

# Agentes Autônomos

Simulação Multiagente de uma  
Peixaria utilizando *Jason* e  
*CARTAGO*

Olimar Teixeira Borges



# *CONTEXTO...*

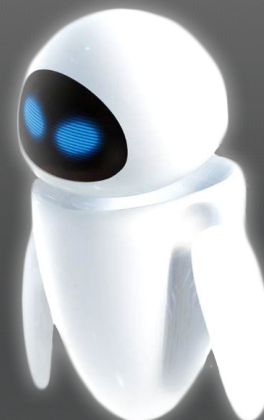


Para o desenvolvimento deste trabalho, foi escolhido o contexto de uma empresa de peixaria, que contemplará desde a pescaria em alto mar até o transporte dos peixes em terra.





# PROBLEMA...



O problema é verificar quais são as configurações de barcos e caminhões mais adequadas para uma melhor eficiência na pesca e transporte de peixes.



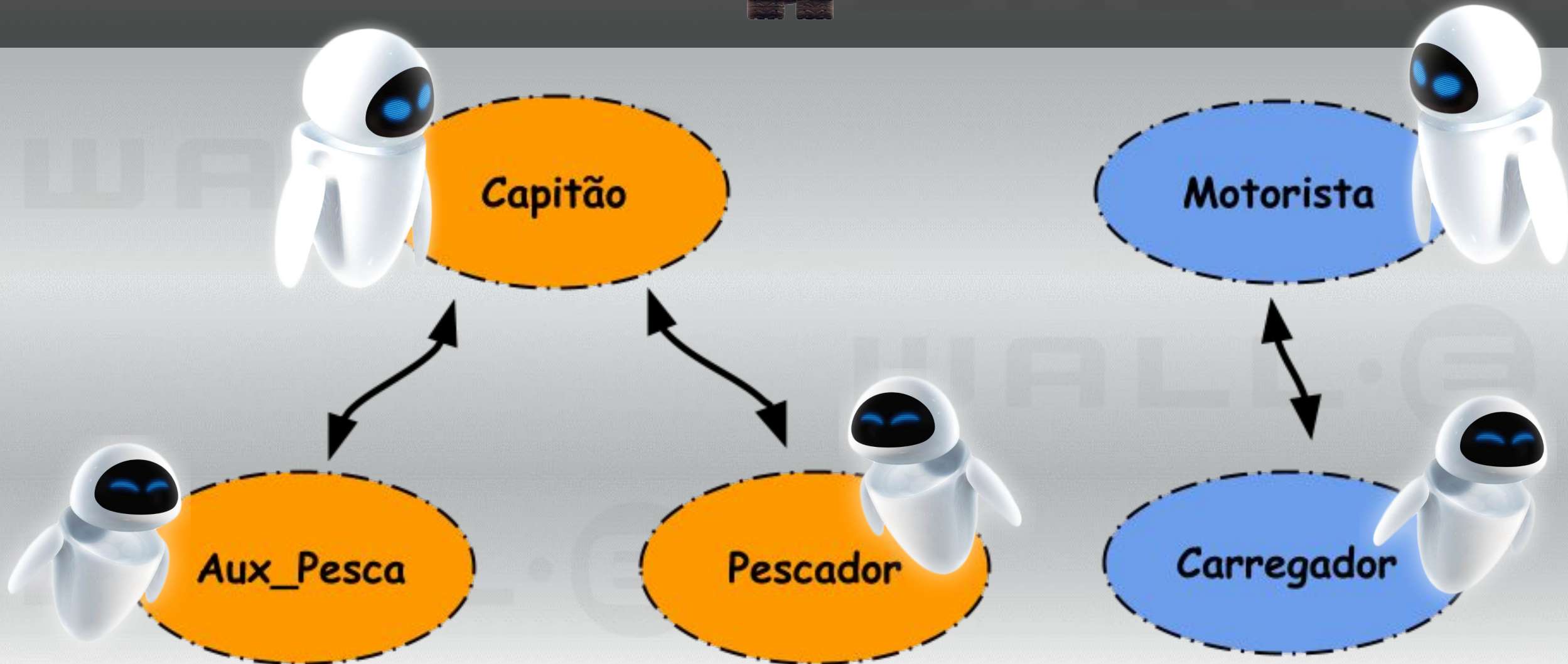
# AGENTES...



- CAPITÃO: comanda o Barco e inicia a navegação pelo Oceano.
- PESCADOR: responsável pela pescaria, joga as redes e captura os peixes do Oceano.
- AUX\_PESCA: responsável por recolher e jogar a âncora do Barco e por descarregar os peixes quando estiver no Porto.
- MOTORISTA: dirige o Caminhão e inicia o carregamento de peixes no Porto.
- CARREGADOR: carrega os peixes no Caminhão do Porto e descarrega do Caminhão no Distribuidor.



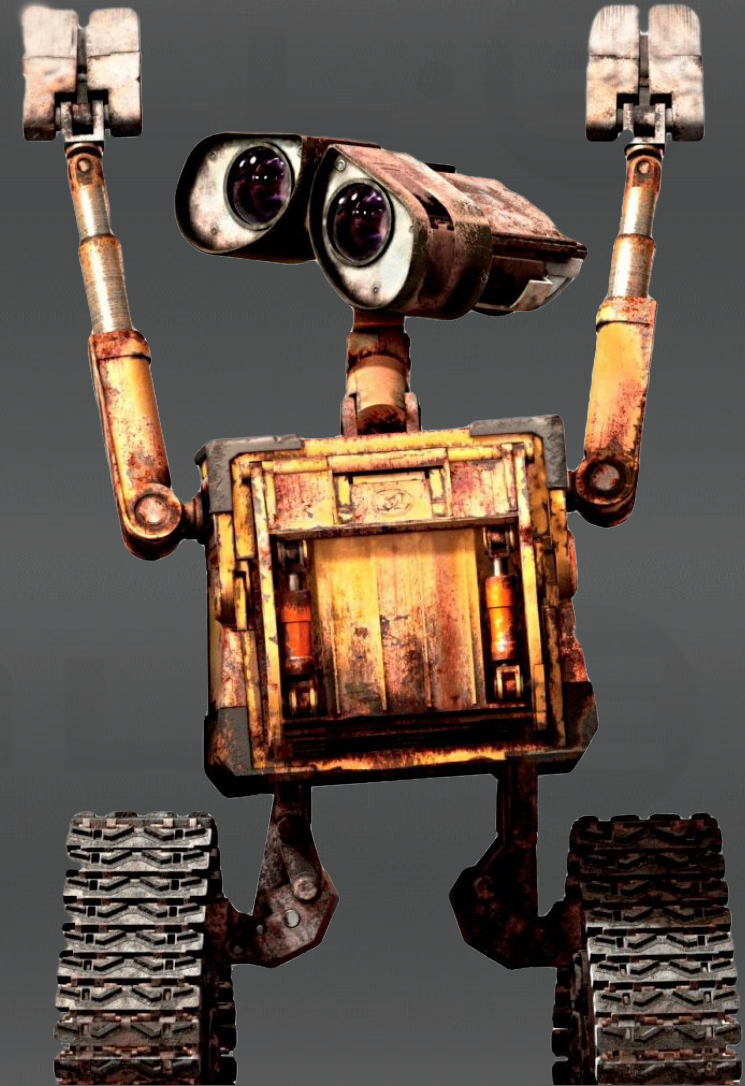
# AGENTES...





