



## WiiMote para Navegação na Web

---

INSTITUTO POLITÉCNICO DE BEJA  
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO  
LICENCIATURA EM ENGENHARIA INFORMÁTICA

# **WiiMote para Navegação na Web**

Relatório de projecto de fim de curso apresentado na Escola Superior de Tecnologia e Gestão do  
Instituto Politécnico de Beja como parte dos requisitos necessários à obtenção do grau de  
Licenciado em Engenharia Informática

**Orientadores do Projecto:**

Luís Bruno  
João Paulo Barros



# Sumário

Serve o presente trabalho para se efectuar uma abordagem à possibilidade de se substituir o periférico conhecido como rato por um *Wii Mote*, com vista a facilitar a navegação na Web em situações de aulas e/ou apresentações.

Através da aplicação desenvolvida é possível usar uma variedade de funções orientadas para o *browser* Mozilla Firefox [1] e definir quais os botões que efectuam essas funções, sendo também permitido alterar a velocidade e a orientação da movimentação do cursor do rato.

Comparativamente ao rato, e após testes efectuados, concluímos tratar-se de um sistema com precisão e eficiência, necessitando apenas de alguma prática.



# Dedicatória

Dedico este trabalho aos meus pais, que sempre lutaram por mim, se sacrificaram e fizeram tudo para que eu pudesse concluir este curso, sem eles não teria conseguido.

Boris Júnior

Dedico este projecto aos meus pais por todo o apoio que me deram, pelos sacrifícios que ambos fizeram para que eu pudesse ter condições para estudar e pelos valores que me transmitiram que fizeram com que nunca desistisse dos meus objectivos nas alturas mais difíceis e me deram sempre força para continuar.

Jorge Pereira



# Índice

Sumário .....	3
Dedicatória .....	5
Introdução .....	9
1. Estado da Arte .....	11
1.1 Software de suporte à utilização do Wii Remote em computadores pessoais .....	11
1.2 Análise de bibliotecas existentes .....	12
1.3 Critérios de escolha das bibliotecas .....	13
2. Análise e Especificação de Requisitos .....	15
2.1 Descrição do <i>Wii Remote</i> .....	15
2.2 Análise à movimentação do ponteiro do rato .....	16
2.3 Utilização da aceleração na movimentação do cursor do rato .....	17
2.3.1 Movimentação vertical .....	18
2.3.2 Movimentação horizontal .....	18
2.3.3 Repouso .....	20
2.4 Requisitos Funcionais .....	20
2.5 Requisitos não funcionais .....	22
3 Desenho .....	23
3.1 Ideias Iniciais .....	23
3.2 Movimento do Comando .....	23
3.3 Configuração Actual .....	24
3.4 Desenhos Alternativos .....	24
3.5 Diagrama de Classes .....	24
3.6 Primeira Fase de Testes com Utilizadores .....	26
4. Implementação .....	27
4.1 Diagramas de Sequência .....	27
4.2 Ligação ao comando .....	29
4.3 Leitura da Configuração dos Botões .....	31



4.4	Detecção de eventos .....	33
4.5	Movimento do cursor .....	35
5	Testes com utilizadores .....	39
5.1	Descrição do ambiente em que se realizaram as tarefas.....	39
5.2	Características dos participantes .....	39
5.3	Metodologia .....	39
5.4	Tarefas.....	40
5.5	Testes e medidas.....	41
5.6	Análise dos resultados.....	44
5.6.1	Análise à fase de ambientação.....	44
5.6.2	Análise à fase de testes.....	45
	Conclusão e trabalho futuro .....	49
	Apêndice(s) .....	51
	Apêndice 1 – Diagrama de Casos de Uso .....	51
	Apêndice 2 – Documento de especificação.....	51
	Apêndice 3 – Guião de Tarefas .....	55
	Apêndice 4 – Resultados dos testes de utilizadores .....	57
	Apêndice 5 – Inquérito aos utilizadores .....	58
	Bibliografia .....	61
	Agradecimentos.....	63

# Introdução

A utilização de interfaces com o utilizador alternativas ou complementares aos ratos e teclados usuais assume cada vez maior importância. Existe actualmente no mercado um comando com maiores capacidades do que um simples rato por um preço semelhante ao preço médio de um rato: o comando *Wii Remote* para a consola *Wii* da Nintendo [2]. Este projecto pretende aproveitar este dispositivo na utilização de apresentações e navegação na Internet com utilização de um browser. Tal permitirá uma interface especialmente útil e atractiva como alternativa à utilização típica de um rato e de um apontador laser em apresentações, especialmente em espaços de grandes dimensões, interiores ou exteriores. Para tal, pretende-se tirar partido das potencialidades específicas do comando, nomeadamente a possibilidade de utilizar uma maior quantidade e variedade de botões e de detectar movimentos.

A concepção deste projecto teve como base a linguagem de programação JAVA<sup>TM</sup> REF e o IDE Eclipse SDK [3]. Foram também utilizadas algumas bibliotecas disponíveis na internet, nomeadamente o *WiiRemoteJ* [4] e o *BlueCove* [5], que faz a gestão da ligação *Bluetooth* entre o computador e o comando *Wii Remote*.

No capítulo 1 será feito um estudo e uma análise do actual estado da arte, verificando o actual nível de software de suporte à utilização do *Wii Remote* em computadores pessoais. Também será feita uma análise das bibliotecas existentes através de tabelas de comparação. No final serão esclarecidos os critérios usados na escolha das bibliotecas.

No capítulo 2 será efectuada a análise e especificação de requisitos. Haverá, inicialmente, uma descrição do comando e das suas potencialidades, existirá a exibição de alguns gráficos de modo a serem contextualizados os requisitos (funcionais e não-funcionais) e finalmente, será feita uma especificação individual de cada função, seja a nível do rato, teclado ou do comando.

