caso o laboratório não possua computadores com Linux, é possível executar a ChonIDE através de uma máquina virtual. Para isso é necessário seguir os seguintes passos:

1. Realize o Download e Instalação do Oracle Virtual Box, disponível em: <https://www.virtualbox.org/wiki/Downloads>
2. Realize o Download da versão mais atualizada da VM disponível em: <https://sourceforge.net/projects/chonide/files/> e proceda com a importação para o Virtual Box.
3. Na primeira execução da VM, execute a atualização. Na tela inicial você encontra um ícone update . Clique no ícone e escolha a opção executar em um terminal.

As credenciais de acesso padrão da VM são **usuário=chon**; **senha=chon**.

2.1.4 Teste de Funcionamento da ChonIDE

A ChonIDE conta com alguns projetos de demonstração que facilitam o teste de funcionamento. Para tal, execute a ChonIDE. No Ubuntu, clique em mostrar aplicativos (1), na barra de pesquisa (2), digite chonide, por fim clique no ícone (3) para executar conforme a Figura 1. Caso esteja utilizando a ChonIDE no VirtualBox, o ícone já se encontra na Área de Trabalho.

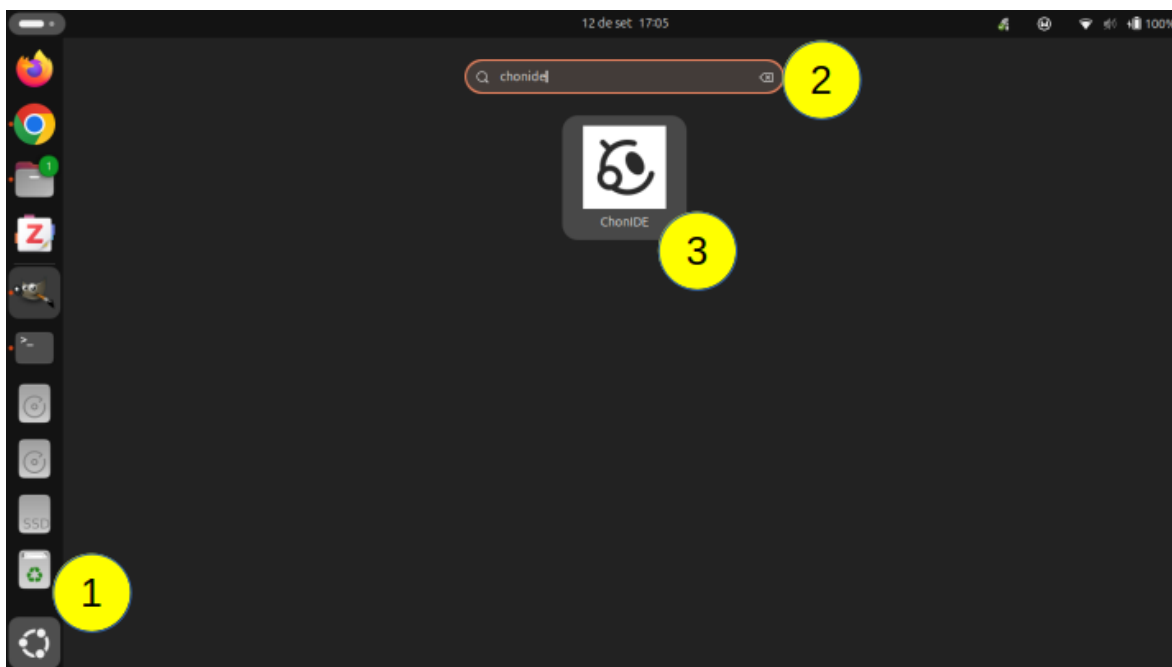


Figura 1: Iniciando a ChonIDE

Uma vez que a ChonIDE é executada como uma página web, para utilizá-la, é necessário realizar a autenticação na página inicial. Informe um nome de usuário válido no computador (1) e a sua respectiva senha (2). Após isso, clique em entrar (3), conforme a Figura 2.

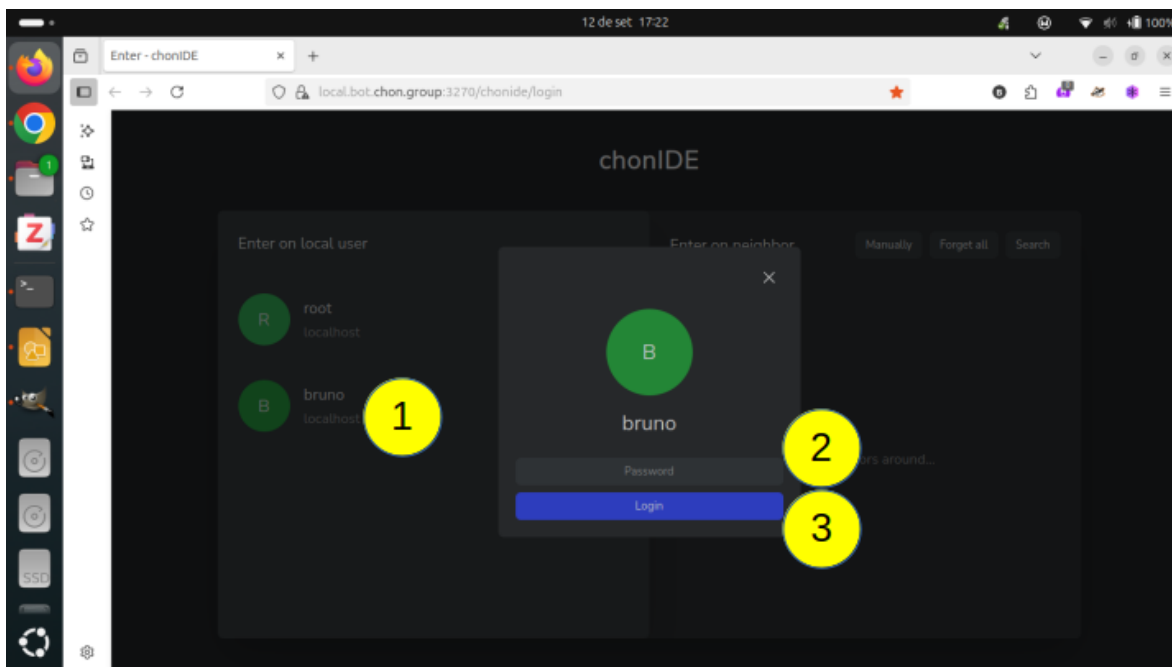


Figura 2: Autenticação na ChonIDE

Na tela principal, abra o projeto modelo *helloAgent*, conforme Figura 3.