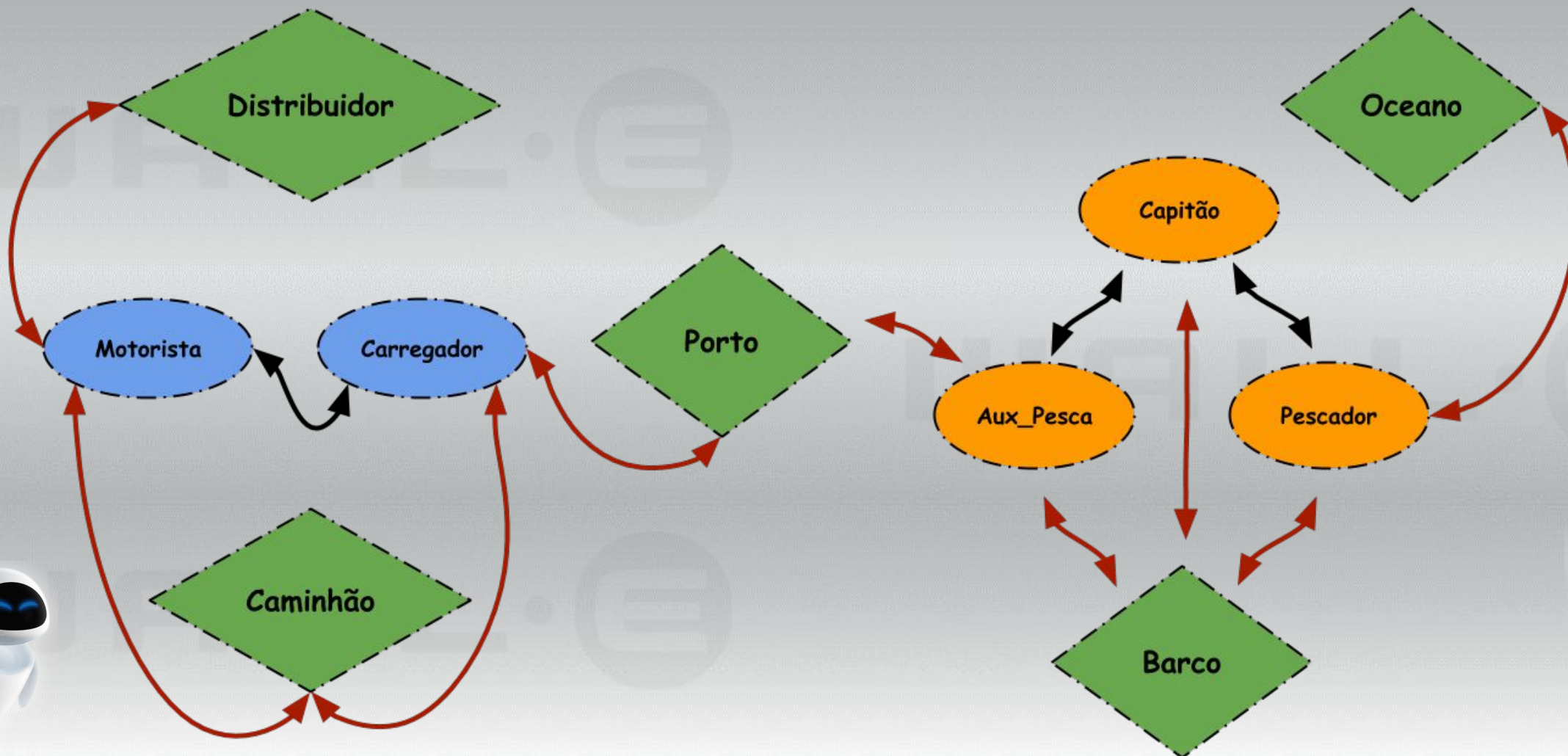
# ARTEFATOS...

