

VisEdu – Aquário Virtual

Simulador de Ecossistema Utilizando

Animação Comportamental

Aluno: Kevin Eduard Piske

Orientador: Dalton Solano dos Reis



FURB - Universidade Regional de Blumenau
DSC - Departamento de Sistemas e Computação
GCG - Grupo de Pesquisa em Computação Gráfica,
Processamento de Imagens e Entretenimento Digital
<http://gcg.inf.furb.br>

Roteiro

- introdução;
- objetivos;
- fundamentação teórica;
- trabalhos correlatos;
- requisitos;
- especificação;
- implementação;
- operacionalidade da implementação;
- resultados e discussões;
- conclusões e sugestões.

Introdução

- aulas tradicionais X Tecnologia;
- uso de simuladores no auxílio da educação;
- ecossistema – biocenose – biótopo.

Objetivos

Desenvolver um simulador de ecossistema de aquário marinho.

- estender o módulo de raciocínio desenvolvido por Feltrin (2014);
- ter um ambiente que permita a inserção de agentes dotados de representações gráficas;
- adicionar funcionalidades que permitam gerar animações comportamentais.

Fundamentação Teórica

- simulador:
 - conjunto de hardware e software.
- simulação:
 - processo de imitar a realidade.

Fundamentação Teórica

- modelo *Belief Desire Intention* – BDI:
 - crenças;
 - desejos;
 - intenções.
- animação comportamental:
 - percepção;
 - raciocínio;
 - ação.

Trabalhos Correlatos

- MASSIVE:
 - simulador de multidões;
 - espaço tridimensional;
 - trabalha com agentes inteligentes;
 - permite a simulação de milhões de agentes simples e até 100.000 agentes complexos;
 - utiliza o conceito de animação comportamental.

Trabalhos Correlatos



Trabalhos Correlatos

- Fish School and Obstacles:
 - aplicação WEB;
 - espaço bidimensional;
 - simula comportamentos de peixes:
 - formação de cardumes;
 - desviar de obstáculos;
 - fugir de predadores;
 - trabalha com agentes inteligentes (implícito);
 - utiliza o conceito de animação comportamental.

Trabalhos Correlatos

The screenshot displays a NetLogo simulation window. At the top, there is a toolbar with standard icons. Below the toolbar, on the left, are two blue buttons: "setup" and "go" (with a circular arrow icon). To the right of these buttons are several sliders, each with a green bar and a numerical value:

Slider Name	Value
num-prey	102
accounted-prey	2
num-predators	2
randomness	25
sensory-range	50
frame-world	0
dist-repulsive	4
dist-parallel	33
update-neighbourhood-every	100 steps
obstacles	4
latency	5

The main simulation area is a large black rectangle. It contains several red rectangular obstacles of varying sizes and positions. A cluster of small yellow dots, representing prey, is located in the lower-middle part of the area. A single red star-like shape, representing a predator, is positioned near the bottom center, close to the prey cluster. A small red cross is visible on the left side of the simulation area.

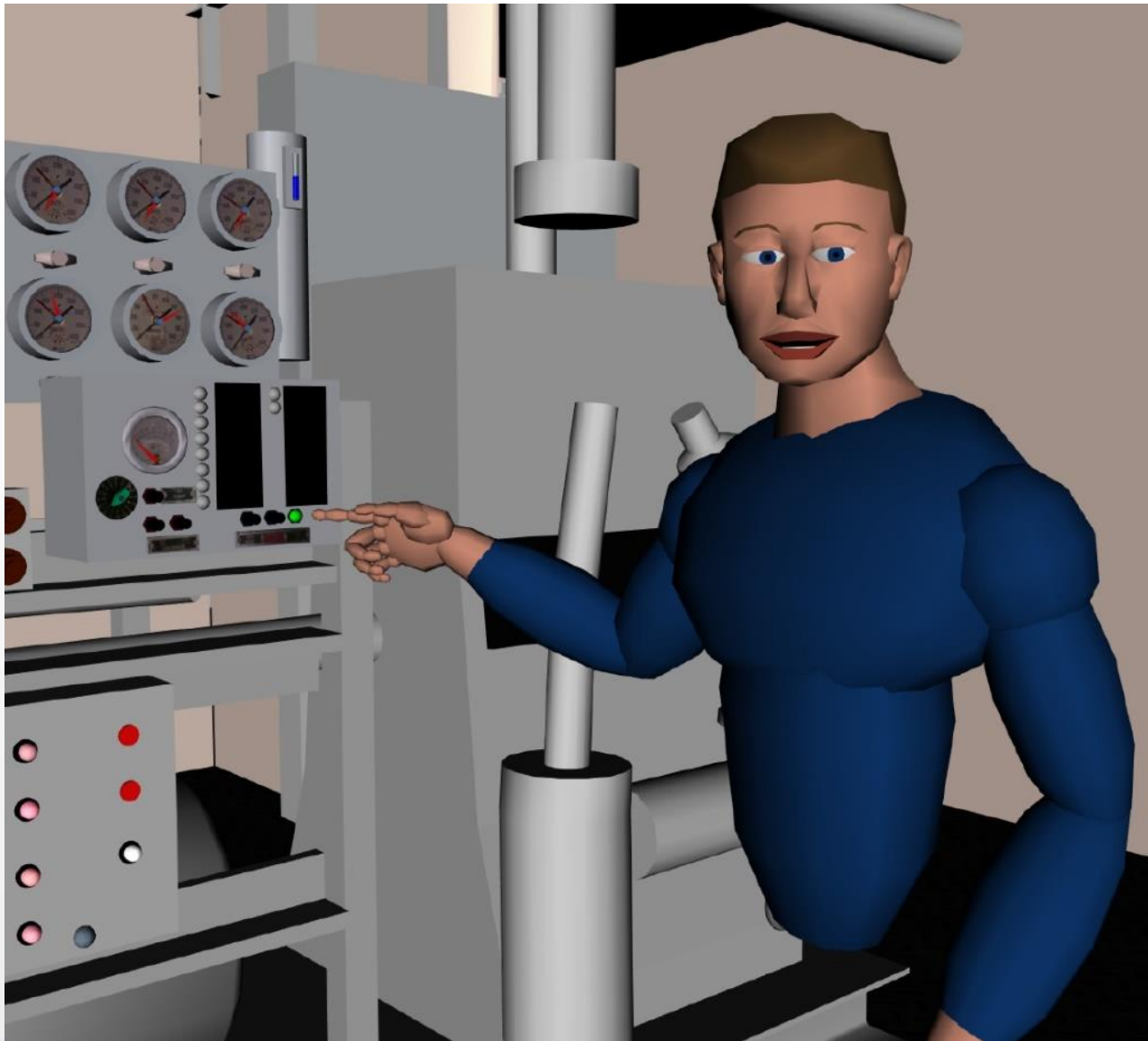
Click [here](#) to go back.

URB
UNIVERSIDADE DE BLUMENAU

Trabalhos Correlatos

- STEVE:
 - tutor virtual:
 - interage com alunos;
 - demonstra ações;
 - usa o olhar e gestos para direcionar a atenção;
 - auxilia na educação;
 - espaço tridimensional;
 - trabalha com planos e metas;
 - utiliza o conceito de animação comportamental.