No capítulo 3 serão abordadas questões a nível do desenho do projecto, nomeadamente decisões tomadas sobre como os movimentos são efectuados, sobre ideias iniciais e o porquê de ter havido alterações, justificando as decisões a nível da configuração original dos botões ao mesmo tempo que será indicado o que essa configuração actualmente permite. Serão discutidos desenhos alternativos e indicado o porquê de terem ou não sido usados. Haverá uma apresentação do diagrama de classes, de modo a haver um melhor entendimento e percepção das classes e dos objectos da aplicação. No final do capítulo falar-se-á da primeira fase de testes de utilizadores, a qual levou a que algumas decisões fossem tomadas.

No capítulo 4, a fase de implementação, será explicado detalhadamente o funcionamento do programa. Serão analisados alguns eventos através de diagramas de sequências. Referenciam-se

alguns aspectos da programação que levaram a que tivessem sido tomadas algumas decisões ao nível de desenho.

No capítulo 5 serão abordados os testes efectuados com utilizadores, especificamente a fase onde já foi usada uma versão final do programa e onde foram efectuadas algumas tarefas com grupos heterogéneos de utilizadores de modo a se verificarem os erros dados e haver uma medição do tempo levado. No final, os resultados são analisados.

# 1. Estado da Arte

Neste capítulo será feito um estudo e uma análise do actual estado da arte, verificando o actual nível de software de suporte à utilização do *Wii Remote* em computadores pessoais. Também será feita uma análise das bibliotecas existentes através de tabelas de comparação. No final serão esclarecidos os critérios usados na escolha das bibliotecas.

## 1.1 Software de suporte à utilização do Wii Remote em computadores pessoais

Actualmente existem várias bibliotecas que permitem que uma aplicação do computador controle o Wii Remote. Estas bibliotecas denominam-se controladores de dispositivos (*drivers*) que estabelecem uma ligação Bluetooth com o comando utilizando o Bluetooth HID (Human Interface Device).

## 1.2 Análise de bibliotecas existentes

Após pesquisa das bibliotecas existentes para dar suporte ao sistema que se pretende implementar foram consideradas as bibliotecas (*API's*) MoteJ 0.9 [6], WiiuseJ 0.12b [7], WiiRemoteJ v1.6, Wiiuse 0.12 [8] e WiiYourself v1.14 beta [9]. Na tabela 1 podemos observar a comparação entre as bibliotecas estudadas:

	Motej	Wiiusej	WiiRemoteJ	Wiiuse	Wiiyourself
<b>Linguagem</b>	Java	Java	Java	C	C++
<b>Descoberta e Ligação ao Wiiremote</b>	Sim	Sim	Sim	Não	Não
<b>Windows</b>	Sim	Sim	Sim	Sim	Sim
<b>Linux</b>	Sim	Sim	Sim	Sim	Não
<b>Macintosh</b>	Sim	Não	Sim	Não	Não
<b>Bluetooth API (Win/Linus/Mac)</b>	(Bluecove / JBlueZ [10] / Bluecove)	(Bluecove/ JBlueZ / -)	(Bluecove / JBlueZ / Bluecove)	N/A	N/A
<b>Stacks Suportados</b>	Widcomm	Widcomm	Widcomm	Todos [11]	Todos
<b>Código-fonte / Exemplos</b>	Sim	Sim	Sim	Sim	Sim
<b>Última Actividade</b>	27/06/2009	18/01/2009	27/02/2009	03/04/2008	24/07/2009

**Tabela 1 - Comparação entre as bibliotecas *Wii Remote* estudadas.**

Observando a tabela podemos verificar que as bibliotecas existentes se dividem entre as linguagens Java e C/C++.

As bibliotecas Java em conjunto com a biblioteca *Bluetooth* permitem a ligação entre o comando e o computador sem que seja necessário emparelhar o comando recorrendo ao software de gestão de dispositivos (*stack*) *Bluetooth*. Permitem desta forma que toda a gestão da ligação seja feita pelo próprio sistema.

Além da vantagem referida, são bibliotecas multi-plataforma, possibilitando a sua utilização em diferentes sistemas operativos.

Apresentam no entanto uma desvantagem: apenas funcionam correctamente com a *stack* Widcomm. Esta situação verifica-se devido à incompatibilidade das bibliotecas *Bluetooth* utilizadas na descoberta e ligação do comando com outros *stacks Bluetooth*, restringindo, assim, a sua utilização a receptores *Bluetooth* compatíveis com a *stack* Widcomm.

### 1.3 Critérios de escolha das bibliotecas

Todas as bibliotecas Java mostram actividade recente e todas fornecem exemplos da sua utilização. A biblioteca WiiRemoteJ fornece uma classe de testes denominada “WRLImpl.java” que exemplifica grande parte dos métodos da biblioteca, permitindo uma melhor compreensão da biblioteca, para implementação no sistema.

As bibliotecas C/C++ necessitam do *stack Bluetooth* para emparelhar o computador com o comando e apenas a Wiiuse permite a sua utilização em sistemas operativos diferentes, embora falte a sua implementação em sistemas operativos Macintosh.

Apresentam uma importante vantagem: ao não utilizar uma biblioteca *Bluetooth* não estão dependentes da sua compatibilidade com *stacks Bluetooth*, possibilitando a sua utilização com todos os *stacks* conhecidos. Esta situação pode no entanto dificultar a utilização do sistema devido ao processo de emparelhamento do comando com o computador.

A biblioteca Wiiuse é mais antiga que a biblioteca Wiiyourself mas parece estar mais completa, é multi-plataforma e apresenta exemplos mais fáceis de compreender e mais organizados. A Wiiyourself fornece um projecto que permite observar a captura dos eventos, mais completo, mas que está mais desorganizado e uma mistura de C e C++. Necessitando, pois, de ser refactorizado, o que implica trabalho adicional.

De acordo com os requisitos enumerados pretende-se que o sistema a implementar seja multi-plataforma, de fácil utilização e compatível com vários *stacks Bluetooth*.

Verifica-se que as bibliotecas Java são multi-plataforma e de fácil utilização pela forma como dispensam o emparelhamento manual do comando. Sendo uma das prioridades do sistema a facilidade de utilização, entende-se que é uma importante vantagem em relação às bibliotecas C/C++.

Optou-se então por uma biblioteca java em detrimento da compatibilidade com outras *stacks Bluetooth*. Esta opção implica que a *stack* Widcomm passe a ser um requisito para utilizar o sistema.