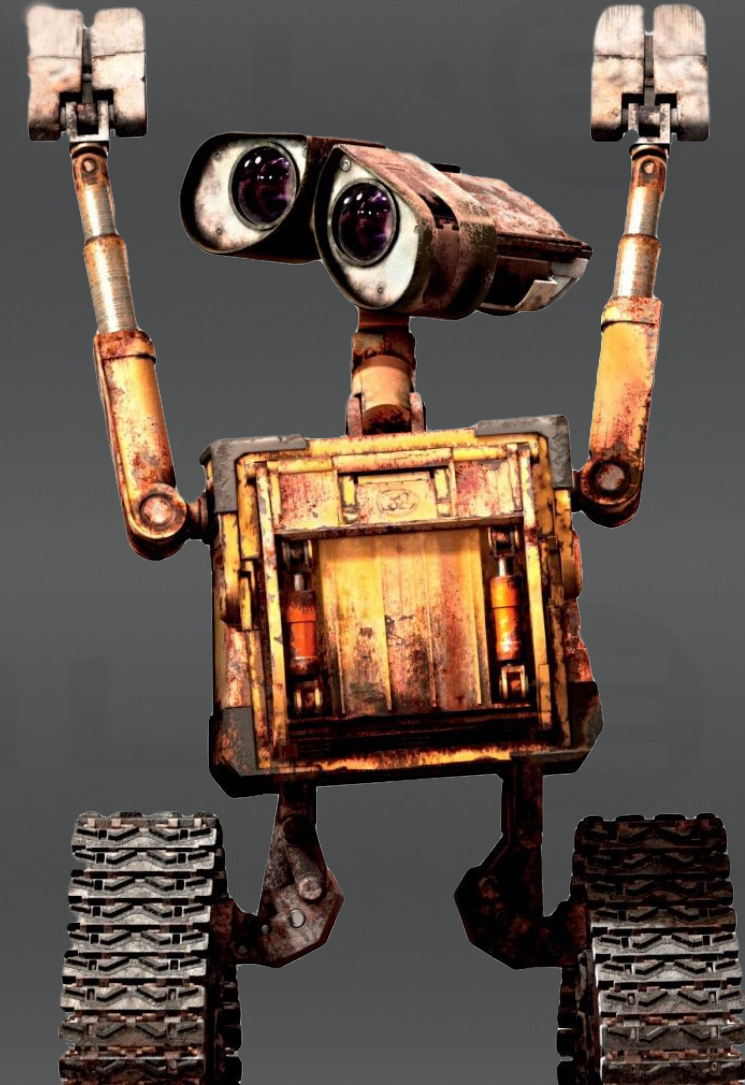
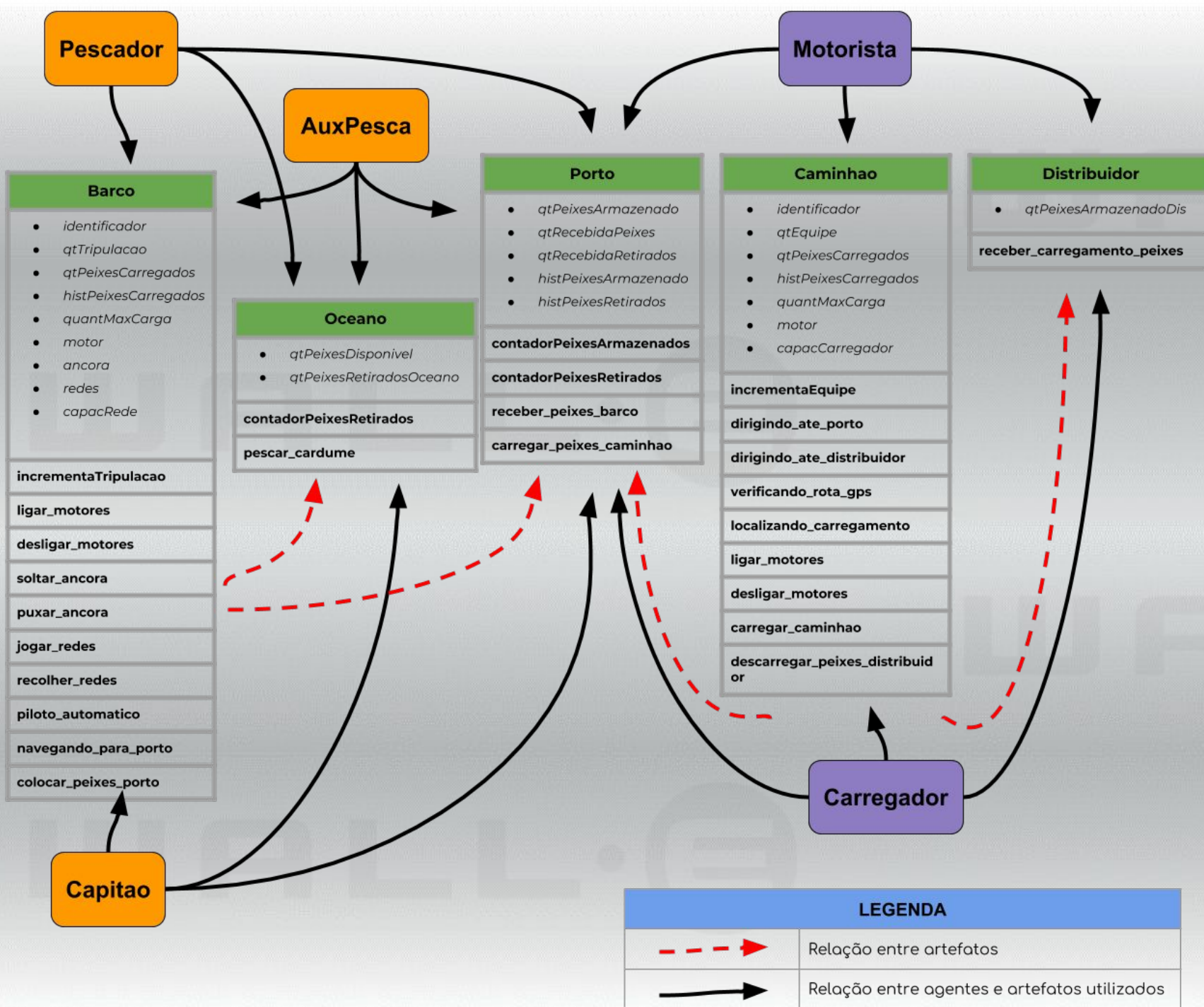
- Artefatos CArtAgO:
  - *BARCO*
  - *OCEANO*
  - *PORTO*
  - *CAMINHÃO*
  - *DISTRIBUIDOR*