Trabalhos Correlatos



Requisitos

- Aquário Virtual:
 - implementar controle orbital de câmera (RF);
 - possuir ao menos um predador e uma presa (RF);
 - possuir uma câmera secundária para exibir a visão dos peixes (RF);
 - permitir que somente a visão do peixe selecionado apareça na câmera secundária (RF);

Requisitos

- Aquário Virtual:
 - remover a câmera secundária caso nenhum peixe estiver selecionado (RF);
 - possuir uma área de texto para informar os comportamentos enviados pelo Reasoner (RF);
 - permitir que somente os comportamentos do peixe selecionado apareçam na área de texto (RF);

Requisitos

- Aquário Virtual:
 - limpar a área de texto caso nenhum peixe estiver selecionado (RF);
 - garantir que nenhum peixe saia dos limites do aquário (RF);
 - permitir a alteração de propriedades dos peixes, do aquário e do mundo virtual (RF);
 - permitir a inclusão e remoção de peixes (RF);

Requisitos

- Aquário Virtual:
 - excluir as presas devoradas do aquário e da árvore de peças (RF);
 - permitir a procriação de presas e predadores (RF);
 - excluir predadores que não se alimentem até determinado tempo (RF);

Requisitos

- Aquário Virtual:
 - aumentar a população de plâncton caso a população de sardinhas seja baixa e diminuir caso seja alta. Também deve refletir na cor da água do aquário, caso haja muito plâncton a água deve ficar esverdeada, caso contrário azulada (RF);
 - bloquear a procriação de presas caso a população de plâncton for muito baixa (RF);

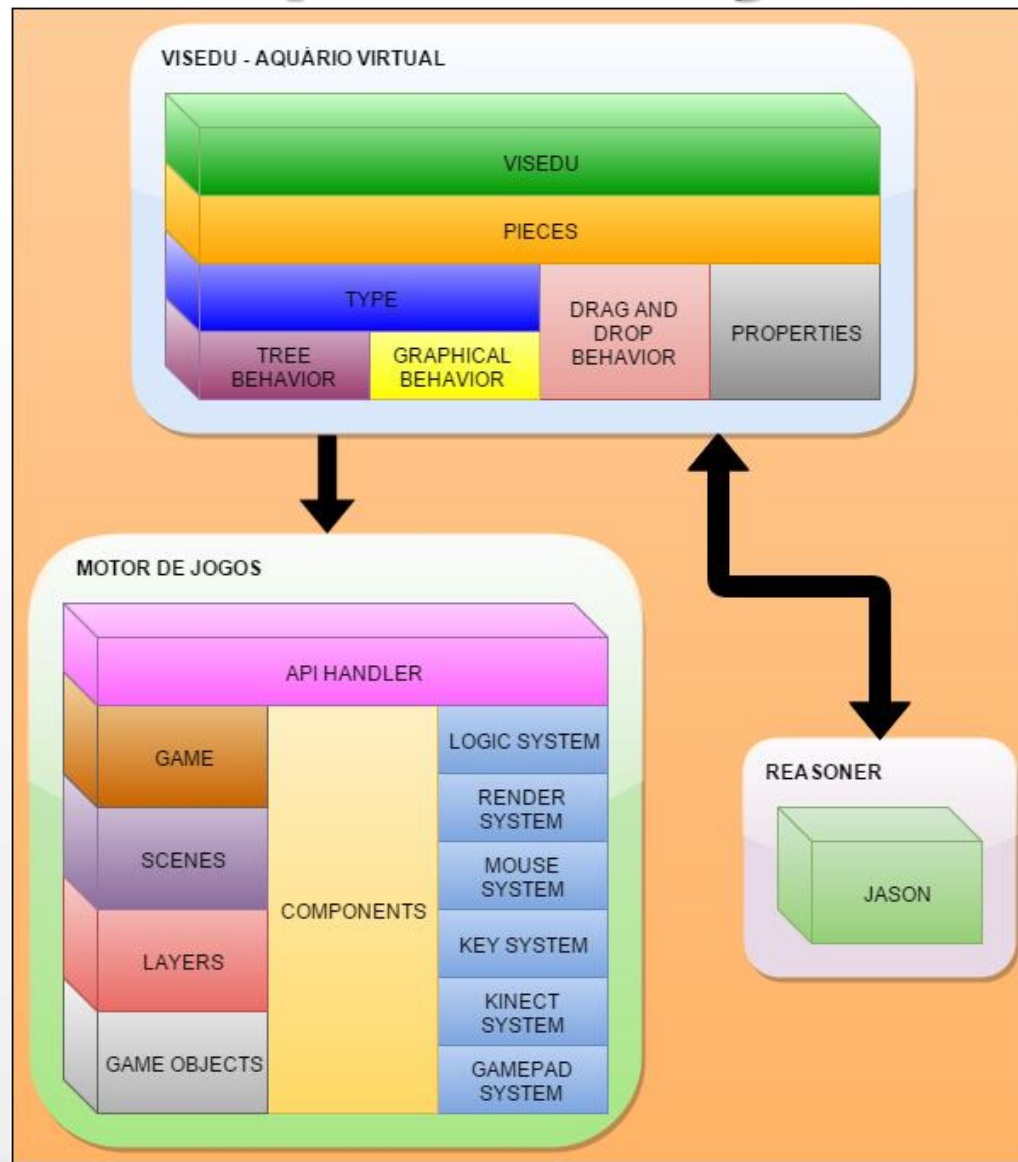
Requisitos

- Aquário Virtual:
 - trabalhar com espaço tridimensional (RNF);
 - utilizar o Reasoner para processar o raciocínio (RNF);
 - ser compatível com os mesmos navegadores que os módulos desenvolvidos por Feltrin (2014) e Koehler (2015) (RNF);
 - ser implementado com as tecnologias HTML5, Javascript e CSS (RNF);
 - utilizar a biblioteca gráfica ThreeJS (RNF).

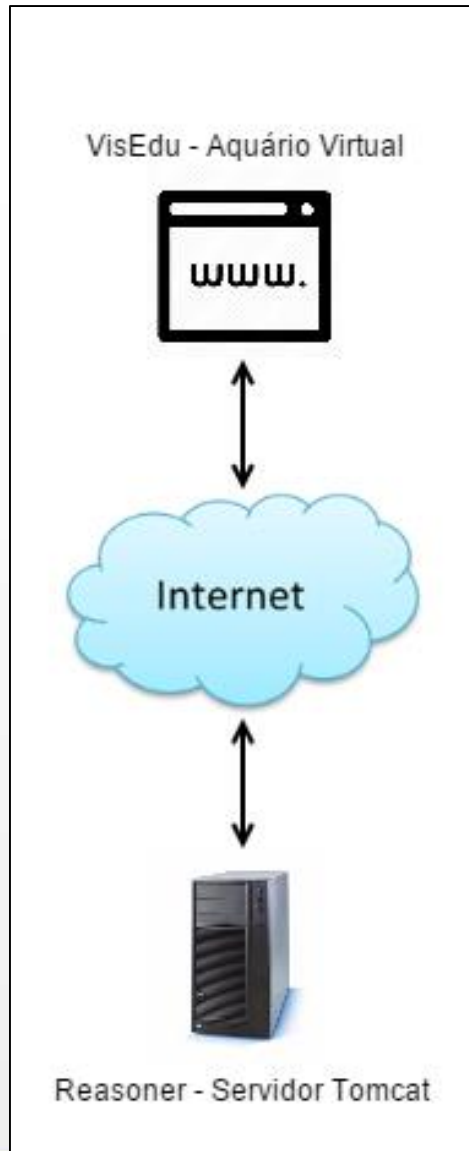
Requisitos

- Reasoner:
 - permitir a comunicação de vários agentes e de mais de um tipo com o interpretador Jason (RF);
 - utilizar o modelo BDI (RNF);
 - utilizar o interpretador Jason para gerenciar o modelo BDI (RNF);
 - ser desenvolvido com a linguagem de programação Java (RNF).