Entre as bibliotecas Java não se encontram grandes diferenças de funcionamento uma vez que todas têm funções para as várias formas de entrada e saída do comando. Em termos da sua utilização em sistemas verifica-se que entre elas apenas a Wiiusej não é compatível com Macintosh, sendo portanto uma desvantagem em relação às outras.

Entre a WiiRemoteJ e a Motej, a diferença consiste principalmente nos exemplos que são fornecidos que permitem uma melhor compreensão da biblioteca e implementação no sistema.

Conclui-se portanto que a biblioteca Java WiiRemoteJ é a mais adequada para utilizar no sistema.

O próximo capítulo inicia-se com uma descrição do comando e das suas potencialidades. Seguidamente, apresentam-se alguns gráficos de modo a serem contextualizados os requisitos (funcionais e não-funcionais) e finalmente, será feita uma especificação individual de cada função, seja a nível do rato, teclado ou do comando.

## 2. Análise e Especificação de Requisitos

Neste capítulo haverá uma descrição do comando e das suas potencialidades, existirá a exibição de alguns gráficos de modo a serem contextualizados os requisitos (funcionais e não-funcionais) e finalmente, será feita uma especificação individual de cada função, seja a nível do rato, teclado ou do comando.

### 2.1 Descrição do *Wii Remote*

O *Wii Remote* é um dispositivo similar a um típico comando de televisão, com 48 mm de comprimento, 36.2 mm de largura e 30.8 mm espessura [12], sendo composto por [13]:

- Acelerómetro que proporciona 3 dimensões de dados de aceleração (x, y, z).
- Câmara de infravermelhos.
- Doze botões.
- Motor de vibração.
- Quatro LEDS azuis.
- Altifalante.

A Figura 1 exemplifica a forma como o comando interpreta os dados relativamente à aceleração.

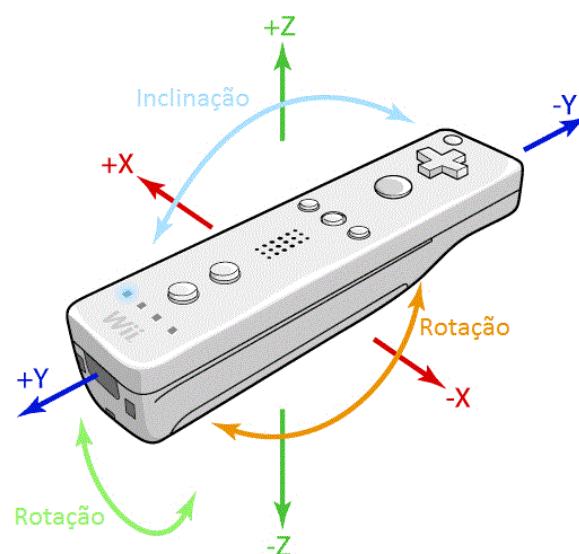


Figura 1 – Imagem exemplificativa de como o registo da aceleração é feita na aplicação.



## 2.2 Análise à movimentação do ponteiro do rato

Procurou-se movimentar o ponteiro do rato utilizando o *Wii Remote* da mesma forma que é feita a movimentação utilizando o rato, ou seja, efectuando apenas o movimento horizontal e vertical. Este movimento seria equivalente no plano (x, y) ao deslocamento ao longo do eixo X, no caso do movimento horizontal, e ao deslocamento ao longo do eixo Y, no caso do movimento vertical, como se pode observar na figura 2.

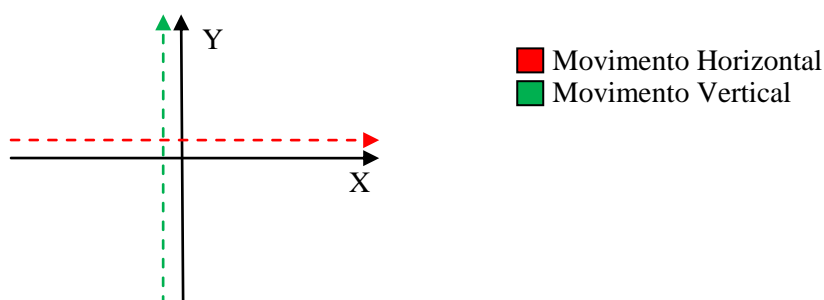


Figura 2 – Gráfico da movimentação do comando

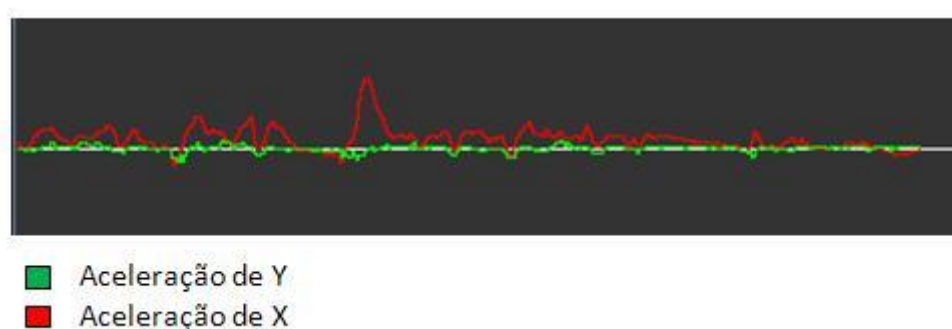
Para efectuar estas movimentações deveria ser suficiente movimentar o comando sem o rodar, à semelhança do que acontece durante a interacção do comando com a consola Wii quando este desempenha a função de ponteiro. No entanto, verificou-se ser necessário inclinar para cima ou para baixo a parte frontal do comando que contém o IR para o movimento vertical, ou ainda rodar o comando para o movimento horizontal. No entanto verificou-se que esse tipo de movimento é necessário, mesmo utilizando o adaptador *Wii Motion Plus*, a barra de sensores da consola [14].

Dado que esta barra na sua forma original não permite a comunicação com o computador, seja por *Bluetooth* ou USB, a única maneira de permitir a sua ligação é a sua modificação de forma artesanal ligando a esta uma ficha USB [15].