# ARTEFATOS & AGENTES...

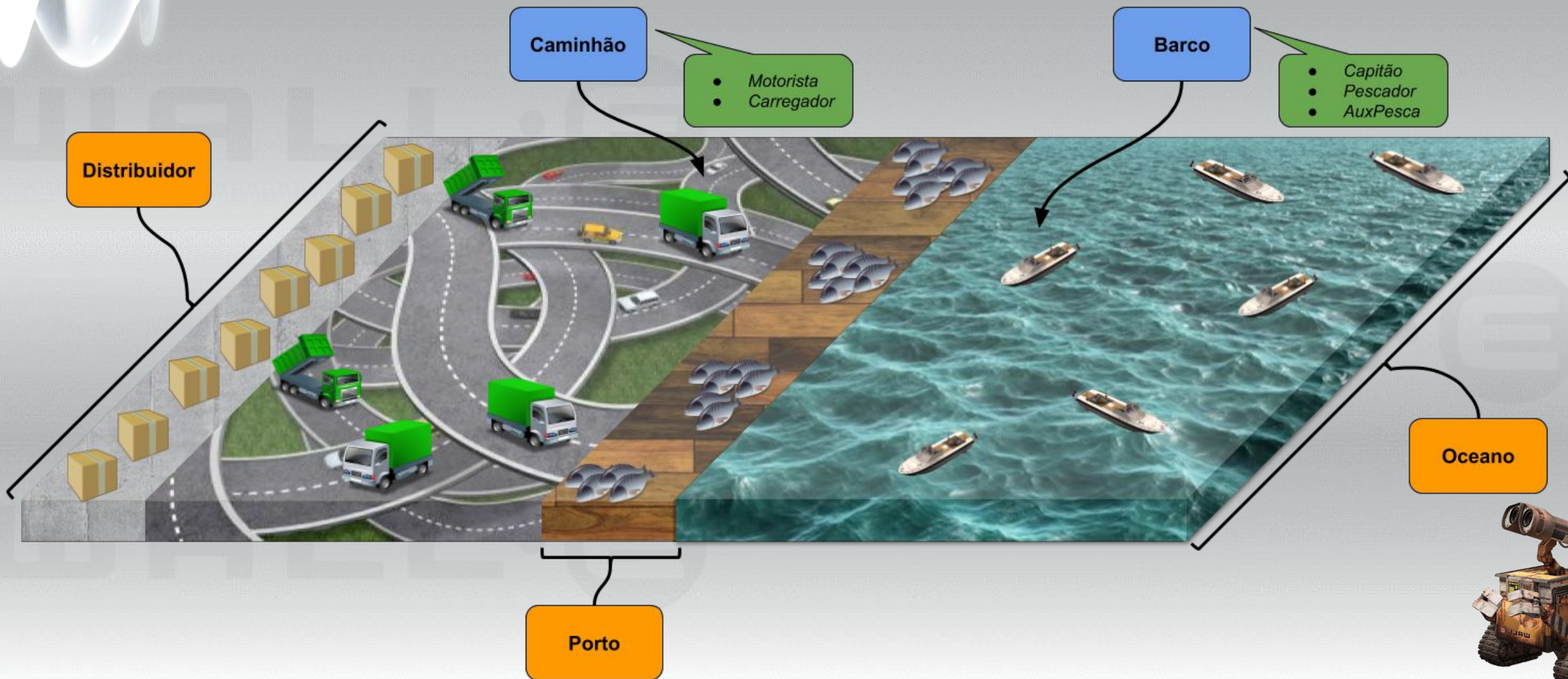








# VISÃO GERAL...







# Programando em *Jason*





EM ALTO MAR...





```
!iniciarNavegacao.
```

```
+!iniciarNavegacao
```

```
  : .my_name(NomeAgent) & identificador(IdBarco)
  <-
  .concat("barco_0", IdBarco, NomeBarco);
  .print("Nome do Barco - Capitao: ", NomeBarco);
  lookupArtifact(NomeBarco, ArtId);
  focus(ArtId);
  incrementaTripulacao(IdBarco, NomeAgent);
  .print("Eu sou o: ", NomeAgent, " do: ", NomeBarco);
  +meuBarcoEh(NomeBarco, IdBarco);
  !verificarPeixesOceano;
```

```
+!verificarPeixesOceano
```

```
  : qtPeixesDisponivel(QuantPeixesDisp) & QuantPeixesDisp > 0
  & qtTripulacao(QuantTripbarco) & QuantTripbarco < 3
  <-
  !montarTripulacao;
```

```
+!montarTripulacao
```

```
  : meuBarcoEh(NomeBarco, IdBarco)
  <-
  .concat("pescadorAg", IdBarco, NomePescador);
  .concat("auxPescaAg", IdBarco, NomeAuxPesca);
  .create_agent(NomePescador, "pescador.asl");
  .create_agent(NomeAuxPesca, "aux_pesca.asl");
  //Adiciona à sua base de crença o nome de seus companheiros de tripulacao
  +nomePescador(NomePescador);
  +nomeAuxPesca(NomeAuxPesca);
  .print("criou o agente: ", NomePescador);
  .print("criou o agente: ", NomeAuxPesca);
  .send(NomePescador, tell, meuBarcoEh(NomeBarco, IdBarco));
  .send(NomeAuxPesca, tell, meuBarcoEh(NomeBarco, IdBarco));
  !verificarPeixesOceano;
```



Aux\_Pesca

Pescador

```
+meuBarcoEh(NomeBarco, IdBarco)
: .my_name(NomeAgent)
<-
//Entrando no mesmo ambiente
joinWorkspace("peixaria",NomeWorkspace);
//Focando no Barco
.concat("barco_0", IdBarco, NomeBarco);
lookupArtifact(NomeBarco, ArtBarcoId)[wid(NomeWorkspace)];
focus(ArtBarcoId);
incrementaTripulacao(IdBarco, NomeAgent);
.print("Eu sou o: ", NomeAgent, " do: ", NomeBarco);
//Adiciona à sua base de crença o nome de seus companheiros de tripulacao
.concat("capitaoAg", IdBarco, NomeCapitao);
.concat("pescadorAg", IdBarco, NomePescador);
+nomeCapitao(NomeCapitao);
+nomePescador(NomePescador);
//Focando no Oceano
lookupArtifact(oceano, ArtOceanoId)[wid(NomeWorkspace)];
focus(ArtOceanoId);

//Focando no Porto
lookupArtifact(porto, ArtPortoId)[wid(NomeWorkspace)];
focus(ArtPortoId);
```

```

+!verificarPeixesOceano
: qtPeixesDisponivel(QuantPeixesDisp) & QuantPeixesDisp > 0 & qtTripulacao(QuantTripbarco)
& QuantTripbarco == 3
<-
?nomeAuxPesca(NomeAuxPesca);
.print("CAPITÃO verifica com o AUX_PESCA se ele está pronto para iniciar a Pescaria...");
.send(NomeAuxPesca, achieve, iniciar_pescaria);
.

```

Capitão

```

+!iniciar_pescaria
: meuBarcoEh(_, IdBarco)
<-
.print("AUX_PESCA verifica se está pronto para iniciar a Pescaria...");
?nomeCapitao(NomeCapitao);
.print("AUX_PESCA responde ao CAPITÃO que está pronto para iniciar a Pescaria...");
!puxar_ancora;
.send(NomeCapitao, achieve, auxPescaPronto);
.

+!puxar_ancora <-
.print("...puxando ancora para iniciar a navegacao em mar aberto...");
puxar_ancora; //função do artefato Barco
.

```

Aux\_Pesca

Capitão

```

+!auxPescaPronto <-
?nomePescador(NomePescador);
?nomeAuxPesca(NomeAuxPesca);
.print("CAPITÃO verifica com o PESCADOR se ele está pronto para iniciar a Pescaria...");
.send(NomePescador, achieve, iniciar_pescaria);
.

```



```
+!iniciar_pescaria
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & QuantPeixes < CapaMaxArm
<-
.print("PESCADOR verifica se tem capacidade disponível no Barco para iniciar uma pescaria...");
?nomeCapitao(NomeCapitao);
.print("PESCADOR responde ao CAPITÃO que está pronto para iniciar a Pescaria...");
.send(NomeCapitao, achieve, pescadorPronto);
.
```

Pescador

```
+!pescadorPronto <-
?nomePescador(NomePescador);
!ligar_motor;
.

+!ligar_motor <-
.print("...ligando os motores para iniciar a navegacao em mar aberto...");
ligar_motores; //função do artefato Barco
!localizar_cardume;
.

+!localizar_cardume <-
?nomePescador(NomePescador);
.print("...se preparando para localizar os cardumes no mar...");
piloto_automatico; //função do artefato Barco
.print("cardume encontrado...");
.print("CAPITÃO pede para o PESCADOR jogar as redes no Oceano...");
.send(NomePescador, achieve, jogar_redes);
.
```