Especificação

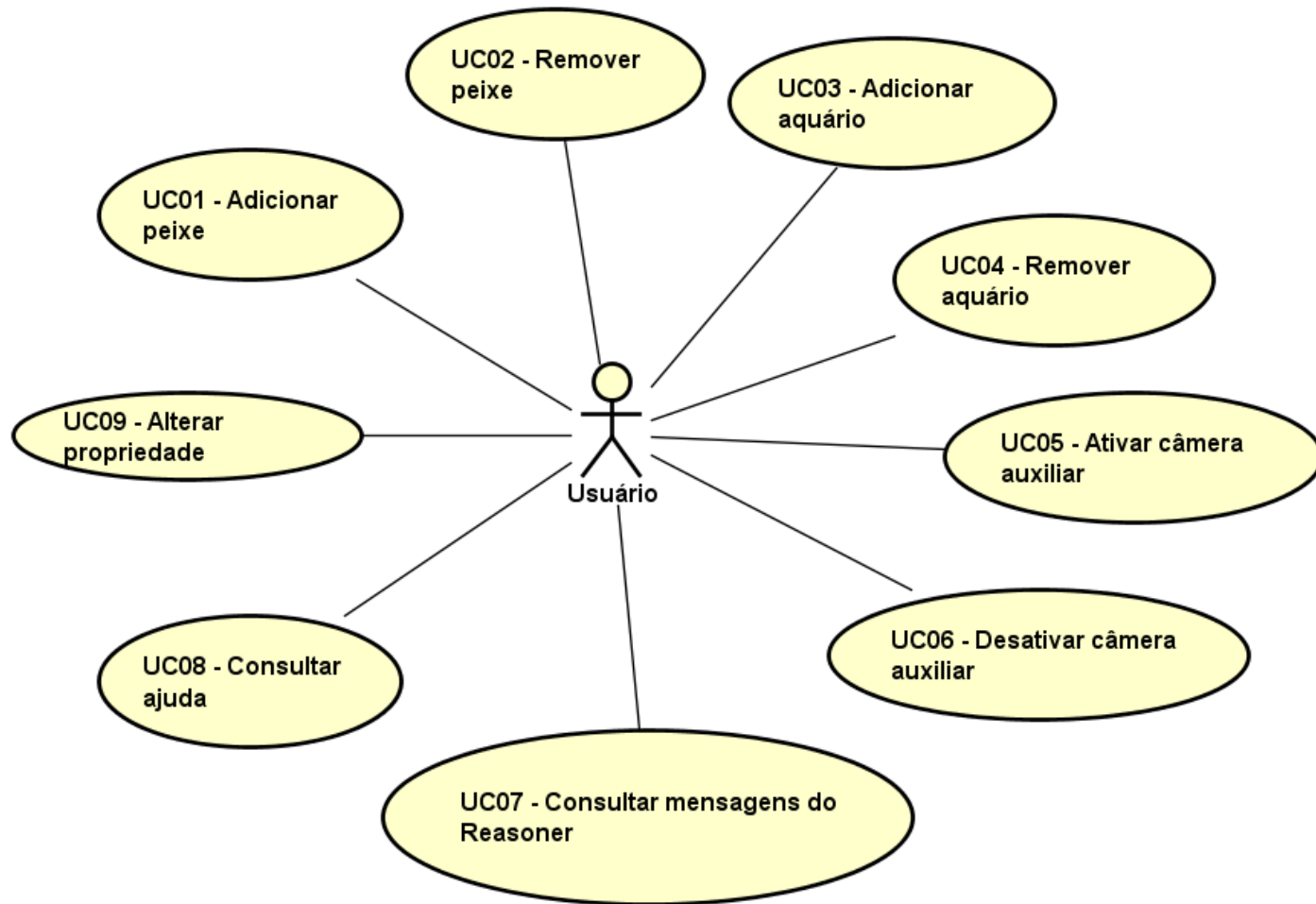


Especificação

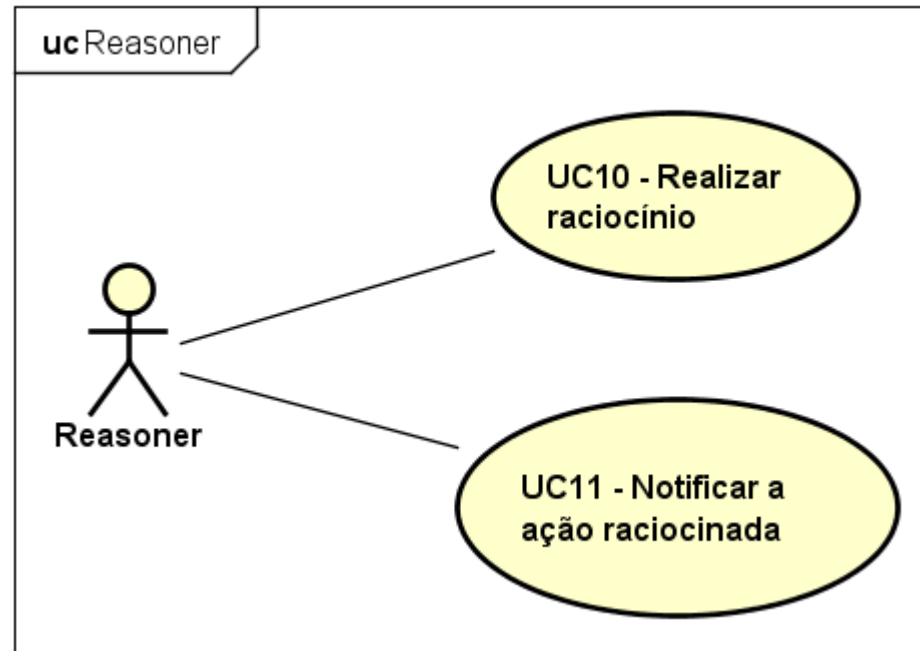


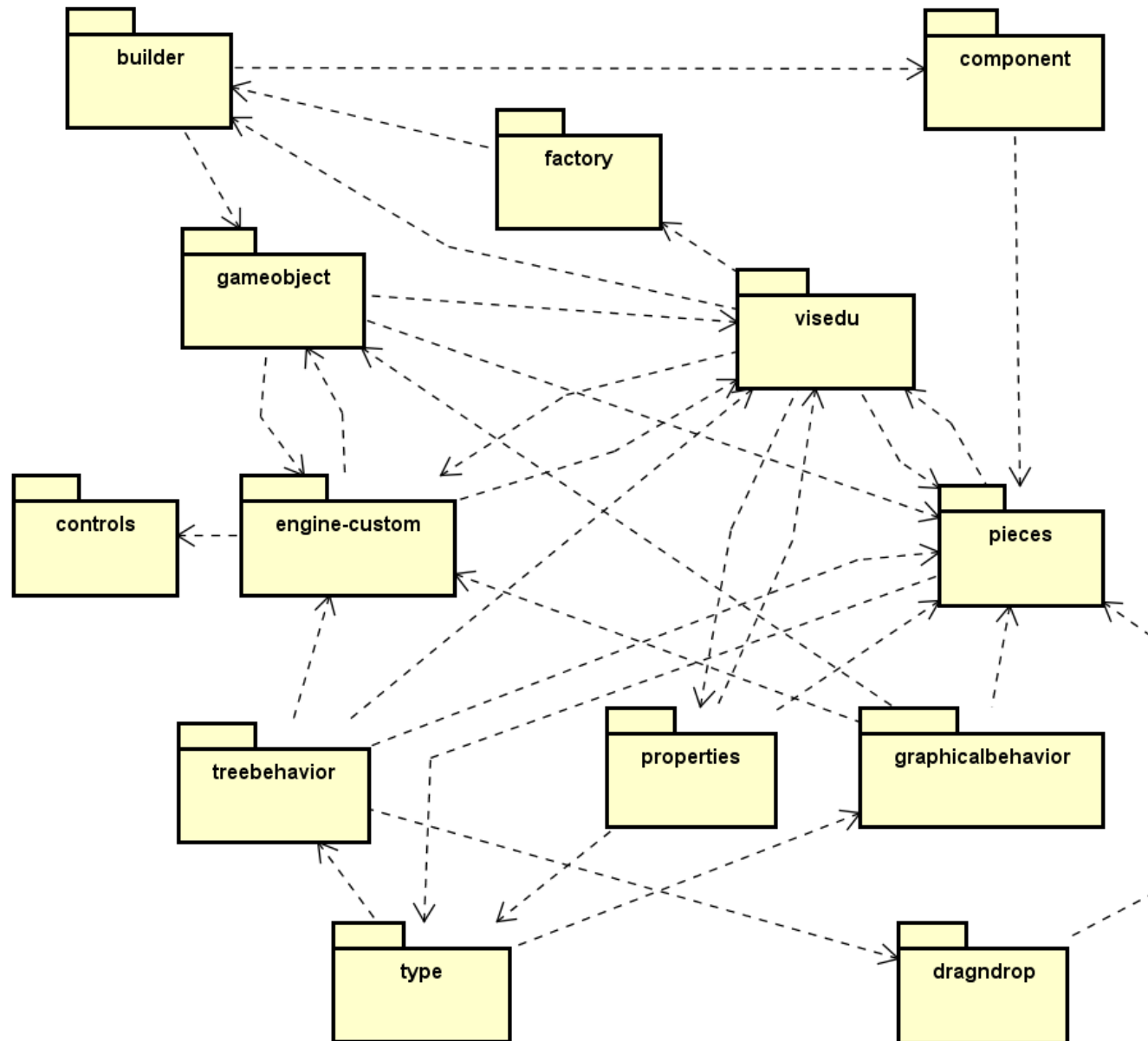
Especificação

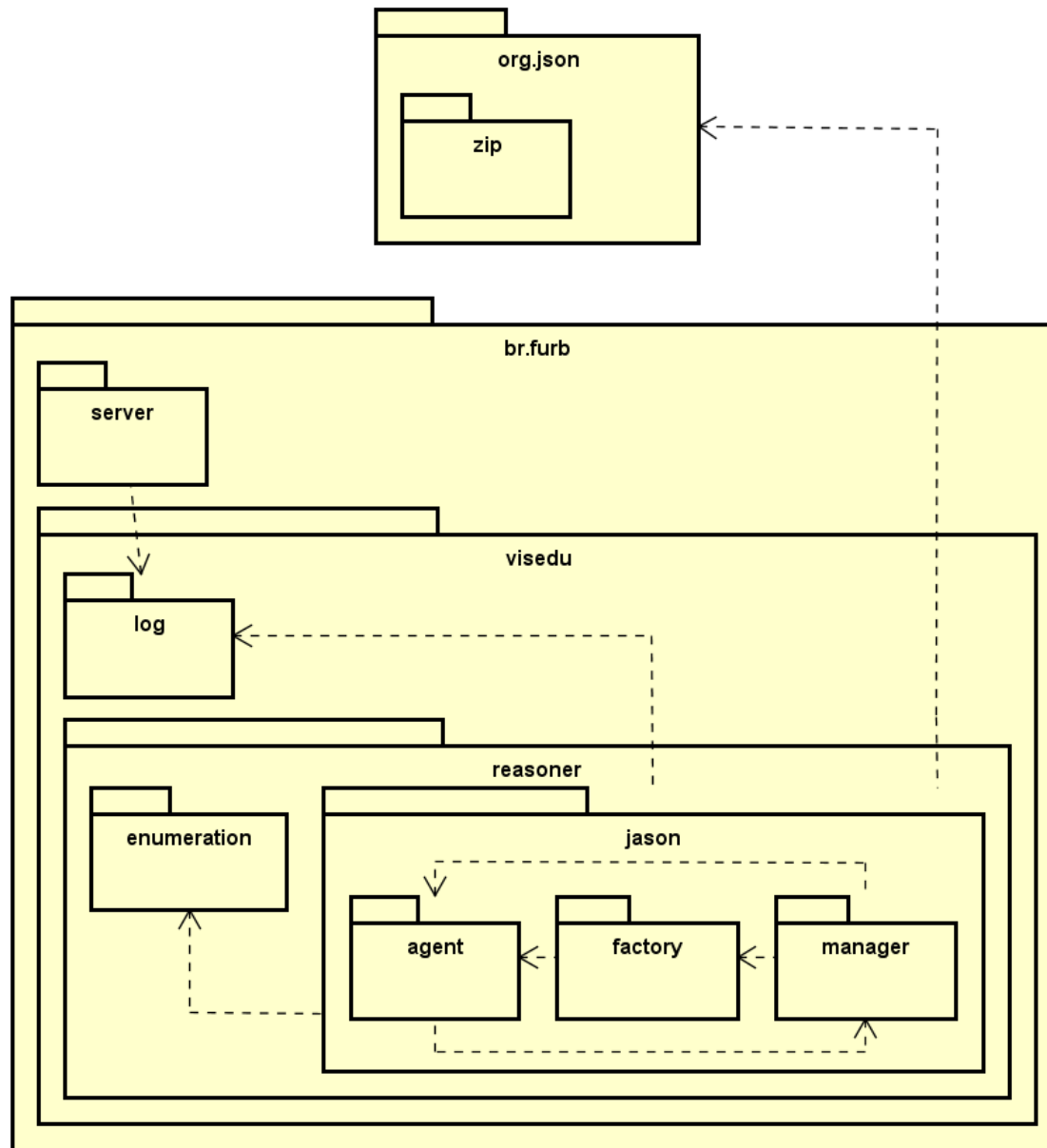
uc VisEdu - Aquário Virtual



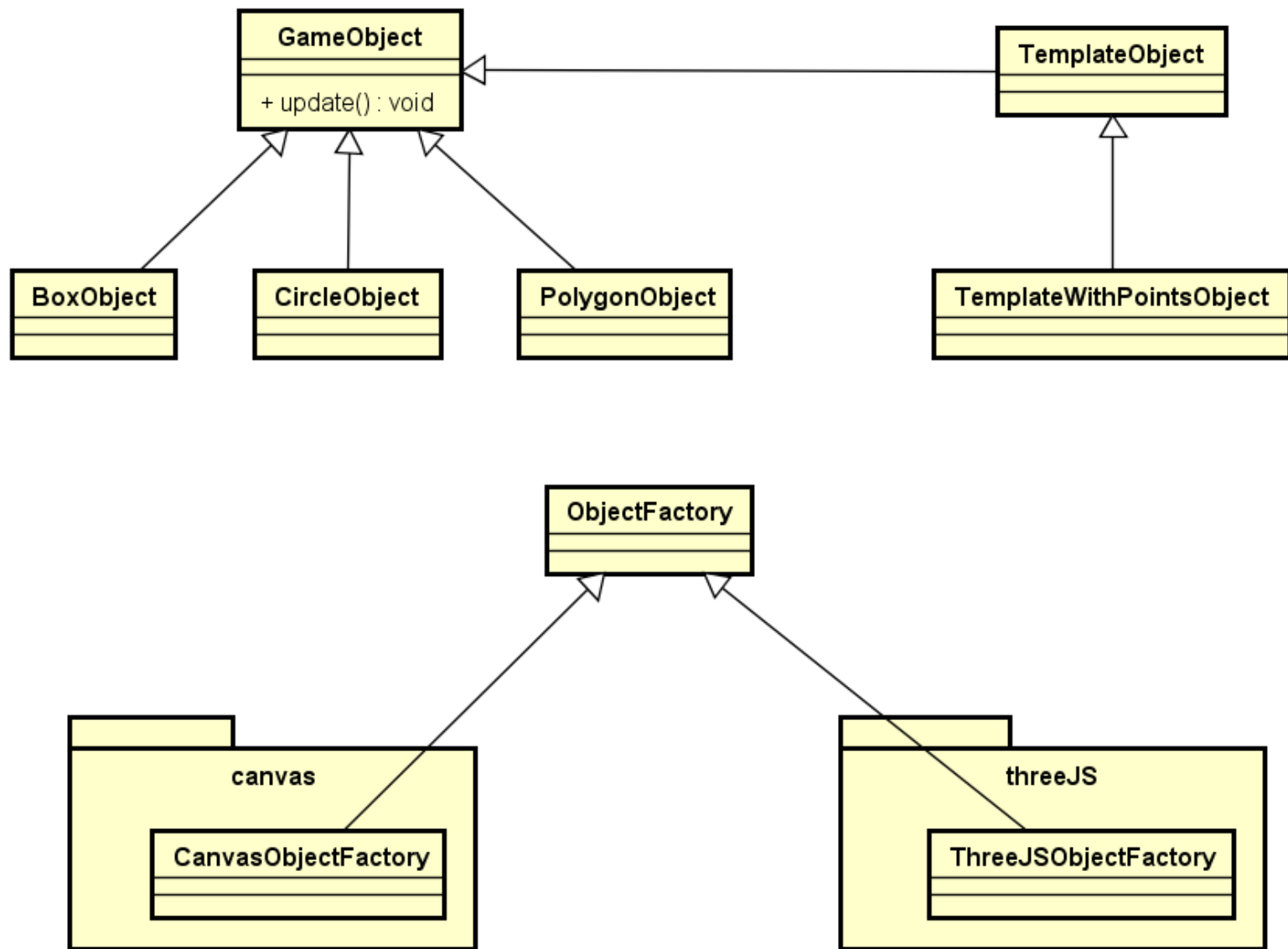
Especificação







pkggameobject



Implementação

- implementação do servlet do Reasoner:

```
@WebServlet("/jason")
public class ReasonerJasonServlet extends WebSocketServlet {

    private static final long serialVersionUID = 1L;

    protected StreamInbound createWebSocketInbound(String
subProtocol, HttpServletRequest request) {
        return new ReasonerJasonWebSocket();
    }
}
```

Implementação

- implementação do WebSocket no Reasoner:

```
public class ReasonerJasonWebSocket extends MessageInbound {  
  
    @Override  
    protected void onOpen(WsOutbound outbound) {  
        Log.info("onOpen: websocket is open");  
        MessageManager.getInstance().setWebSocket(this);  
    }  
  
    @Override  
    protected void onBinaryMessage(ByteBuffer bb) throws IOException {  
        throw new IOException("Method is not implemented!");  
    }  
}
```

Implementação

- implementação do Websocket no Reasoner (continuação):

```
@Override
protected void onTextMessage(CharBuffer message) throws IOException {
    Log.info("onTextMessage: " + message);
    MessageManager.getInstance().manage(message.toString());
}

public void sendMessage(String message) {
    Log.info("sendMessage: " + message);

    try {

getWsOutbound().writeTextMessage(CharBuffer.wrap(message));
    } catch (IOException e) {
        e.printStackTrace();
    }
}

}
```

Implementação

- configuração do agente:

```
public void configureAgent() {  
    try {  