Entende-se que esta barra de sensores artesanal iria melhorar muito a utilização do comando como ponteiro mas que seria demasiado complexo fabricar a barra, uma vez que exige conhecimentos de electrónica. Além disso existem outras limitações como a interferência das luzes, solar ou artificial, no funcionamento da comunicação por infravermelhos. Tendo conhecimento que uma sala de aula tem vários pontos luminosos, prevêem-se alguns impedimentos.

Tentou-se uma solução de software, onde a movimentação do rato seria feita sem rodar o comando, como um ponteiro.

Podemos observar na Figura 3 os valores da aceleração para X e Y quando é feita a deslocação do comando da forma pretendida:



**Figura 3 – Gráfico do registo da deslocação horizontal**

Verifica-se a alteração do valor da aceleração de X, a coordenada pretendida, ainda que de forma inconstante. Constata-se também que o valor é demasiado reduzido para servir de coordenada X ao movimento do cursor do rato, sendo necessária a multiplicação deste por um factor.

Ao analisar esta solução viu-se que, de facto, o cursor se movimenta, mas de uma forma demasiado inconstante para que esta fosse considerada.

Conclui-se que as hipóteses consideradas para tentar movimentar o cursor do rato horizontalmente como um ponteiro, sem rodar o comando, não são viáveis. A nível de hardware devido à dificuldade em construir a barra e a nível de software demasiado inconstante.

Decidiu-se portanto abandonar a hipótese de movimentar o comando horizontalmente como um ponteiro, continuando esta a ser feita através da rotação do comando.

## **2.3 Utilização da aceleração na movimentação do cursor do rato**

A análise dos valores da aceleração foi feita recorrendo à classe de teste “WRLImpl.java” existente no projecto de teste “WiiRemoteJ v1.6” pertencente à biblioteca “WiiRemoteJ.jar”.

Esta classe de teste cria um gráfico onde é possível visualizar os valores da aceleração para as coordenadas dos eixos X, Y e Z.

As coordenadas do eixo X surgem a vermelho, as do Y a verde e as do Z a azul. É de notar que no gráfico, os valores positivos ficam abaixo da linha branca horizontal e os valores negativos acima da mesma.

### 2.3.1 Movimentação vertical

Na Figura 4 pode-se verificar que estando o comando numa posição de repouso e efectuando um movimento descendente verificam-se as seguintes alterações no gráfico: Z diminui, X mantém, Y aumenta.

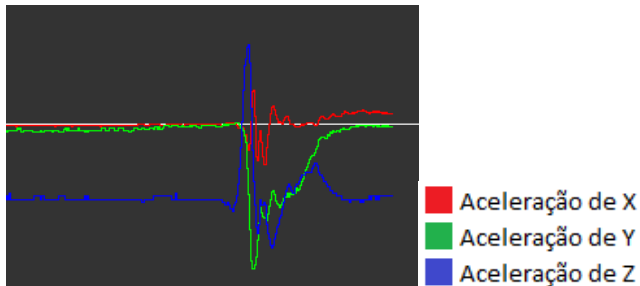


Figura 4 – Gráfico com o registo da movimentação vertical descendente do comando

Constata-se na Figura 5 que estando o comando numa posição de repouso e efectuando um movimento ascendente verificam-se as seguintes alterações no gráfico: Z aumenta, X mantém, Y aumenta.

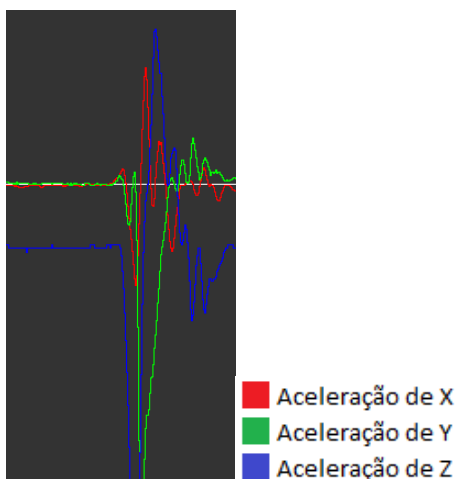
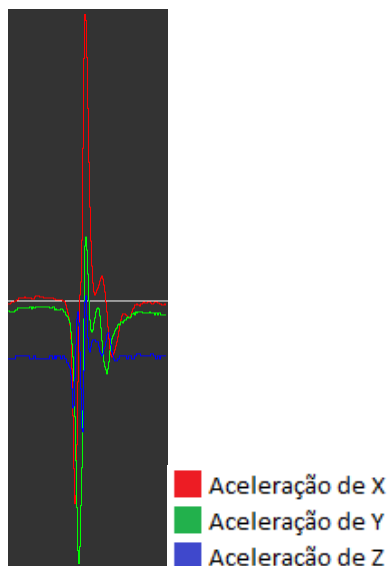


Figura 5 - Gráfico com o registo da movimentação vertical ascendente do comando

### 2.3.2 Movimentação horizontal

Observa-se na Figura 6 que estando o comando numa posição de repouso e efectuando um movimento para o lado esquerdo verificam-se as seguintes alterações no gráfico: Z mantém, X diminui, Y aumenta.

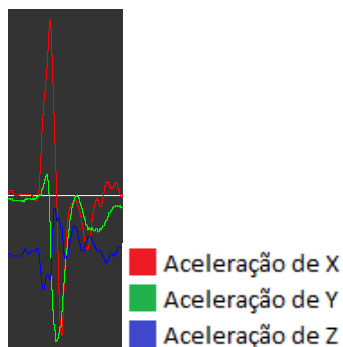
É possível verificar que primeiro há uma acentuação para valores de X positivos, ocorrendo então um efeito ricochete onde X devolve valores negativos até atingir o estado de repouso.



**Figura 6 - Gráfico com o registo da movimentação horizontal do comando para o lado esquerdo**

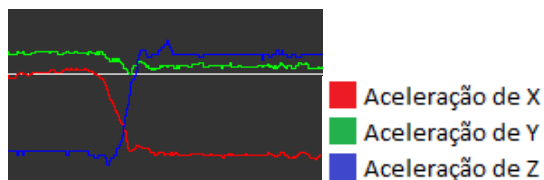
Verifica-se na Figura 7 que estando o comando numa posição de repouso e efectuando um movimento para o lado direito verificam-se as seguintes alterações no gráfico: X diminui, Z mantém, Y aumenta.