Capitão

```
+!jogar_redes : quantMaxCarga(QtMax) & qtPeixesCarregados(QtCarregados) & QtCarregados < QtMax
<-
.print("...jogando as redes nos cardumes...");
jogar_redes; //função do artefato Barco
!recolher_redes;
.

+!jogar_redes : quantMaxCarga(QtMax) & qtPeixesCarregados(QtCarregados) & QtCarregados >= QtMax
<-
!recolher_redes;
.
```

```
+!recolher_redes
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & capacRede(Capac) & (Capac+QuantPeixes)>CapaMaxArm
<-
?nomeCapitao(NomeCapitao);
.print("Quantidade atual de Peixes no Barco: ", QuantPeixes);
.print("...01_recolhendo as redes para ir em direção ao Porto...");
.print("PESCADOR pede ao CAPITÃO para irem ao Porto descarregar os peixes no Barco...");
.send(NomeCapitao, achieve, nav_porto);
.

+!recolher_redes
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & qtPeixesDisponivel(QtRestante) & QtRestante<=0
<-
?nomeCapitao(NomeCapitao);
.print("Quantidade atual de Peixes no Barco: ", QuantPeixes);
.print("...02_recolhendo as redes para ir em direção ao Porto...");
.print("PESCADOR pede ao CAPITÃO para irem ao Porto descarregar os peixes no Barco...");
.send(NomeCapitao, achieve, nav_porto);
.

+!recolher_redes
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & qtPeixesDisponivel(QtRestante) & QtRestante>=0
<-
.print("...03_recolhendo as redes com os peixes pescados...");
lookupArtifact(oceano, ArtOceanoId); //busca o identificador do artefato Oceano
recolher_redes(ArtOceanoId); //função do artefato Barco
//consulta na base de crenças do agente
?qtPeixesCarregados(QuantAtualPeixes);
!jogar_redes;
.
```



Capitão

Aux\_Pesca

```
+!nav_porto <-  
  .print("...navegando em direção ao Porto para descarregar...");  
  navegando_para_porto; //função do artefato Barco  
  !desligar_motores;  
.  
  
+!desligar_motores <-  
  ?nomeAuxPesca(NomeAuxPesca);  
  .print("...desligando os motores...");  
  desligar_motores; //função do artefato Barco  
  .print("CAPITÃO pede para o AUX_PESCA soltar as âncoras no Oceano...");  
  .send(NomeAuxPesca, achieve, soltar_ancora);  
.
```

```
+!soltar_ancora <-  
  .print("...soltar âncora para iniciar a parada...");  
  soltar_ancora; //função do artefato Barco  
  !retir_peixes;  
.  
  
+!retir_peixes <-  
  ?nomeCapitao(NomeCapitao);  
  .print("...retirando peixes do Barco...");  
  lookupArtifact(porto, ArtId); //busca o identificador do artefato Porto  
  colocar_peixes_porto(ArtId); //função do artefato Barco  
  
  .print("AUX_PESCA pede ao CAPITÃO para iniciarem a busca por peixes no Oceano novamente, após já terem descarregado sua carga no Porto...");  
  .send(NomeCapitao, achieve, verificarPeixesOceano);  
.
```

EM TERRA...

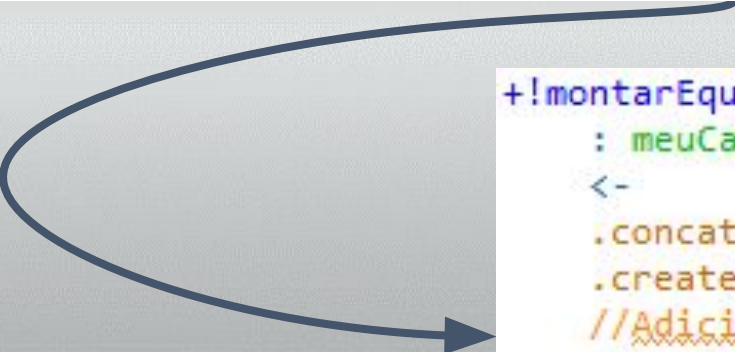




Motorista

```
+!iniciarTransporte
: .my_name(NomeAgent) & identificador(IdCaminhao)
<-
.concat("caminhao_0", IdCaminhao, NomeCaminhao);
.print("Nome do Caminhão - Motorista: ", NomeCaminhao);
lookupArtifact(NomeCaminhao, ArtId);
.print("ArtId do Caminhão - Motorista: ", ArtId);
focus(ArtId);
incrementaEquipe(IdCaminhao, NomeAgent);
.print("Eu sou o: ", NomeAgent, " do: ", NomeCaminhao);
+meuCaminhaoEh(NomeCaminhao, IdCaminhao);
!verificarPeixesPorto;
.

+!verificarPeixesPorto
: qtPeixesArmazenado(QuantPeixesArm) & QuantPeixesArm > 0 & qtEquipe(QuantEqCaminhao) & QuantEqCaminhao < 2
<-
!montarEquipe;
.
```



```
+!montarEquipe
: meuCaminhaoEh(NomeCaminhao, IdCaminhao)
<-
.concat("carregadorAg", IdCaminhao, NomeCarregador);
.create_agent(NomeCarregador, "carregador.asl");
//Adiciona à sua base de crença o nome de seus companheiros de equipe
+nomeCarregador(NomeCarregador);
.print("criou o agente: ", NomeCarregador);
.send(NomeCarregador, tell, meuCaminhaoEh(NomeCaminhao, IdCaminhao));
!verificarPeixesPorto;
.
```

```
+meuCaminhaoEh(NomeCaminhao, IdCaminhao)
: .my_name(NomeAgent)
<-
//Entrando no mesmo ambiente
joinWorkspace("peixaria", NomeWorkspace);|
//Focando no Caminhão
.concat("caminhao_0", IdCaminhao, NomeCaminhao);
lookupArtifact(NomeCaminhao, ArtCaminhaoId)[wid(NomeWorkspace)];
focus(ArtCaminhaoId);
incrementaEquipe(IdCaminhao, NomeAgent);
.print("Eu sou o: ", NomeAgent, " do: ", NomeCaminhao);
//Adiciona à sua base de crença o nome de seus companheiros de equipe
.concat("motoristaAg", IdCaminhao, NomeMotorista);
+nomeMotorista(NomeMotorista);
//Focando no Distribuidor
lookupArtifact(distribuidor, ArtDistribuidorId)[wid(NomeWorkspace)];
focus(ArtDistribuidorId);
//Focando no Porto
lookupArtifact(porto, ArtPortoId)[wid(NomeWorkspace)];
focus(ArtPortoId);
```



+!verificarPeixesPorto

```
: qtPeixesArmazenado(QuantPeixesArm) & QuantPeixesArm > 0 & qtEquipe(QuantEqCaminhao) & QuantEqCaminhao == 2
<-
?nomeCarregador(NomeCarregador);
.print("MOTORISTA verifica com o CARREGADOR se ele está pronto para iniciar o Carregamento...");
.send(NomeCarregador, achieve, buscar_peixes_porto);
```

Motorista

+!buscar\_peixes\_porto

```
: meuCaminhaoEh(_, IdCaminhao)
<-
?nomeMotorista(NomeMotorista);
.print("CARREGADOR responde ao MOTORISTA que está pronto para iniciar o Carregamento...");
.send(NomeMotorista, achieve, carregadorPronto);
```