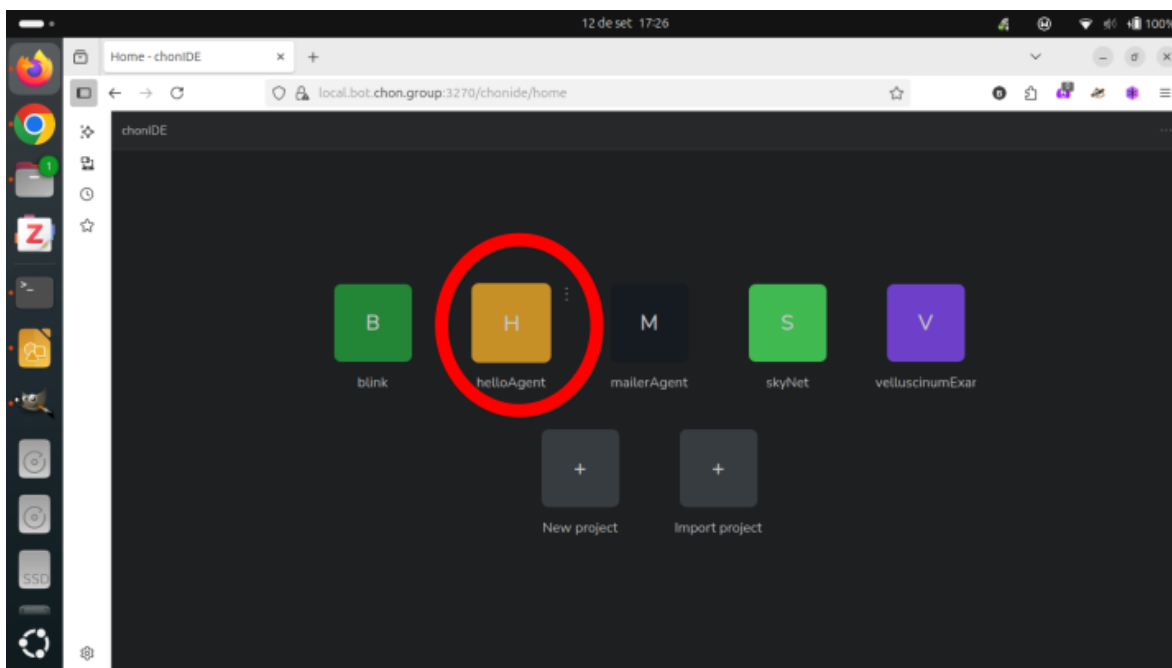


Figura 3: Abrindo o projeto *helloAgent*

Uma vez aberto o projeto helloAgent, clique em *Run MAS* (1) para executar o exemplo, conforme a Figura 4. Caso a mensagem “*Never Send A Human To Do A Machine’s Job!*” no Log seja exibida (2), sua instalação está correta. Parabéns!

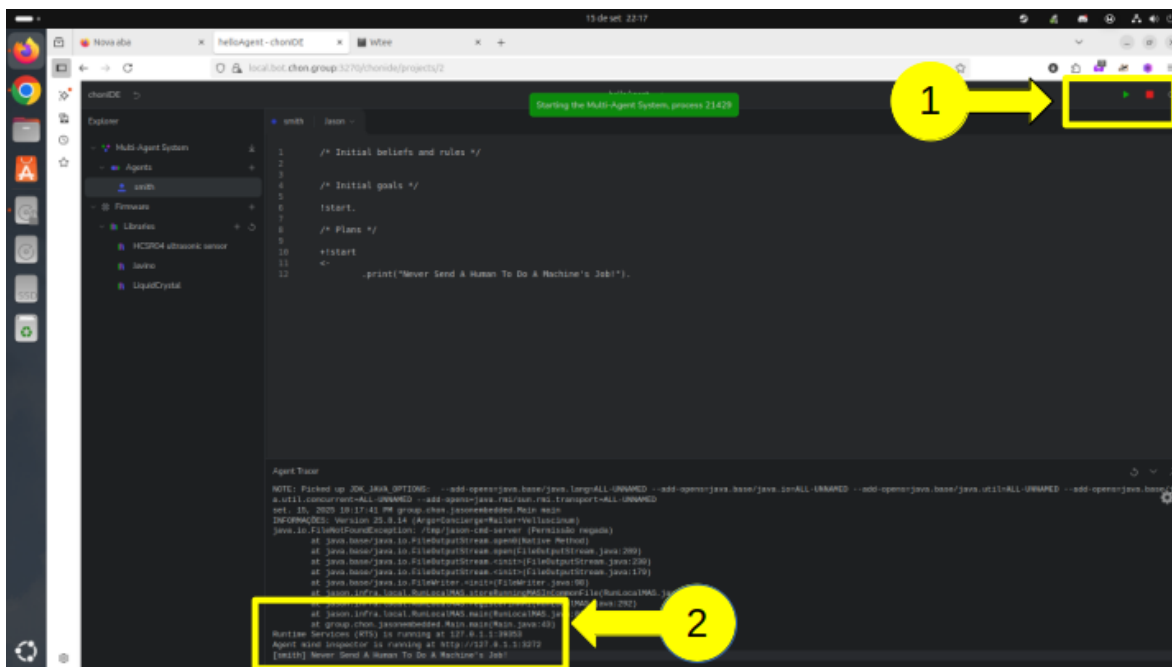


Figura 4: Verificação do SMA *helloAgent*

2.2 Testes dos simuladores

2.2.1 Webots

Primeiro, clone o projeto de exemplo com o git. Para isso, abra uma instância do terminal (CTRL + ALT + T), vá até a pasta desejada, e clone o projeto https://github.com/bptfreitas/FourWheels_With_ChonIDE_Webots.git . Por exemplo, para baixar o projeto de exemplo na pasta "Documentos" execute os comandos abaixo. A saída deverá ser tal como a Figura 5.

```
cd ~/Documentos ;  
git clone https://github.com/bptfreitas/FourWheels_With_ChonIDE_Webots.git ;
```