É possível verificar que primeiro há uma acentuação para valores de X negativos, ocorrendo então um efeito ricochete onde X devolve valores positivos até atingir o estado de repouso.



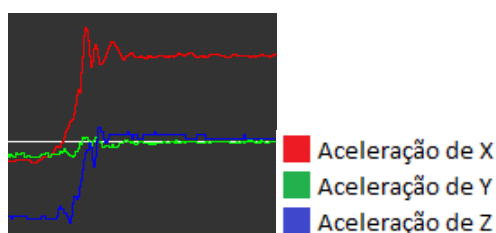
**Figura 7 - Gráfico com o registo da movimentação horizontal do comando para o lado direito**

Comprova-se na Figura 8 que estando o comando numa posição de repouso e efectuando um movimento de rotação para a direita, deixando o comando, posteriormente nesse estado de repouso, verificam-se as seguintes alterações no gráfico: Z diminui para 0, X aumenta para 1, Y mantém.



**Figura 8 - Gráfico com o registro da rotação do comando para o lado direito**






Nota-se na Figura 9 que estando o comando numa posição de repouso e efectuando um movimento de rotação para a esquerda, deixando o comando, posteriormente nesse estado de repouso, verificam-se as seguintes alterações no gráfico: Z diminui para 0, X diminui para -1, Y mantém.



**Figura 9 - Gráfico com o registro da rotação do comando para o lado esquerdo**

### 2.3.3 Repouso

Na Tabela 2 serão mostrados os valores registados consoante a posição do comando em repouso.

	 (repouso horizontal)	 (comando a apontar para cima)	 (comando a apontar para baixo)	 (comando deitado com botões para a direita)	 (comando deitado com botões para a esquerda)
X	0	0	0	1	-1
Y	0	-1	1	0	0
Z	1	0	0	0	0

**Tabela 2 – Tabela demonstrativa dos valores registados pelo comando em várias posições de repouso**

## 2.4 Requisitos Funcionais

Os requisitos funcionais dividem-se em 4 contextos: funções no rato, funções no teclado, funções no *browser* e funções no comando.

As funções no rato num âmbito mais amplo, podendo ser utilizado em todo o sistema operativo, pois a resposta é sempre a mesma. Permite controlar o sistema operativo.

As funções no teclado num contexto mais restrito. Apenas a função da tecla *Enter* é apropriada para ser utilizada noutras aplicações do sistema operativo, enquanto as restantes funções listadas restringem a sua utilização ao *browser*.

As funções no *browser* são funções para serem utilizadas apenas na interacção com o *browser*, enquanto as funções do comando servem para definir os botões do comando que desempenham as funções anteriores.

Funções no Rato:

- Controlar o cursor do rato.
- Activação do clique esquerdo do rato.
- Activação do clique direito do rato.
- Activação do clique do meio do rato.

Funções no Teclado:

- Activar a tecla *Enter*.
- Permitir a escrita de texto.
- Ir para a próxima página (*Page up*).
- Ir para a página anterior (*Page down*).
- Activação da tecla de tabulação.

Funções no *Browser*:

- Abrir o browser pré-definido.
- Avançar e retroceder utilizando a aceleração do comando.
- Aumentar tamanho da letra.
- Diminuir o tamanho da letra.
- Abrir os favoritos.
- Abrir uma hiperligação sem a necessidade de parar o cursor do rato sobre essa hiperligação e clicar.
- Fechar aplicação.
- Fechar actual separador.
- Abrir um novo separador.

Funções no *Wii Remote*:

- Botões configuráveis.

## 2.5 Requisitos não funcionais

Foram considerados os seguintes requisitos não funcionais do sistema:

- **Operacionais:** O sistema deverá funcionar em diferentes sistemas operativos.
- **Segurança:** o sistema apenas deverá permitir a ligação a um comando.
- **Desempenho:** O tempo de resposta deve ser o mesmo do rato/teclado. Movimentação do rato deverá ser precisa.
- **“Usabilidade”:** Sistema de fácil utilização e intuitivo.
- **Compatibilidade:** ser compatível com vários adaptadores *Bluetooth*.

No próximo capítulo serão abordadas questões a nível do desenho do projecto, nomeadamente decisões tomadas sobre como os movimentos são efectuados, sobre ideias iniciais e o porquê de ter havido alterações, justificando as decisões a nível da configuração original dos botões ao mesmo tempo que será indicado o que essa configuração actualmente permite. Serão discutidos desenhos alternativos e indicado o porquê de terem ou não sido usados. Haverá uma apresentação do diagrama de classes, de modo a haver um melhor entendimento e percepção das classes e dos objectos da aplicação. No final do capítulo falar-se-á da primeira fase de testes de utilizadores, a qual levou a que algumas decisões fossem tomadas.

## 3 Desenho

Neste capítulo serão abordadas questões a nível do desenho do projecto, nomeadamente decisões tomadas sobre como os movimentos são efectuados, sobre ideias iniciais e o porquê de ter havido alterações, justificando as decisões a nível da configuração original dos botões ao mesmo tempo que será indicado o que essa configuração actualmente permite. Serão discutidos desenhos alternativos e indicado o porquê de terem ou não sido usados. Haverá uma apresentação do diagrama de classes, de modo a haver um melhor entendimento e percepção das classes e dos objectos da aplicação. No final do capítulo falar-se-á da primeira fase de testes de utilizadores, a qual levou a que algumas decisões fossem tomadas.

### 3.1 Ideias Iniciais

Inicialmente pensou-se em otimizar o comando para navegação na internet e para apresentações. Algumas das acções originalmente idealizadas incluíam o avançar e retroceder no navegador de internet serem feitos através de gestos do comando, activar-se uma opção que permitisse a introdução de texto a partir do próprio comando, uma funcionalidade que ao ser usada fizesse com que o cursor saltasse para a hiperligação mais próxima, abrir uma hiperligação sem a necessidade de parar o cursor do rato sobre essa hiperligação para clicar e outras funções mais simples como aumentar e diminuir o tamanho de letra.