Carregador

Motorista

```
+!carregadorPronto <-
?nomeCarregador(NomeCarregador);
!dirigir_ate_porto;

+!dirigir_ate_porto
<-
.print("...dirigindo ate o porto para buscar carregamento de peixes...")
dirigindo_ate_porto; //função do artefato Caminhao
!desligar_caminhao;

+!desligar_caminhao
<-
?nomeCarregador(NomeCarregador);
.print("...desligando o caminhao...")
desligar_motores; //função do artefato Caminhao
.print("MOTORISTA pede ao CARREGADOR que localize o carregamento de peixes no Porto...");
.send(NomeCarregador, achieve, localizar_carregamento);
```

```
+!carregar_caminhao
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & capacCarregador(Capac) & (Capac+QuantPeixes) > CapaMaxArm
<-
?nomeMotorista(NomeMotorista)
.print("...01_preparando o Caminhao para ir em direção ao Distribuidor...");
.print("CARREGADOR pede ao MOTORISTA para irem em direção ao Distribuidor descarregar, pois o Caminhão já está carregado...");
.send(NomeMotorista, achieve, ligar_caminhao);
.

+!carregar_caminhao
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & qtPeixesArmazenado(QtRestante) & QtRestante<=0
<-
?nomeMotorista(NomeMotorista)
.print("...02_preparando o Caminhao para ir em direção ao Distribuidor...");
.print("CARREGADOR pede ao MOTORISTA para irem em direção ao Distribuidor descarregar, pois o Caminhão já está carregado...");
.send(NomeMotorista, achieve, ligar_caminhao);
.

+!carregar_caminhao
: quantMaxCarga(CapaMaxArm) & qtPeixesCarregados(QuantPeixes) & qtPeixesArmazenado(QtRestante) & QtRestante>=0
<-
.print("...03_carregando o caminhao com o carregamento de peixes...")
lookupArtifact(porto, ArtId); //busca o identificador do artefato Porto
carregar_caminhao(ArtId); //função do artefato Caminhao
//consulta na base de crenças do agente
?qtPeixesCarregados(QuantTotalPeixes);
.print("Quantidade de peixes retirados do Porto e recolhidos para este Caminhao: ", QuantTotalPeixes); //-QuantPeixes
!carregar_caminhao ;
.
```



```
+!ligar_caminhao
  <-
  .print("...ligando o motor do caminhao para ir em direção ao Distribuidor...")
  ligar_motores; //função do artefato Caminhao
  !verificar_rota_gps;
.
+!verificar_rota_gps
  <-
  .print("...verificando o melhor caminho até o distribuidor...")
  verificando_rota_gps; //função do artefato Caminhao
  !dirigir_ate_distribuidor;
.
+!dirigir_ate_distribuidor
  <-
  .print("...dirigindo ate o distribuidor...")
  dirigindo_ate_distribuidor; //função do artefato Caminhao
  !descarr_peixes;
.
+!descarr_peixes <-
  .print("...retirando peixes do Caminhão e descarregando no Distribuidor...");
  lookupArtifact(distribuidor, ArtId); //busca o identificador do artefato Distribuidor
  descarregar_peixes_distribuidor(ArtId); //função do artefato Caminhão
  !verificando_peixes_porto;
.
+!verificando_peixes_porto
  : qtPeixesArmazenado(QuantPeixesArm) & QuantPeixesArm > 0
  <-
  .print("...ainda existem peixes a serem carregados no Porto...");
  !dirigir_ate_porto;
.
+!verificando_peixes_porto
  : qtPeixesArmazenado(QuantPeixesArm) & QuantPeixesArm <= 0
  <-
  .print("...neste momento não existem peixes a serem carregados no Porto... aguardando sinal do Porto...");
.
```



*SIMULAÇÕES...*





# CONFIGURAÇÕES DAS SIMULAÇÕES...

Simulações	Artefatos	Ident.	Capacidade Redes	Capacidade Armazenamento	Artefatos	Ident.	Capacidade Carregador	Capacidade Carregamento
01	Barco01	1	25	200	Caminhao01	1	25	200
02	Barco01	1	25	50	Caminhao01	1	25	100
	Barco02	2	25	50	Caminhao02	2	10	100
03	Barco01	1	30	100	Caminhao01	1	50	100
	Barco02	2	15	200	Caminhao02	2	50	100
	Barco03	3	100	100	Caminhao03	3	50	100
	Barco04	4	200	200	Caminhao04	4	50	100



# SIMULAÇÃO 01...

Simulações	Artefatos	Ident.	Capacidade Redes	Capacidade Armazenamento	Artefatos	Ident.	Capacidade Carregador	Capacidade Carregamento
01	Barco01	1	25	200	Caminhao01	1	25	200



# SIMULAÇÃO 01...

```
mas peixaria {

    //barco_01 #####
    agent capitaoAg1 : capitao.asl{ //capitao1
        focus: oceano, porto, barco_01
    }
    //caminhao_01 #####
    agent motoristaAg1 : motorista.asl{ //motorista1
        focus: distribuidor, porto, caminhao_01
    }

    //Artefatos
    workspace peixaria{

        /*Artefato: Barco
        * @parametro1: identificador
        * @parametro2: capacidade máxima de peixes na rede por pescador
        * @parametro3: quantidade máxima de carga permitida de peixes
        */
        artifact barco_01: peixaria.Barco(1, 25, 200)

        /*Artefato: Caminhao
        * @parametro1: identificador
        * @parametro2: capacidade máxima de peixes que o carregador consegue carregar por vez
        * @parametro3: quantidade máxima de carga permitida de peixes
        */
        artifact caminhao_01: peixaria.Caminhao(1, 25, 200)

        /*Artefato: Oceano*/
        artifact oceano: peixaria.Oceano()
        /*Artefato: Porto*/
        artifact porto: peixaria.Porto()
        /*Artefato: Distribuidor*/
        artifact distribuidor: peixaria.Distribuidor()
    }

    // agent source path
    asl-path: src/agt, src/agt/inc
}
```





# Resultados da Simulação 01...



Artefatos	Histórico de Peixes Retirados	Histórico de Peixes Carregados	Histórico de Peixes Armazenados
Barco01	-	-	65400
Caminhao01	-	65200	-
Oceano	65400	-	-
Porto	65200	65400	-
Distribuidor	-	-	65200





# Resultados da Simulação 01...

- Para esta configuração, os resultados mostram que quando os artefatos **Barco e Caminhão** possuem as mesmas configurações, as quantidades de **peixes retirados, carregados e armazenados**, ficam muito próximas.
- A única observação desta primeira observação foi que quando a simulação foi parada, o **Caminhão** ainda estava em deslocamento ao **Porto** e por isso a quantidade de **peixes carregados no Caminhão** era menor que a quantidade de **peixes carregados no Porto**.