        Agent jasonAgent = new Agent();  
        new TransitionSystem(jasonAgent, null, null, this);  
        jasonAgent.initAg(getAslFilePath());  
        showInfo(String.format("Agent \"%s\" using static  
mind @ %s", getAgName(), getAslFilePath()));  
    } catch (JasonException e) {  
        e.printStackTrace();  
        showError("Error initializing agent");  
    }  
}
```

Implementação

- método que informa a ação determinada:

```
@Override
public void act(ActionExec action, List<ActionExec> feedback) {
    showInfo("Agent " + getAgName() + ": doing: " +
action.getActionTerm());
    MessageManager.getInstance().sendMessage(
action.getActionTerm().toString() );
    action.setResult(true);
    feedback.add(action);
}
```

- execução do raciocínio:

```
public void run() {
    showInfo("Reasoning cycle...");
    getTS().reasoningCycle();
}
```

Implementação

- mente da sardinha:

```
/* Initial beliefs and rules */  
/* Crenças e regras iniciais */
```

```
predator("Tubarão").
```

```
/* Initial goals */  
/* Objetivos iniciais */  
+!explore.
```

```
/* Initial plans */  
/* Planos iniciais */
```

```
+onPercept(Perceiver, Perceived, PerceivedType) : predator(PerceivedType)  
    <- flee(Perceiver, Perceived).
```

```
+onPercept(Perceiver, Perceived, PerceivedType) : not predator(PerceivedType)  
    <- explore(Perceiver).
```

```
+onCollide(Perceiver, Perceived, PerceivedType)  
    <- explore(Perceiver).
```


Implementação

- mente do tubarão:

```
/* Initial beliefs and rules */  
/* Crenças e regras iniciais */
```

```
prey("Sardinha").
```

```
/* Initial goals */  
/* Objetivos iniciais */  
+!explore.
```

```
/* Initial plans */  
/* Planos iniciais */
```

```
+onPercept(Perceiver, Perceived, PerceivedType) : prey(PerceivedType)  