### 3.2 Movimento do Comando

Com base na análise efectuada verificou-se que, teoricamente, seria possível controlar o cursor do rato como se o *Wii Mote* fosse um ponteiro tendo por base os valores de aceleração registados. Esses mesmos valores seriam os usados para o registo da acção de retroceder e avançar. No entanto, como já foi referenciado no capítulo da análise, constatou-se que o movimento ficava demasiado errático e nada fiável, motivo pelo qual optou-se por deixar de parte essa forma de movimento. Também devido à situação da aceleração, decidiu-se abandonar a questão do avançar e retroceder com os movimentos do comando, apesar de funcionar esporadicamente, não atingia os níveis de qualidade esperados e durante a primeira fase de testes verificou-se que os utilizadores preferiam fazer estas acções através de botões do comando e não através de gestos.

Desta forma, o movimento do comando é feito através de rotações quando se quer fazer uma deslocação horizontal com o cursor e com inclinações se se pretender uma deslocação vertical.



### 3.3 Configuração Actual

A actual configuração do comando está optimizada para o funcionamento com o navegador de internet Mozilla Firefox, com atalhos no próprio comando para variadas acções, nomeadamente o avançar e retroceder, abrir e fechar separadores, aumentar e diminuir o tamanho de letra e abrir os marcadores. Outras funcionalidades implementadas funcionam de uma forma mais geral a nível do sistema operativo sem estarem vocacionadas para o Firefox, como é o caso do clique esquerdo e direito, o botão enter e as teclas cursoras ao que se acrescenta a tabulação, o botão home e end.

Para que estas funções funcionem noutros browsers seria necessário ajustar a configuração no próprio código e modificar as próprias teclas de atalho.

### 3.4 Desenhos Alternativos

Das ideias originalmente pensadas como o caso do avançar e retroceder no navegador de internet ser feito através de gestos do comando, activar-se uma opção que permita a introdução de texto a partir do próprio comando e uma funcionalidade que ao ser usada fizesse com que o cursor saltasse para a hiperligação mais próxima, nenhuma acabou por ser usada.

A questão do avançar e retroceder através de gestos no comando chegou a ser implementada, mas funcionava de forma por vezes errática e houve queixas de alguns utilizadores na primeira fase de testes relativamente ao seu funcionamento, pelo que se optou por eliminar essa função da aplicação.

A funcionalidade que permitisse a introdução do texto a partir do comando foi apenas idealizada não tendo havendo nenhuma tentativa de implementação.

Relativamente à funcionalidade que permite que o cursor salte para a ligação mais próxima, ainda houve um processo de investigação e pesquisa sobre a forma de implementar isso, mas devido a outras prioridades, esta opção foi colocada de parte.

### 3.5 Diagrama de Classes

O diagrama de classes da Figura 10 mostra a representação da estrutura e relações das classes desenvolvidas e das classes e *packages* utilizadas no sistema.

Não foram incluídos os atributos e as operações para facilitar a sua compreensão.

A classe `wiimando.model.WiiMando` herda da classe `wiiremotej.WiiRemoteAdapter` métodos que permitem a captura de eventos da aceleração e dos botões. Depende da classe `WiiRemote`

para a ligação ao comando e activar as suas funcionalidades. Depende da *package* javax.swing devido aos *JOptionPanels* utilizados nas mensagens de erro e da *package* java.awt para movimentar o rato através da sua classe Robot, além de outras utilidades.

As classes das funcionalidades, por exemplo a wiimando.model.Up, são classes que herdam as principais operações da classe abstracta wiimando.model.ButtonEvent. Todas elas possuem um objecto da classe wiimando.model.XMLReader e esta um objecto de cada, como se pode observar na cardinalidade entre ambas. Esta classe tem também uma relação de agregação com a classe wiimando.model.WiiMando, uma relação forte em que a classe wiimando.model.WiiMando tem um objecto seu, e uma associação com a classe wiimando.model.Combo com uma cardinalidade de 1:1. Existem depois outras dependências de classes relacionadas com a leitura dos documentos XML.