# SIMULAÇÃO 02...

Simulações	Artefatos	Ident.	Capacidade Redes	Capacidade Armazenamento	Artefatos	Ident.	Capacidade Carregador	Capacidade Carregamento
02	Barco01	1	25	50	Caminhao01	1	25	100
	Barco02	2	25	50	Caminhao02	2	10	100



# SIMULAÇÃO 02...



```
mas peixaria {
  //barco_01 #####
  agent capitaoAg1 : capitao.asl{ //capitao1
    focus: oceano, porto, barco_01
  }
  //barco_02 #####
  agent capitaoAg2 : capitao.asl{ //capitao2
    focus: oceano, porto, barco_02
  }
  //caminhao_01 #####
  agent motoristaAg1 : motorista.asl{ //motorista1
    focus: distribuidor, porto, caminhohao_01
  }
  //caminhao_02 #####
  agent motoristaAg2 : motorista.asl{ //motorista2
    focus: distribuidor, porto, caminhohao_02
  }
}

workspace peixaria{
  /*Artefato: Barco
  * @parametro1: identificador
  * @parametro2: capacidade máxima de peixes na rede por pescador
  * @parametro3: quantidade máxima de carga permitida de peixes
  */
  artifact barco_01: peixaria.Barco(1, 25, 50)
  artifact barco_02: peixaria.Barco(2, 25, 50)
  /*Artefato: Caminhao
  * @parametro1: identificador
  * @parametro2: capacidade máxima de peixes que o carregador consegue carregar por vez
  * @parametro3: quantidade máxima de carga permitida de peixes
  */
  artifact caminhohao_01: peixaria.Caminhao(1, 25, 100)
  artifact caminhohao_02: peixaria.Caminhao(2, 10, 100)
  /*Artefato: Oceano*/
  artifact oceano: peixaria.Oceano()
  /*Artefato: Porto*/
  artifact porto: peixaria.Porto()
  /*Artefato: Distribuidor*/
  artifact distribuidor: peixaria.Distribuidor()
}
```



# Resultados da Simulação 02...



Artefatos	Histórico de Peixes Retirados	Histórico de Peixes Carregados	Histórico de Peixes Armazenados
Barco01	-	-	25950
Barco02	-	-	26000
Caminhao01	-	34615	-
Caminhao02	-	13865	-
Oceano	52000	-	-
Porto	51900	51950	-
Distribuidor	-	-	51380



# Resultados da Simulação 02...



- Mantendo os Barcos com as mesmas capacidades, a quantidade de peixes armazenados por ambos, continua se mantendo muito próxima, mas como suas capacidades de armazenamento são baixas, eles precisam se deslocar mais vezes entre Oceano e Porto para pescar e descarregar.
- Enquanto que os Caminhões com capacidades de carregamento diferentes mas com capacidades de armazenamento iguais, são mais eficientes, quando sua capacidade de carregamento é maior.



# SIMULAÇÃO 03...

Simulações	Artefatos	Ident.	Capacidade Redes	Capacidade Armazenamento	Artefatos	Ident.	Capacidade Carregador	Capacidade Carregamento
03	Barco01	1	30	100	Caminhao01	1	50	100
	Barco02	2	15	200	Caminhao02	2	50	100
	Barco03	3	100	100	Caminhao03	3	50	100
	Barco04	4	200	200	Caminhao04	4	50	100



# SIMULAÇÃO 03...

```
mas peixaria {
  //barco_01 #####
  agent capitaoAg1 : capitao.asl{ //capitao1
    focus: oceano, porto, barco_01
  }
  //barco_02 #####
  agent capitaoAg2 : capitao.asl{ //capitao2
    focus: oceano, porto, barco_02
  }
  //barco_03 #####
  agent capitaoAg3 : capitao.asl{ //capitao3
    focus: oceano, porto, barco_03
  }
  //barco_04 #####
  agent capitaoAg4 : capitao.asl{ //capitao4
    focus: oceano, porto, barco_04
  }

  //caminhao_01 #####
  agent motoristaAg1 : motorista.asl{ //motorista1
    focus: distribuidor, porto, caminho_01
  }
  //caminhao_02 #####
  agent motoristaAg2 : motorista.asl{ //motorista2
    focus: distribuidor, porto, caminho_02
  }
  //caminhao_03 #####
  agent motoristaAg3 : motorista.asl{ //motorista3
    focus: distribuidor, porto, caminho_03
  }
  //caminhao_04 #####
  agent motoristaAg4 : motorista.asl{ //motorista4
    focus: distribuidor, porto, caminho_04
  }
}
```

```
workspace peixaria{
  /*Artefato: Barco
  * @parametro1: identificador
  * @parametro2: capacidade máxima de peixes na rede por pescador
  * @parametro3: quantidade máxima de carga permitida de peixes
  */
  artifact barco_01: peixaria.Barco(1, 25, 50)
  artifact barco_02: peixaria.Barco(2, 25, 100)
  artifact barco_03: peixaria.Barco(3, 25, 200)
  artifact barco_04: peixaria.Barco(4, 25, 300)
  /*Artefato: Caminhao
  * @parametro1: identificador
  * @parametro2: capacidade máxima de peixes que o carregador consegue carregar por vez
  * @parametro3: quantidade máxima de carga permitida de peixes
  */
  artifact caminho_01: peixaria.Caminhao(1, 50, 100)
  artifact caminho_02: peixaria.Caminhao(2, 50, 100)
  artifact caminho_03: peixaria.Caminhao(3, 50, 100)
  artifact caminho_04: peixaria.Caminhao(4, 50, 100)
  /*Artefato: Oceano*/
  artifact oceano: peixaria.Oceano()
  /*Artefato: Porto*/
  artifact porto: peixaria.Porto()
  /*Artefato: Distribuidor*/
  artifact distribuidor: peixaria.Distribuidor()
}
// agent source path
asl-path: src/agt, src/agt/inc
```





# Resultados da Simulação 03...



Artefatos	Histórico de Peixes Retirados	Histórico de Peixes Carregados	Histórico de Peixes Armazenados
Barco01	-	-	36000
Barco02	-	-	46215
Barco03	-	-	50300
Barco04	-	-	100600
Caminhao01	-	52900	-
Caminhao02	-	52015	-
Caminhao03	-	51430	-
Caminhao04	-	51680	-
Oceano	233220	-	-
Porto	233115	233115	-
Distribuidor	-	-	241995





# *Resultados da Simulação 03...*

- A capacidade de armazenamento de um Barco é determinante durante a pescaria, ou seja, Barcos com capacidades de armazenamento maiores e de carregamento menores, tendem a pescar mais peixes no final de uma simulação, no mesmo tempo de execução, que Barcos com capacidades de carregamento maiores e de armazenamento menores.
- E Caminhões com capacidades iguais, tendem a carregar quantidades de peixes proporcionalmente idênticas.



# SIMULAÇÕES



- Após a execução de ambas as simulações, conclui-se que:
  - A **capacidade de armazenamento** dos artefatos **Barco e Caminhão** é a variável independente predominante nos resultados.
  - Mesmo que suas capacidades de redes ou carregamento sejam pequenas, o que importa nestas simulações é o **armazenamento de cada um destes artefatos**.
  - Ou seja, o Barco pode jogar redes várias vezes, mas para ser eficiente, ele tem que ir **menos vezes no Porto**, da mesma forma que o Caminhão ao Distribuidor, no mesmo tempo de execução.