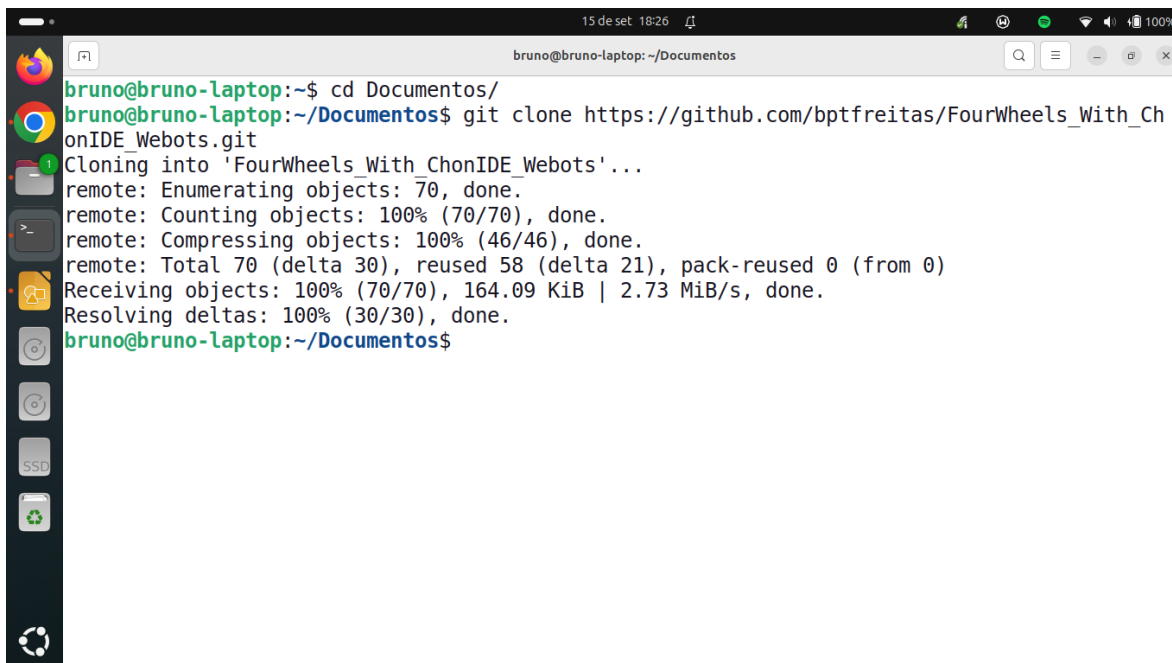
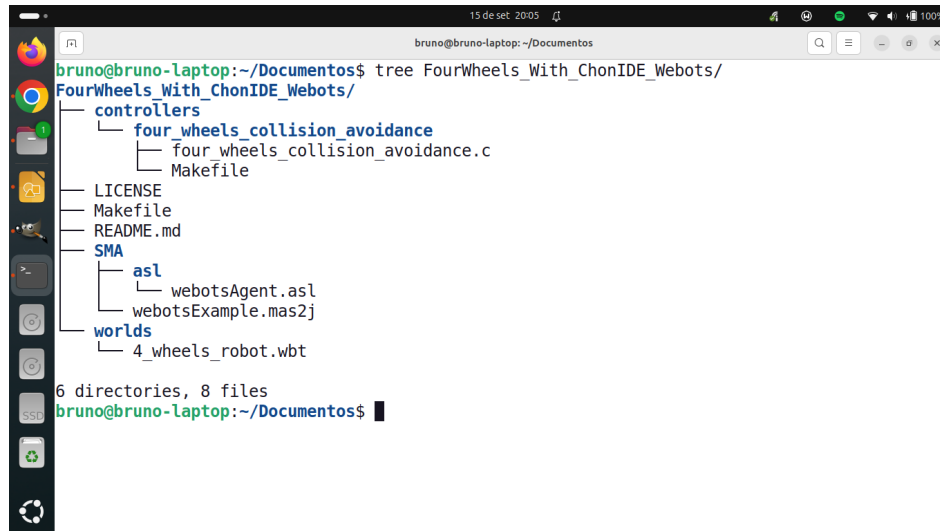
A screenshot of a Linux terminal window. The window title is "bruno@bruno-laptop: ~/Documentos". The terminal shows the following commands and output:
bruno@bruno-laptop:~\$ cd Documentos/
bruno@bruno-laptop:~/Documentos\$ git clone https://github.com/bptfreitas/FourWheels_With_ChonIDE_Webots.git
Cloning into 'FourWheels_With_ChonIDE_Webots'...
remote: Enumerating objects: 70, done.
remote: Counting objects: 100% (70/70), done.
remote: Compressing objects: 100% (46/46), done.
remote: Total 70 (delta 30), reused 58 (delta 21), pack-reused 0 (from 0)
Receiving objects: 100% (70/70), 164.09 KiB | 2.73 MiB/s, done.
Resolving deltas: 100% (30/30), done.
bruno@bruno-laptop:~/Documentos\$
The terminal has a dark background with a light-colored text. On the left side, there is a vertical dock with various application icons. The top of the window shows system status icons and the time "15 de set 18:26".