    <- pursue(Perceiver, Perceived).
```

```
+onPercept(Perceiver, Perceived, PerceivedType) : not prey(PerceivedType)  
    <- explore(Perceiver).
```

```
+onCollide(Perceiver, Perceived, PerceivedType) : prey(PerceivedType)  
    <- eat(Perceiver, Perceived).
```

```
+onCollide(Perceiver, Perceived, PerceivedType) : not prey(PerceivedType)  
    <- explore(Perceiver).
```

Implementação

- implementação do Websocket no Aquário Virtual:

```
this.createWebSocket = function(uri) {  
    var visEdu = this;  
    if ('WebSocket' in window || 'MozWebSocket' in window) {  
        this.webSocket = new WebSocket(uri);  
        this.timeOfInstantiation = Date.now();  
    } else {  
        alert("Browser não suporta Weboscket");  
        return this;  
    }  
    this.webSocket.onmessage = function(evt) {  
        visEdu.onMessage(evt)  
    };  
    this.webSocket.onopen = function(evt) {  
        visEdu.onOpen(evt)  
    };  
    this.webSocket.onclose = function(evt) {  
        visEdu.onClose(evt)  
    };  
    this.webSocket.onerror = function(evt) {  
        visEdu.onError(evt)  
    };  
}
```

Implementação

- implementação da função onRender:

```
ThreeJSCustomHandler.prototype.onRender = function() {  
    this.controls.update();  
    VisEdu.stats.update();  
  
    // Itera sobre os objetos presentes no mapa  
    $.each(ThreeJSCustomHandler.prototype.objectsMap, function(index, value) {  
        var threeObject = value.threeObject;  
        if(threeObject) {  
            if(threeObject.name != "") {  
                value.update(ThreeJSCustomHandler.prototype.objectsMap);  
            }  
        }  
    })  
}
```

Implementação

- implementação da função update da classe FishObject:

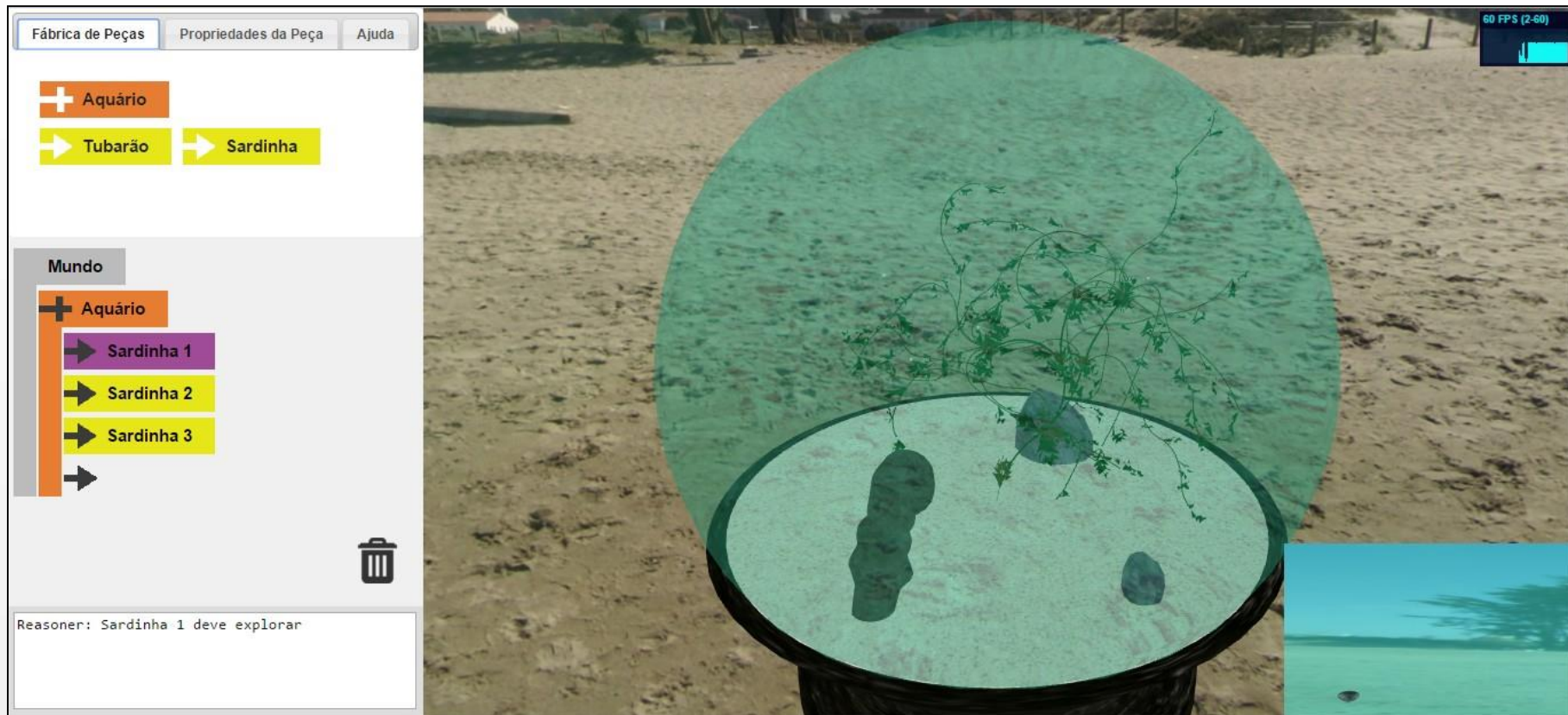
```
FishObject.prototype.update = function(objectsMap) {  
    if(this.isReady()) {  
        if(this.vision == null) {  
            this.vision = this.threeObject.children[0];  
            this.camera = this.threeObject.children[2];  
        }  
  
        if(Game.apiHandler.properties['speedMultiplier'] != 0) {  
            if(this.frameCount >= 2) {  
                if(Game.apiHandler.percept) {  
                    this.percept(objectsMap);  
                }  
                this.frameCount = 0;  
            }  
            this.move();  
        }  
    }  
    this.frameCount++;  
}
```

Implementação

- detecção de colisão via bounding box:

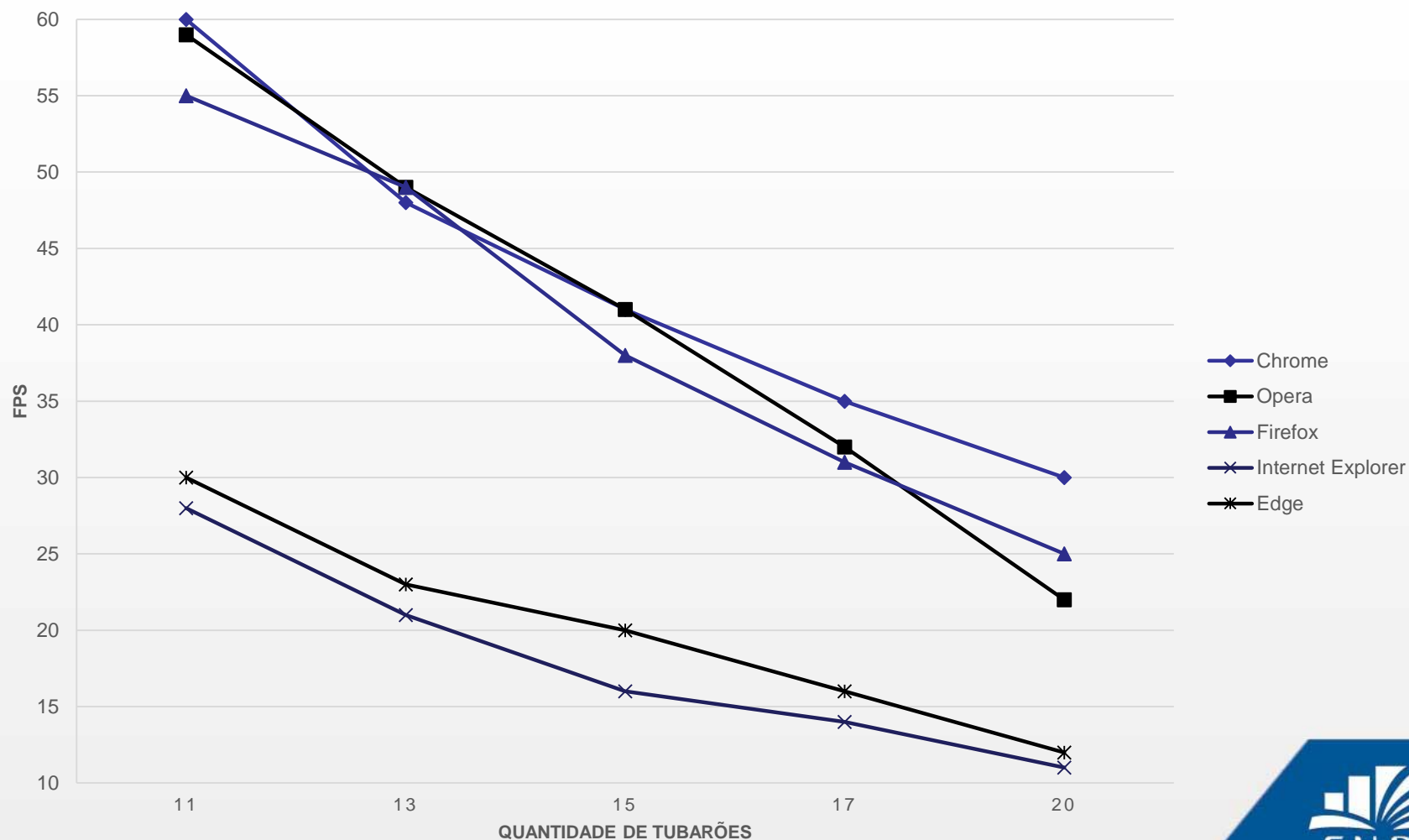
```
FishObject.prototype.detectCollisionBBox = function(object1, object2) {  
    if(object1 && object2) {  
        var object1BBox = new THREE.Box3().setFromObject(object1);  
        var object2BBox = new THREE.Box3().setFromObject(object2);  
        var collision = object1BBox.isIntersectionBox(object2BBox);  
        if(collision) {  
            return true;  
        }  
    }  
    return false;  
}
```