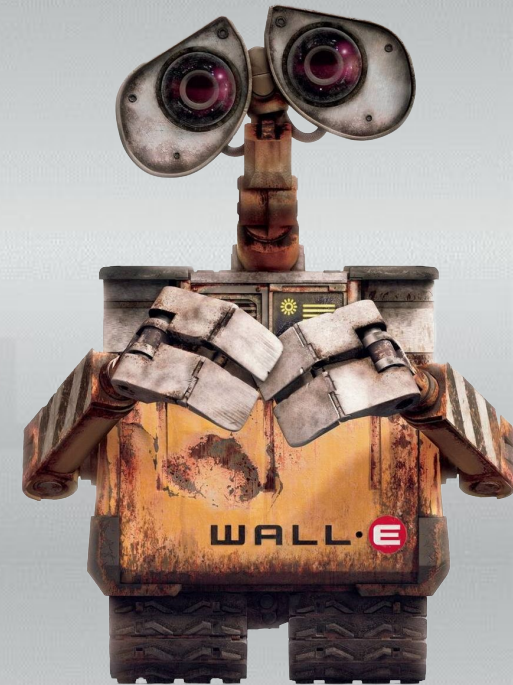


# Considerações Finais



# Considerações Finais...

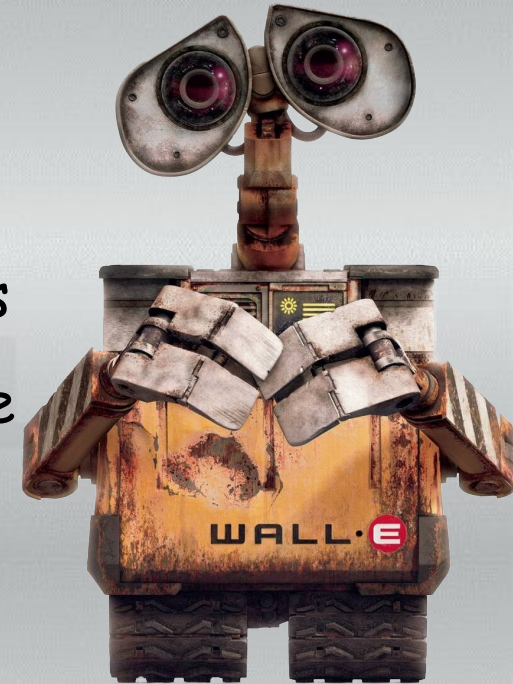
- Esta aplicação foi desenvolvida durante a disciplina de **Agentes Autônomos** e demonstrou, de maneira prática, como utilizar os conceitos de programação orientada a agentes.
- O autor deste artigo já havia realizado a disciplina de Multiagentes e portanto, algumas das funcionalidades fornecidas pelo **Jason** e pelo **CARTAgO** já haviam sido exploradas anteriormente.





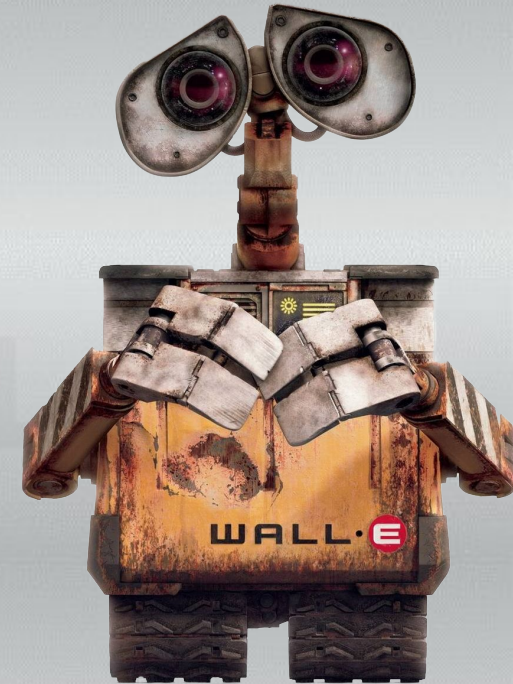
# Considerações Finais...

- Na aplicação desenvolvimento na disciplina de Multiagentes, os agentes executavam suas interações **uma única vez** e em seguida paravam a simulação.
- Para esta aplicação, os agentes **não param** de realizar suas ações enquanto o sistema não é finalizado. Nesta aplicação os agentes se comunicam e enviam informações uns aos outros.
- Desta forma, o aplicação atual se torna **mais completa**, já que fica possível simular várias configurações, por períodos de tempo variados.



# Considerações Finais...

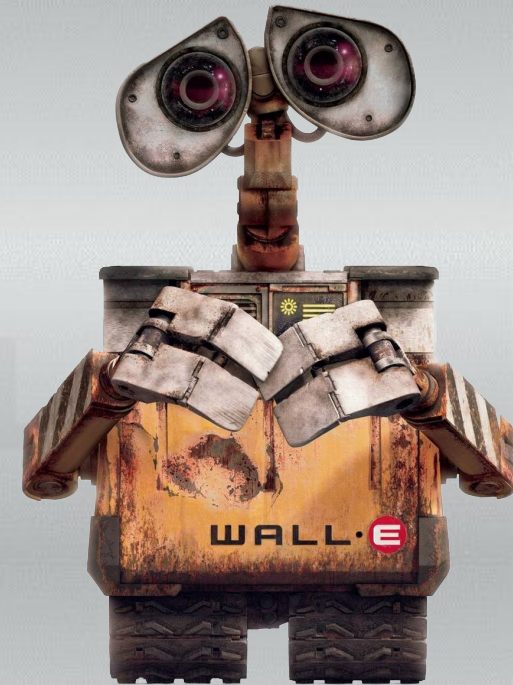
- As dificuldades encontradas durante o desenvolvimento deste trabalho foram muito parecidas com o desenvolvimento do anterior, principalmente na programação da linguagem Jason.
- O desenvolvimento também trouxe grandes aprendizados para o autor na área de programação orientada a agentes. Ao final deste desenvolvimento a sintaxe da linguagem Jason ficou muito mais clara para o autor, do que na disciplina anterior.





# Considerações Finais...

- Como trabalho futuro, a interação no Oceano poderia ser repensada, para que os Barcos naveguem por mais tempo a procura de cardumes e também poderia ser criado um artefato "Estrada", para que os Caminhões interagissem mais durante o percurso de carga e descarga de peixes. E uma interface gráfica do sistema, seria bem interessante para visualizar melhor as interações entre os agentes.



# Obrigado!







Perguntas?



# Agentes Autônomos

Simulação Multiagente de uma  
Peixaria utilizando *Jason* e  
*CARTAGO*

Olimar Teixeira Borges