Figura 5: Baixando o projeto de exemplo do Webots

Em seguida, entre na pasta recém criada. A estrutura de pastas deverá aparecer tal como a Figura 6, onde:

- **controllers:** Código dos robôs do simulador
- **SMA:** O SMA usado para controlar o robô
- **worlds:** Descrição dos "mundos" do simulador



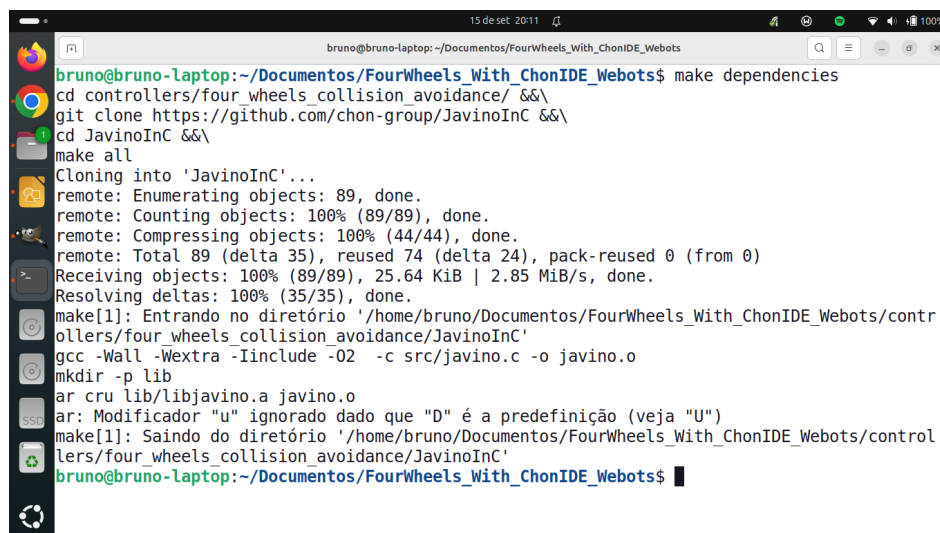
```
bruno@bruno-laptop: ~/Documentos
tree FourWheels_With_ChonIDE_Webots/
FourWheels_With_ChonIDE_Webots/
├── controllers
│   └── four_wheels_collision_avoidance
│       ├── four_wheels_collision_avoidance.c
│       └── Makefile
├── LICENSE
├── Makefile
├── README.md
├── SMA
│   └── asl
│       ├── webotsAgent.asl
│       └── webotsExample.mas2j
└── worlds
    └── 4_wheels_robot.wbt

6 directories, 8 files
bruno@bruno-laptop: ~/Documentos
```

Figura 6: Estrutura do projeto de teste do Webots

Com isso, na pasta do projeto, execute o comando abaixo para instalar as dependências. A saída deverá ser tal como a Figura 7:

```
cd FourWheels_With_ChonIDE_Webots/ ;
make dependencies ;
```



```
bruno@bruno-laptop: ~/Documentos/FourWheels_With_ChonIDE_Webots
make dependencies
cd controllers/four_wheels_collision_avoidance/ &&\
git clone https://github.com/chon-group/JavinoInC &&\
cd JavinoInC &&\
make all
Cloning into 'JavinoInC'...
remote: Enumerating objects: 89, done.
remote: Counting objects: 100% (89/89), done.
remote: Compressing objects: 100% (44/44), done.
remote: Total 89 (delta 35), reused 74 (delta 24), pack-reused 0 (from 0)
Receiving objects: 100% (89/89), 25.64 KiB | 2.85 MiB/s, done.
Resolving deltas: 100% (35/35), done.
make[1]: Entrando no diretório '/home/bruno/Documentos/FourWheels_With_ChonIDE_Webots/controllers/four_wheels_collision_avoidance/JavinoInC'
gcc -Wall -Wextra -Iinclude -O2 -c src/javino.c -o javino.o
mkdir -p lib
ar cru lib/libjavino.a javino.o
ar: Modificador "u" ignorado dado que "D" é a predefinição (veja "U")
make[1]: Saindo do diretório '/home/bruno/Documentos/FourWheels_With_ChonIDE_Webots/controllers/four_wheels_collision_avoidance/JavinoInC'
bruno@bruno-laptop: ~/Documentos/FourWheels_With_ChonIDE_Webots
```

Figura 7: Instalação das dependências do Webots

Com as dependências instaladas, entre na pasta *worlds* e inicie o Webots passando o mundo *4_wheels_robot.wbt* como parâmetro, conforme os comandos abaixo. Isso pode ser feito também utilizando a interface gráfica da ferramenta, em *File* → *Open world*.

```
cd worlds ;
webots 4_wheels_robot.wbt &
```

Ao iniciar o mundo, abra o arquivo da controladora, clicando no botão da pasta no canto superior direito (1). Em seguida, compile a controladora, clicando no botão de engrenagem no canto superior direito (2), indicado na Figura 8. Se tudo der certo, o simulador irá perguntar se você deseja reiniciar o mundo (Reset) (4) e o log na parte de baixo da ferramenta irá apresentar "*Build finished*" (4).

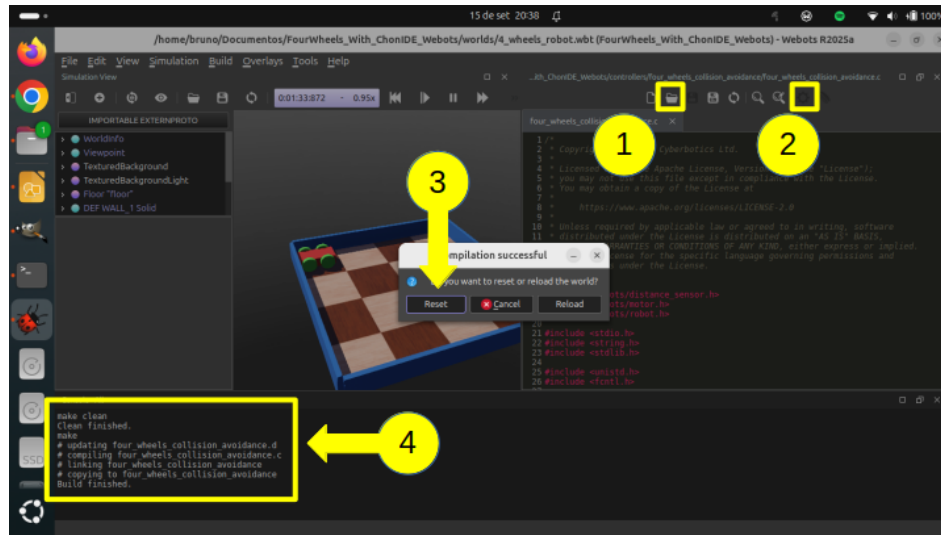


Figura 8: Compilação da controladora

Agora, entre na pasta SMA. Execute o Jason Embedded passando como parâmetro o arquivo *webotsExample.mas2j*. Se tudo estiver correto, você começará a ver o carrinho andar, conforme a Figura 9!