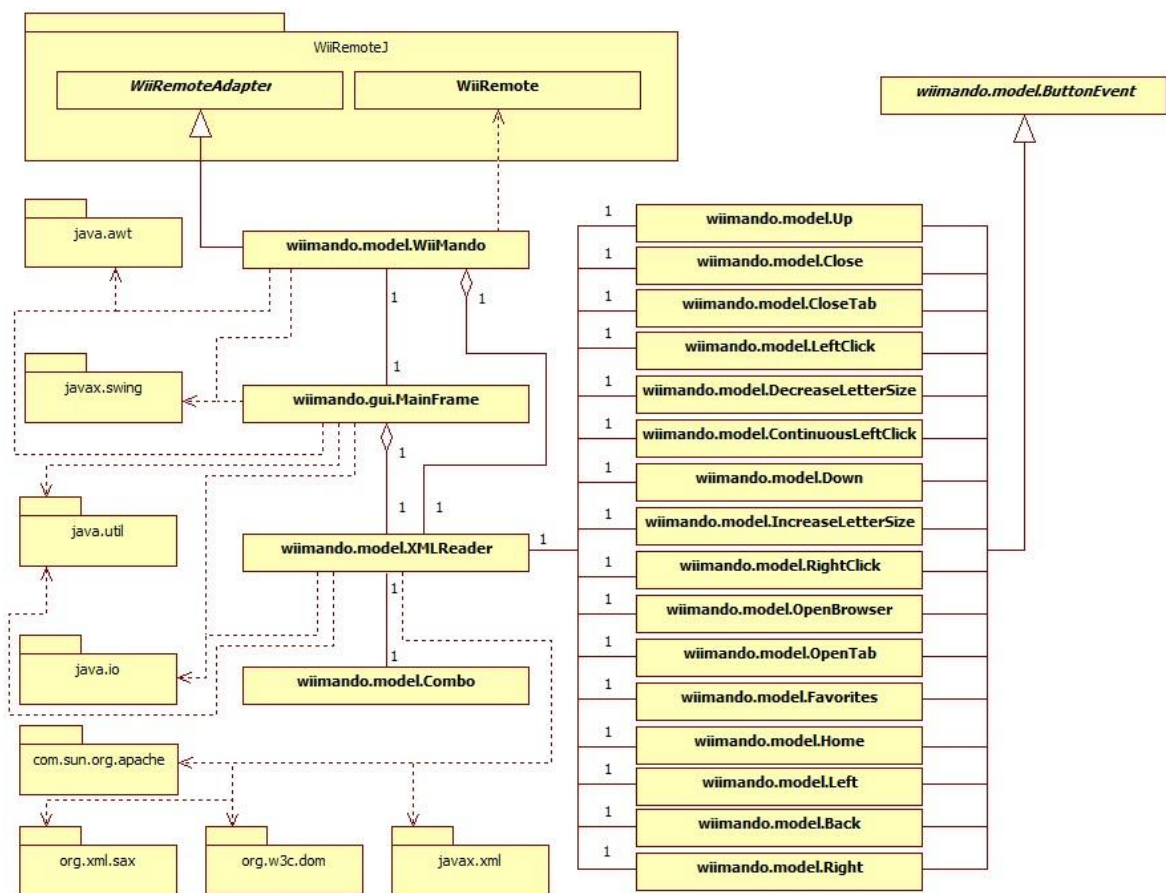


Figura 10 – Diagrama de classes gerado com o programa StarUML [16]

### 3.6 Primeira Fase de Testes com Utilizadores

Na primeira fase de testes com utilizadores foi usado um protótipo inicial da aplicação contendo apenas os movimentos do cursor e alguns eventos de botão implementados. Esta fase serviu para verificar os melhores locais onde colocar as funcionalidades a nível de botões do comando e para ouvir sugestões dos utilizadores sobre funcionalidades adicionais.

Durante esta fase, foi utilizado um grupo de 4 utilizadores com graus variados de conhecimento, foi possível verificar em termos de consenso geral, que a funcionalidade de botão esquerdo do rato ficaria melhor aplicada no botão A do comando enquanto o B ficaria para combinações. Esta troca possibilita a utilização das combinações e do comando só com uma mão.

Verificou-se que era necessário algum tempo de habituação do utilizador ao comando, mas que posteriormente os movimentos começavam a ser fluidos, com a ocorrência de diagonais e os utilizadores a fazerem pequenos gestos, mais bruscos, de modo a obterem uma maior precisão, constatando-se adicionalmente, que os utilizadores viam mais vantagens em usar os botões direccionais do comando para fazer o retroceder e o voltar, do que propriamente fazer essas acções através de movimentações do comando. Para além de disso, alguns utilizadores sentiram desconforto no pulso na utilização da aceleração para retroceder e avançar.

Observou-se a necessidade da existência de dois graus de velocidade do cursor, um para quando a aceleração é menor, o movimento ser mais lento e outra para quando a aceleração é maior, o movimento ser mais rápido. Esta funcionalidade permitirá melhorar a precisão.

Atendendo às sugestões dos utilizadores decidiu-se que deveria ser implementada uma função que permita a inversão do movimento horizontal e vertical. Podendo esta função vir a ser utilizada em aplicações diferentes do browser como simuladores de voo. Desta forma aumenta-se o número potencial de utilizadores pela maneira como aumenta a compatibilidade com outros softwares como jogos.

Foi no final desta fase que se chegou a um certo consenso sobre qual seria a configuração que ficaria pré-definida nos botões, bem como, o nível de velocidade que ficaria no ponteiro.

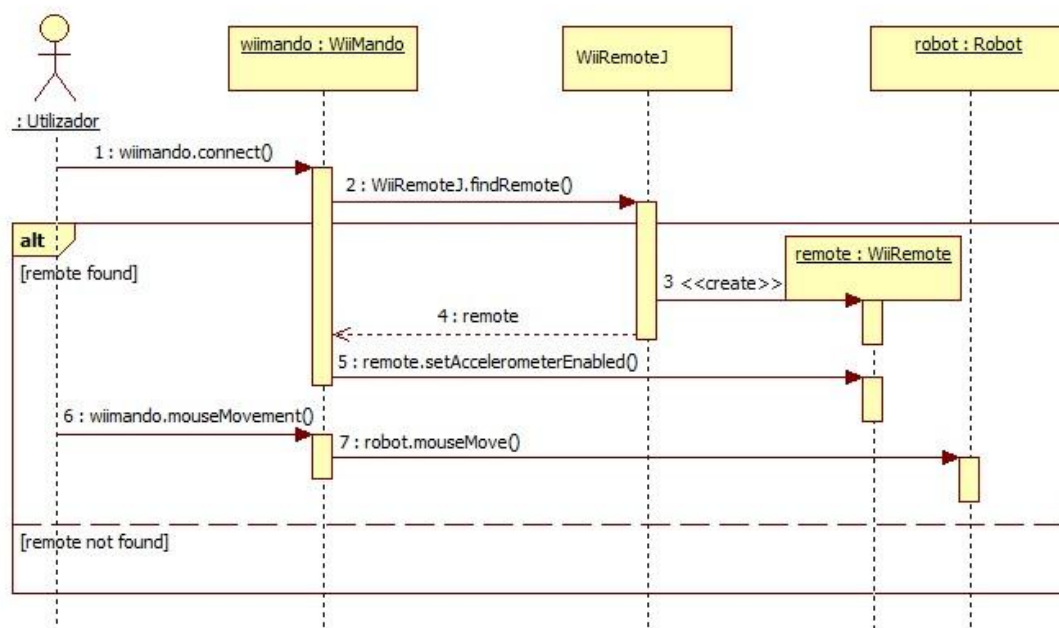
No próximo capítulo será explicado detalhadamente o funcionamento do programa. Serão analisados alguns eventos através de diagramas de sequências. Referenciam-se alguns aspectos da programação que levaram a que tivessem sido tomadas algumas decisões ao nível de desenho.