Operacionalidade da Implementação



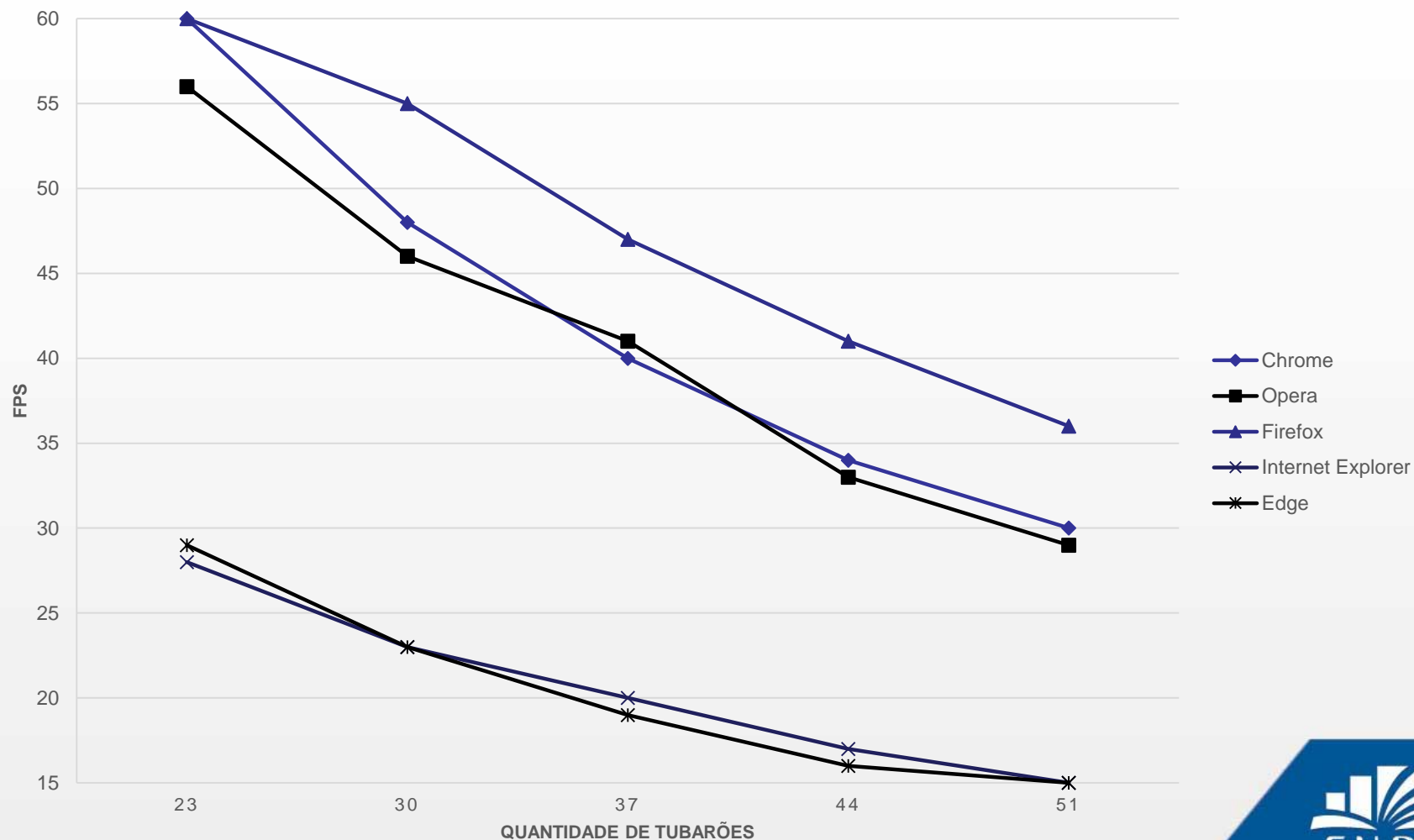
Resultados e Discussões

COMPARAÇÃO ENTRE NAVEGADORES EM RELAÇÃO AO FPS E A QUANTIDADE DE TUBARÕES COM PERCEÇÃO ATIVADA



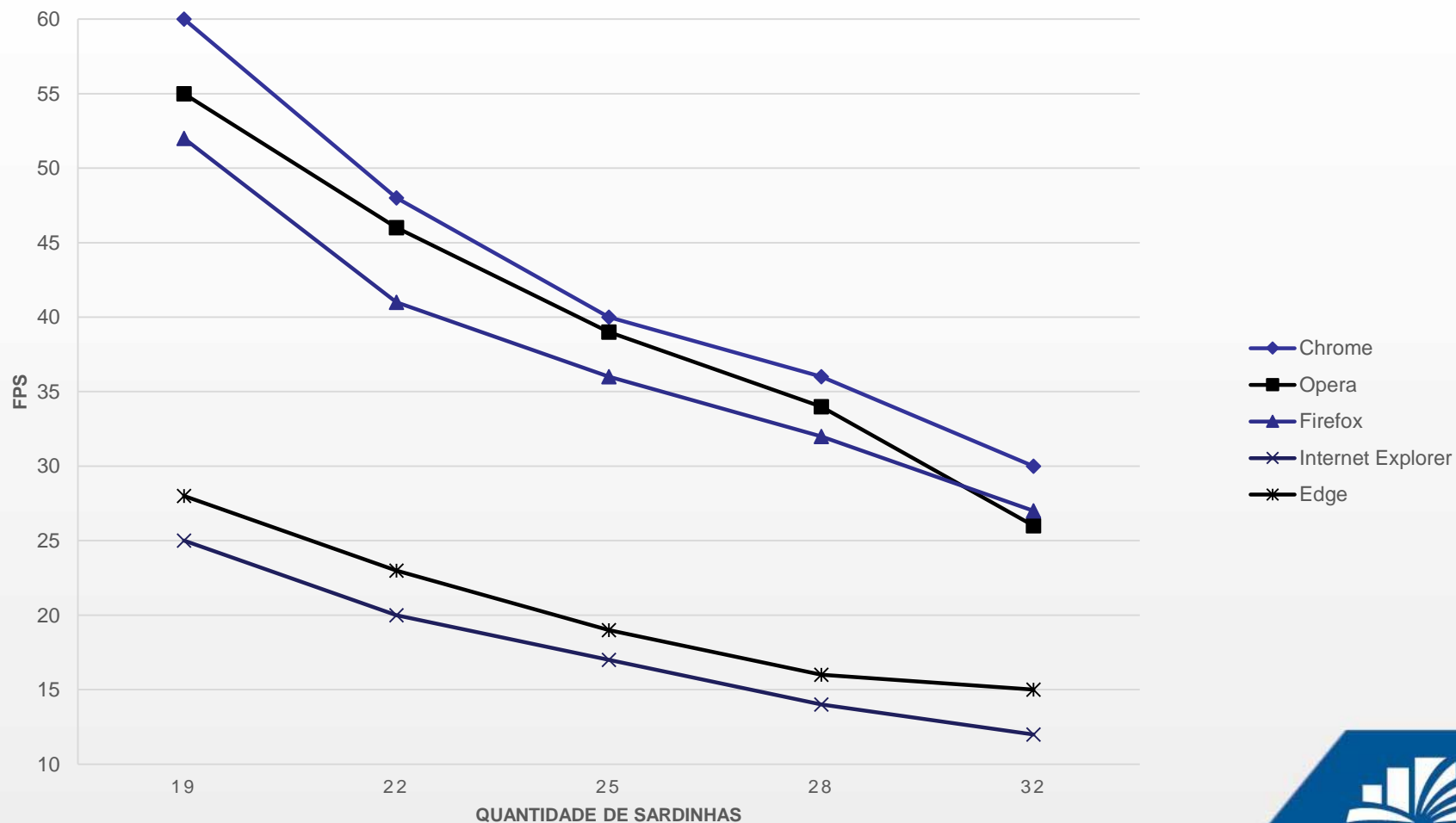
Resultados e Discussões

COMPARAÇÃO ENTRE NAVEGADORES EM RELAÇÃO AO FPS E A QUANTIDADE DE TUBARÕES COM PERCEPÇÃO DESATIVADA



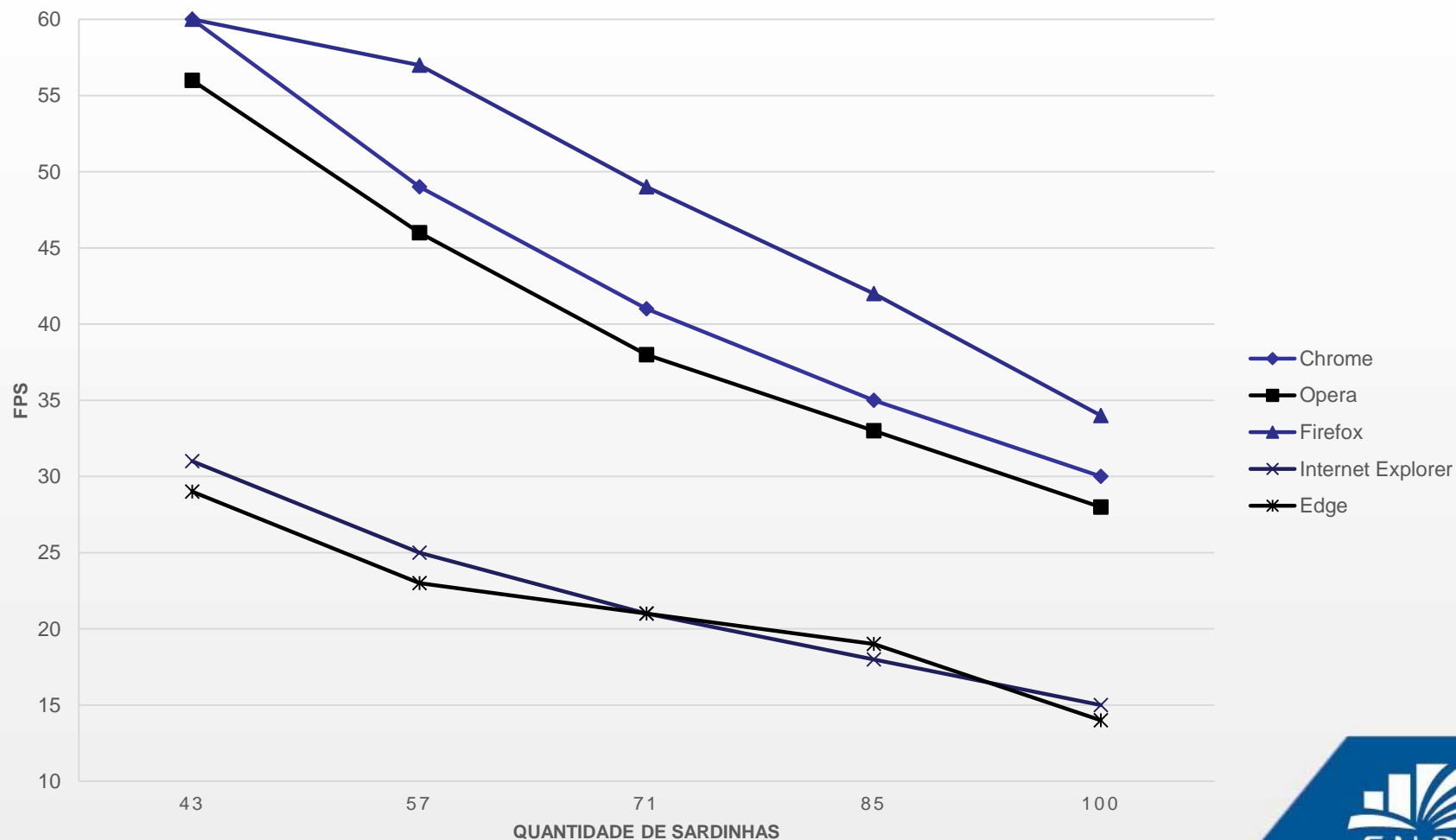
Resultados e Discussões

COMPARAÇÃO ENTRE NAVEGADORES EM RELAÇÃO AO FPS E A QUANTIDADE DE SARDINHAS COM PERCEPÇÃO ATIVADA



Resultados e Discussões

COMPARAÇÃO ENTRE NAVEGADORES EM RELAÇÃO AO FPS E A QUANTIDADE DE SARDINHAS COM PERCEPÇÃO DESATIVADA



Resultados e Discussões

quantidade	Memória consumida (MB) tubarões	Memória consumida (MB) sardinhas
0	23.1	23.1
2	26.9	24.6
4	30.0	26.2
6	33.0	27.3
8	35.9	28.3
10	38.9	29.4

$$Mc = Qt \times 1.58 + 23.1$$

$$Mc = Qs \times 0.63 + 23.1$$

$$Mc = Qt \times 1.58 + Qs \times 0.63 + 23.1$$

Resultados e Discussões

- tempo para estabelecer comunicação:

navegador	Opera	Internet Explorer	Chrome	Edge	Firefox
tempo (s)	0.171	0.212	0.239	0.288	0.291

- tempo médio de raciocínio:

quantidade de peixes	Opera	Firefox	Chrome	Internet Explorer	Edge
3	0.004	0.004	0.004	0.009	0.007
6	0.004	0.006	0.004	0.017	0.022
9	0.004	0.015	0.003	0.041	0.057
12	0.009	0.033	0.008	0.095	10.24
15	0.1	0.056	0.114	0.533	19.677

Resultados e Discussões

- comparação com os trabalhos correlatos:

aspectos/trabalhos	MASSIVE	Fish School and Obstacles	STEVE	VISEDU – Aquário Virtual