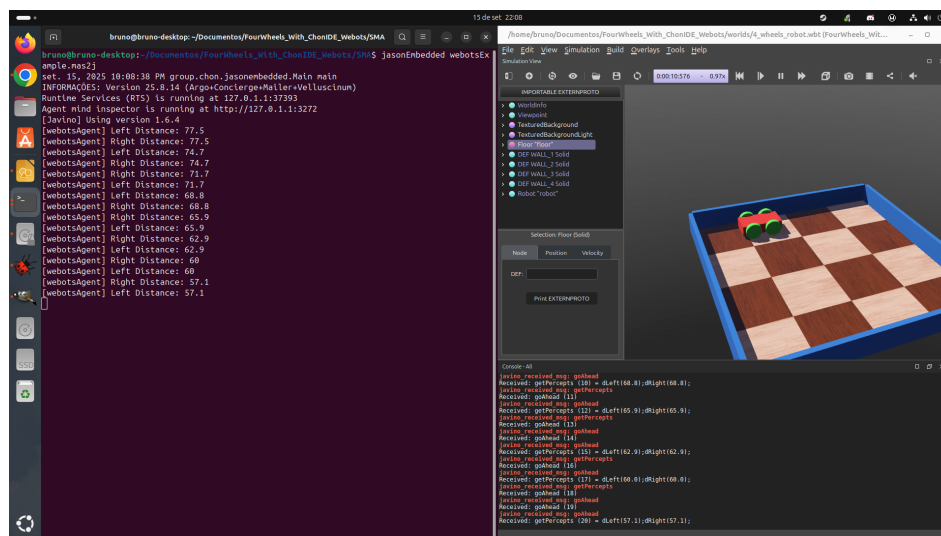


Figura 9: Execução bem sucedida do Webots com o jasonEmbedded

3 Rede

A abordagem Chon (Cognitive Hardware on Networks) busca integrar hardwares cognitivos através da rede. Dessa forma é necessário que os protótipos estejam integrados na mesma rede que os computadores que serão utilizados pelos alunos. Abaixo está descrito as necessidades de rede para uma experiência completa no curso. São elas:

- WLAN
- Middleware Context Net
- Autodescoberta.

3.1 Wireless Local Area Network (WLAN)

É necessário que o laboratório possua um Access Point para que os protótipos (ChonBots) e os computadores dos alunos (executando a ChonIDE) estejam na mesma rede local. Por questões de limitação do ChonOS, a senha de acesso à rede sem fio não pode conter caracteres especiais.

ATENÇÃO! Verifique se em sua instituição existe alguma restrição ao uso de redes sem fio no laboratório, tal como bloqueadores. Este é um fator impeditivo para a realização dos exercícios práticos.

A Figura 10 ilustra a estrutura da rede necessária para a realização das atividades.

Figura 10: Estrutura da rede

3.2 Middleware de comunicação IoT

Para possibilitar a comunicação entre os agentes embarcados nos protótipos é necessário acesso ao Middleware IoT ContextNet. É possível realizar de duas formas:

- Realizando a liberação no firewall;
- Executando um gateway localmente.

3.2.1 Liberação de portas no firewall para uso do Gateway público

Para permitir a comunicação utilizando o servidor comunitário do Chon Group, **solicite a liberação de acesso ao servidor no firewall de sua instituição**, conforme a Tabela 1.

Host	Porta	Protocolo
skynet.chon.group	5500	UDP
skynet2.chon.group	5500	UDP

Tabela 1: Hosts e portas a liberar

3.2.2 Servidor ContextNet local

Caso não seja possível liberar o acesso ao servidor comunitário do Chon Group, é possível executar um gateway ContextNet na Rede Local (<https://github.com/chon-group/contextNetServer/>). É possível executar usando uma máquina dedicada ou uma VM no VirtualBox. Na máquina que será o servidor local, execute os seguintes comandos no terminal:

```
echo "deb [trusted=yes] http://packages.chon.group/ chonos main" |
sudo tee /etc/apt/sources.list.d/chonos.list ;
sudo apt update ;
sudo apt install contextnet-server -y ;
```

3.2.3 Teste de conectividade com o Middleware Context-net

Para realizar o teste de funcionamento do Gateway IoT, acesse a ChonIDE conforme a Subseção 2.1.4 e abra o projeto skyNet, conforme Figura 11.

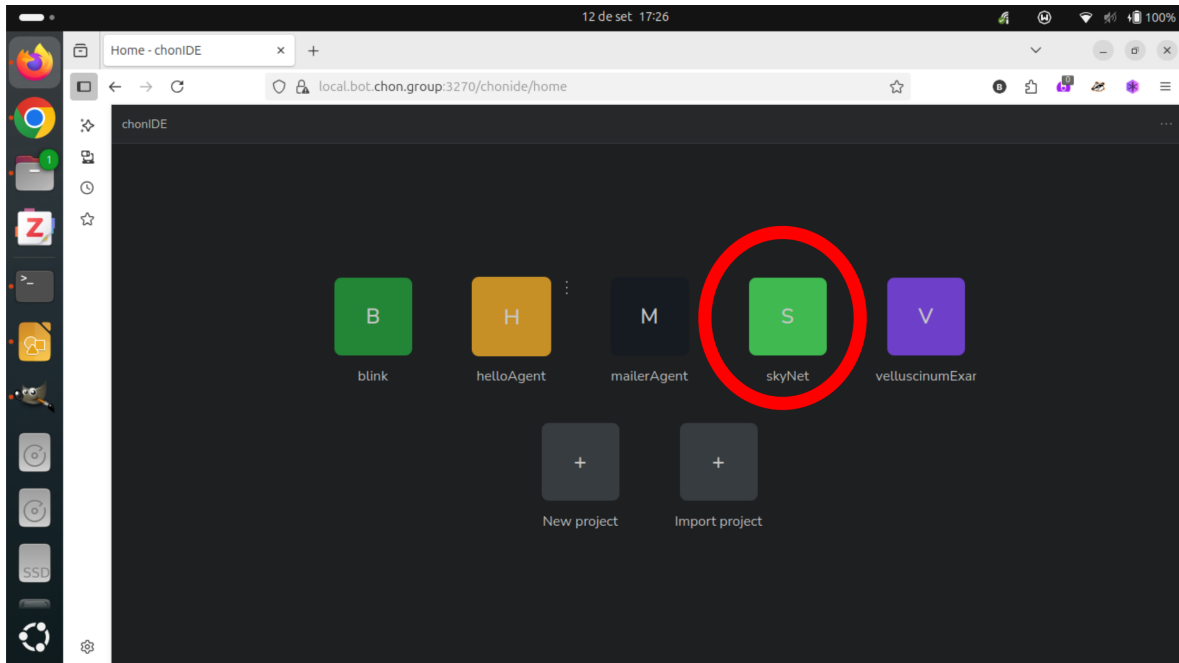


Figura 11: Abrindo o projeto *skyNet*

O exemplo padrão aponta para “*skynet.chon.group*” caso tenha instalado um Gateway local, insira o endereço do seu servidor (1). Após isso, execute o exemplo (2).

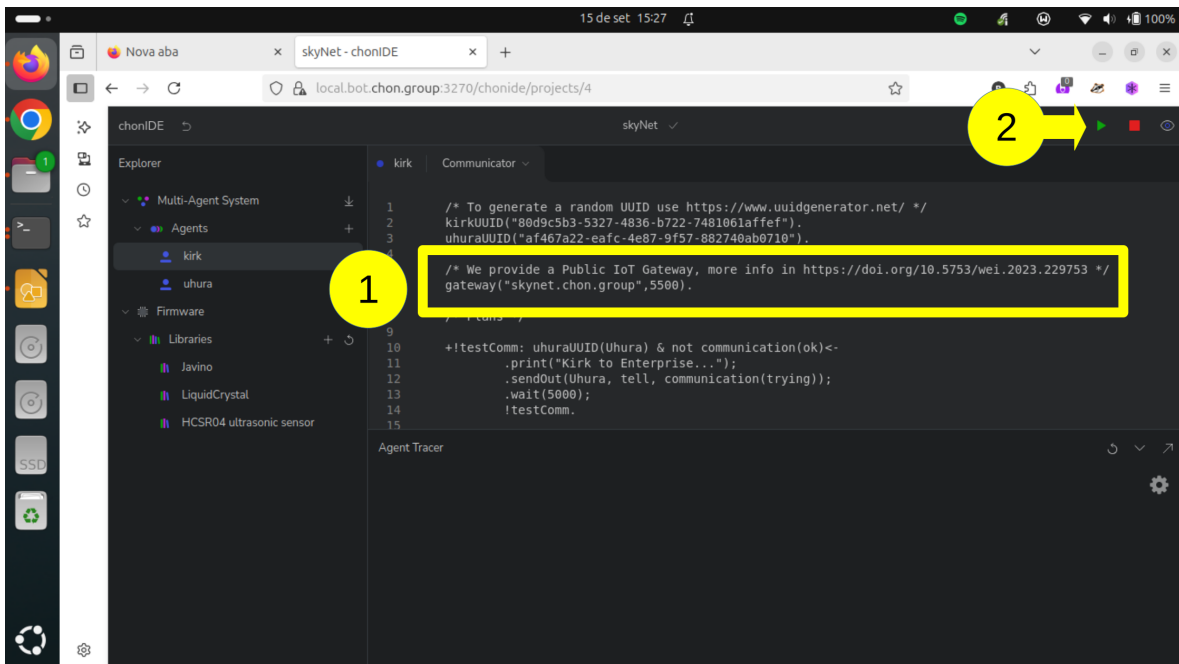


Figura 12: Executando o projeto *skyNet*

Para verificar o funcionamento, acesse o log do Sistema Multiagente, logo abaixo de seu código. Caso a mensagem “IoT Gateway is working!” aparecer na tela, sua instalação está correta. Parabéns!

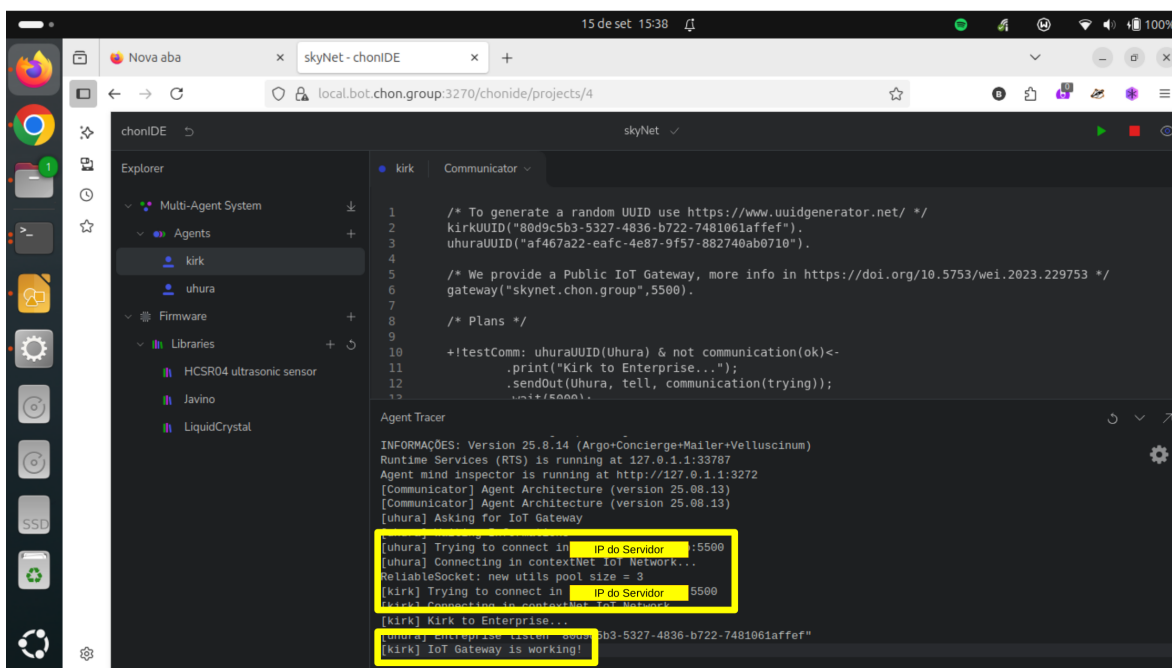


Figura 13: Verificação do funcionamento do Gateway IoT

3.3 Autodescoberta

Para que o acesso aos protótipos seja facilitado, provemos um serviço de DDNS, de forma que os protótipos registram o IP que receberam e assim os alunos não precisam se preocupar com saber endereço ou MAC do dispositivo.