gera animação comportamental	X	X	X	X
possui ambiente tridimensional	X		X	X
desenvolvido para a web		X		X
possui fins educativos			X	X
relacionado à biologia		X		X
permite interação com o ambiente	X	X	X	X
possui tipos variados de mente	X	X		X
desacoplado do modelo de IA			X	X
gera comportamento imprevisível	X	X		X

Conclusões e Sugestões

- objetivos propostos X contemplados;
- ferramentas Aquário Virtual:
 - HTML5 em conjunto com Javascript;
 - ThreeJS;
- ferramentas Reasoner;
 - Interpretador Jason;
 - AgentSpeak;
 - Java;
- principais contribuições;
- principais limitações.

Conclusões e Sugestões

- sugestões:
 - criar modelos mentais mais elaborados;
 - utilizar outras técnicas de IA para a interpretação do raciocínio no Reasoner;
 - integrar o Reasoner com outros interpretadores do modelo BDI;
 - utilizar outras técnicas para detectar colisão;
 - utilizar outras bibliotecas gráficas;

Conclusões e Sugestões

- sugestões:
 - utilizar ferramentas voltadas para a web que permitam o desenvolvimento de forma concorrente;
 - implementar animações mais reais;
 - adicionar novas formas de vida, novas cadeias alimentares;
 - implementar ou integrar com um modelo BDI em Javascript, caso exista;

Conclusões e Sugestões

- sugestões:
 - incluir um aquário de água doce e implementar seu ecossistema;
 - implementar comportamentos cooperativos e competitivos entre os seres do aquário;
 - trazer aspectos de jogos para o aquário (gamificação);
 - implementar novas formas de interação com o aquário.

Demonstração da aplicação