Cada protótipo executa o serviço ChonOS DDNS Client que necessita de acesso ao servidor *ddns.chon.group*. Dessa forma, é necessário acesso direto (sem autenticação via proxy) para a página de atualização e acesso ao servidor de DNS do Chon Group. Abaixo estão as configurações necessárias para liberação junto ao setor de TI.

Além disso, para manter a data e hora dos protótipos atualizada, é necessário a que o firewall permita os pacotes NTP (Network Time Protocol) https://ntp.br/faq/#Como_configurar_firewall_para_. O ChonOS utiliza como padrão os servidores do NTP.br <https://ntp.br/>.

Endereço/Host	Porta	Protocolo
https://ddns.chon.group/update/index.php	443	TCP
ns1.chon.group	53	UDP
a.st1.ntp.br b.st1.ntp.br c.st1.ntp.br d.st1.ntp.br gps.ntp.br a.ntp.br b.ntp.br c.ntp.br	123	UDP
	4460	UDP

Tabela 2: Liberação de portas para NTP

3.3.1 Teste do serviço de autodescoberta

Para verificar se a conectividade com o serviço de DDNS do Chon Group, realize o procedimento descrito abaixo (atualmente só para Debian, Mint ou Ubuntu):

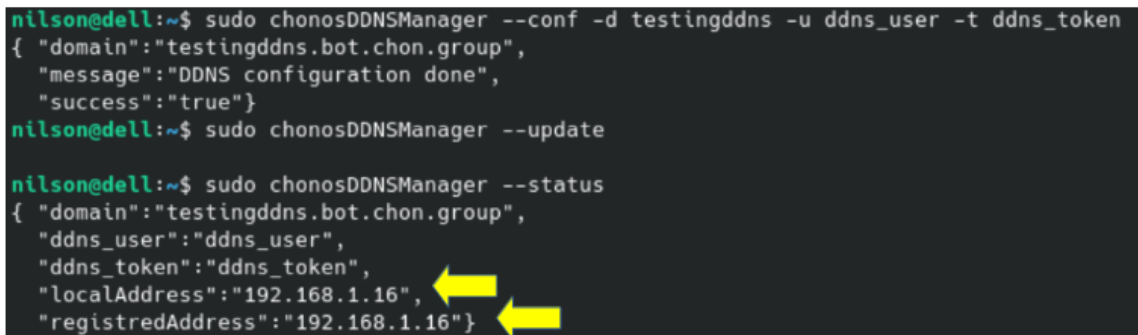
Primeiro é necessário que o cliente DDNS esteja instalado em seu computador. O comando abaixo realiza a instalação, caso necessário

```
echo "deb [trusted=yes] http://packages.chon.group/ chonos main" |  
  sudo tee /etc/apt/sources.list.d/chonos.list ;  
sudo apt update ;  
sudo apt install chonos-ddnsmng ;
```

Agora execute o script de atualização:direto no servidor DDNS, com a seguinte linha de comando:

```
sudo chonosDDNSManager --conf -d testingddns -u ddns_user -t ddns_token ;  
sudo chonosDDNSManager --update ;  
sudo chonosDDNSManager --status ;
```

Para confirmar a atualização no DDNS, verifique qual é o IP do seu computador e consulte a entrada do tipo A no registro de DDNS. Caso o IP informado pelo hostname (localAddress) seja igual ao informado pelo nslookup (registredAddress), significa que a conectividade está funcionando. Parabéns!



```
nilson@dell:~$ sudo chonosDDNSManager --conf -d testingddns -u ddns_user -t ddns_token  
{ "domain":"testingddns.bot.chon.group",  
  "message":"DDNS configuration done",  
  "success":"true"}  
nilson@dell:~$ sudo chonosDDNSManager --update  
  
nilson@dell:~$ sudo chonosDDNSManager --status  
{ "domain":"testingddns.bot.chon.group",  
  "ddns_user":"ddns_user",  
  "ddns_token":"ddns_token",  
  "localAddress":"192.168.1.16",  
  "registredAddress":"192.168.1.16"}
```

Figura 14: Verificação do funcionamento do serviço de Auto-descoberta

4 Bigchain DB

A integração de *Distributed Ledger Technologies* (DLT) e Sistemas Multiagente (SMA) facilita o acordo entre agentes cognitivos e também é útil para gerenciar relações de confiança em cenários distribuídos. Esta integração contém um grande conjunto de desafios abertos com grande potencial. Seja facilitando a execução de processos de negócios interorganizacionais semiautônomos ou mesmo permitindo que agentes inteligentes gerem valor econômico para seu proprietário.

Para possibilitar um cenário de negociação entre os agentes, utilizamos o BigChainDB, um banco de dados distribuído com características de Blockchain. É possível prover essa infraestrutura de duas formas:

- Realizando a liberação no firewall; ou
- Executando um gateway localmente.

4.1 Liberação de portas no firewall para uso da TestChain pública

Para permitir a comunicação utilizando o servidor comunitário do Chon Group, solicite a liberação de acesso ao servidor no firewall de sua instituição, conforme a Tabela 3.

Host	Porta	Protocolo
testchain.chon.group	9984	TCP
skynet2.chon.group	9984	TCP

Tabela 3: Hosts e portas a liberar para o Testchain

4.2 Servidor BigchainDB Local

Caso não seja possível liberar o acesso ao servidor comunitário do Chon Group, é possível executar um servidor BigChainDB Local <https://github.com/bigchaindb/bigchaindb>. É possível executar usando uma máquina dedicada ou uma VM no VirtualBox. Na máquina que será o servidor local, execute os seguintes comandos no terminal:

```
sudo apt update ;
sudo apt install docker-compose git make -y ;
git clone https://github.com/bigchaindb/bigchaindb.git ;
cd bigchaindb ;
sudo make run ;
```

4.3 Teste de Conectividade com o Servidor BigchainDB

Para realizar o teste de funcionamento da DLT, acesse a ChonIDE e abra o projeto velluscinumExample, conforme a Figura 15.

O exemplo padrão aponta para “testchain.chon.group”, conforme a a Figura 16. Caso tenha instalado um servidor local, insira o endereço do seu servidor (1). Após isso, execute o exemplo (2). Acesse o log do Sistema Multiagente (3). Caso a mensagem “Creating Asset... [pushed] [successfully]” aparecer, sua instalação está correta. Tais passos podem ser vistos na

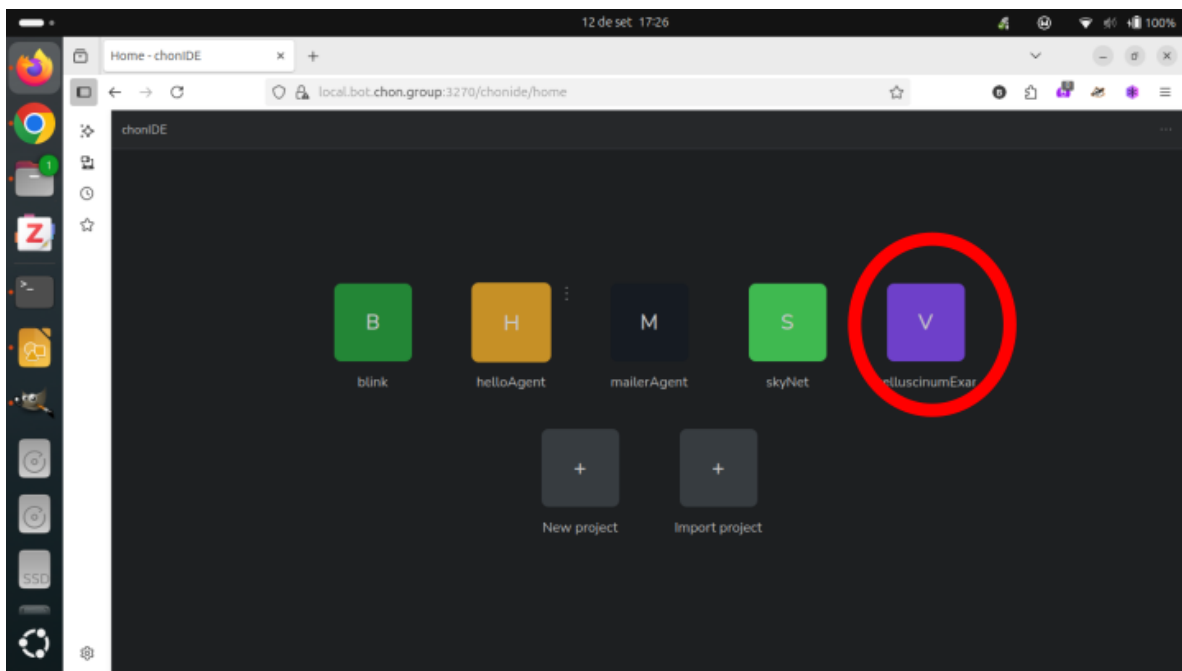


Figura 15: Abrindo projeto Velluscinum agent

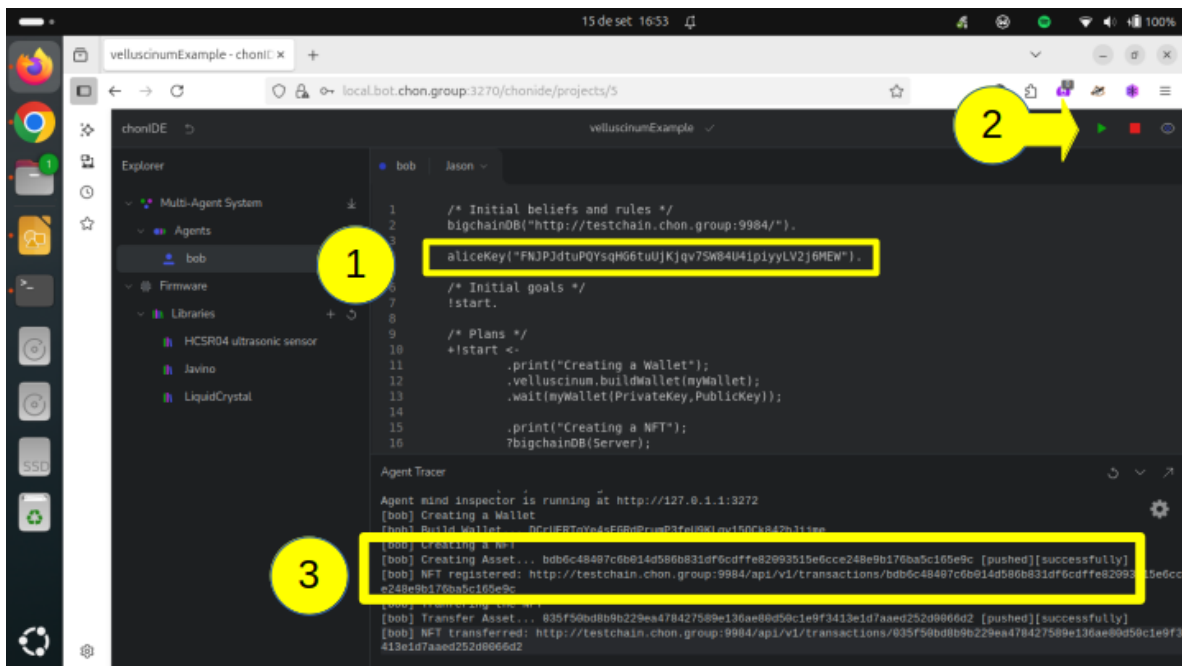


Figura 16: Iniciando e verificando funcionamento do SMA Velluscinum agent