## 4. Implementação

Neste capítulo será explicado detalhadamente o funcionamento do programa. Serão analisados alguns eventos através de diagramas de sequências. Referenciam-se alguns aspectos da programação que levaram a que tivessem sido tomadas algumas decisões ao nível de desenho.

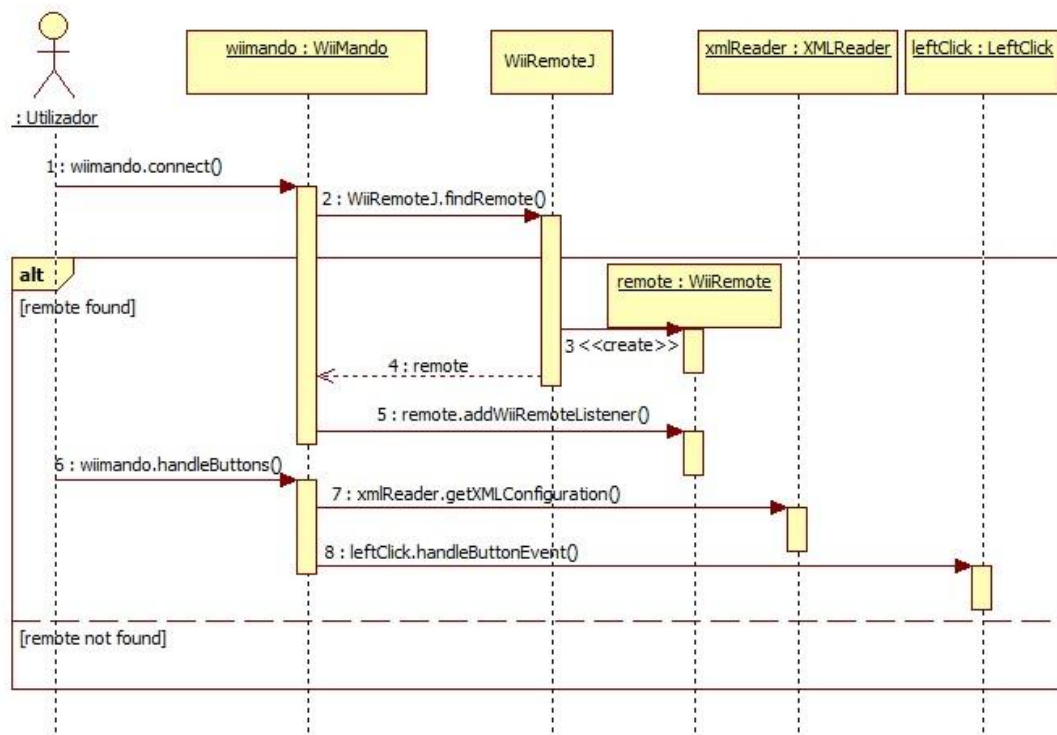
### 4.1 Diagramas de Sequência

O diagrama de classes da Figura 11 representa a sequência do processo da ligação ao comando e o movimento do cursor do rato. É possível observar as interações entre os objectos nestes dois processos. A ligação ao comando, no passo 1, é impulsionada pelo utilizador através da interface gráfica, provocando a tentativa de ligação ao comando no passo 2. Se houver sucesso nessa tentativa, é activado o acelerómetro no passo 3, caso contrário, não acontecerá nenhum dos passos seguintes. O passo 4 corresponde ao processo de movimentação do cursor do rato no ecrã, que é realizado sempre que é alterada a posição do comando e se regista uma aceleração, provocando no fim do processo a movimentação (passo 5). No subcapítulo 4.2 - Ligação ao comando, é descrito com detalhe o processo de ligação e no subcapítulo 4.5 – Movimento do Cursor, é explicado o processo de movimentação do cursor detalhadamente.



**Figura 11 – Ligação do comando e movimento do rato. Diagrama de sequências gerado com o programa StarUML.**

O diagrama de classes da Figura 12 representa a sequência do processo da ligação ao comando e as funções dos botões. As etapas necessárias para a ligação ao comando foram abordadas no diagrama da Figura 11 e como tal vamos analisar o processo de gestão das funções dos botões. Sempre que o utilizador pressiona um botão, esse evento é captado e utilizado no método referido no passo 4 do diagrama. É verificada a configuração actual dos botões no passo 5 e se corresponder, como no exemplo do passo 6, a um clique esquerdo é efectuada essa acção. Foi mostrado apenas o funcionamento do clique esquerdo porque as restantes funcionalidades processam-se da mesma forma. No subcapítulo 4.3 – Leitura da Configuração dos Botões, é explicado com detalhe o funcionamento do processo 5 do diagrama, correspondente à leitura do ficheiro de configuração dos botões. No subcapítulo 4.4 – Detecção de eventos, é explicado com detalhe o processo 6 do diagrama, mostrando como o sistema reage quando é pressionado um botão.



**Figura 12 – Ligação do comando e registo do evento do clique esquerdo do rato. Diagrama de sequências gerado com o programa StarUML.**

## 4.2 Ligação ao comando

O comando é emparelhado com o computador através do método `void connect(MainFrame mainFrame)` da classe `WiiMando` que está localizada no *package* `wiimando.model`:

```
public void connect(MainFrame mainFrame)
{
    (...)
    mainFrame.connecting();
    try {
        this.remote = null;
        long timeout = System.currentTimeMillis();
        while (this.remote == null)
        {
            if ((System.currentTimeMillis() - timeout) > 5000) {
                mainFrame.updateConnection();
            } else {
                try {
                    (...)
                    this.remote = WiiRemoteJ.findRemote();
                } catch (Exception e) {
                    this.remote = null;
                    (...)
                    mainFrame.updateConnection();
                }
            }
        }
        this.remote.addWiiRemoteListener(this);
        this.remote.setAccelerometerEnabled(true);
        (...)
        this.connectionState = true;
        mainFrame.updateConnection();
    }
}
```

Este método é activado pelo utilizador na interface gráfica através do botão `connectButton` inserido na *JFrame* `mainframe` da classe `wiimando.model.Mainframe`. Este botão surge ao utilizador com o valor “Ligar” como se pode observar na Figura 13.

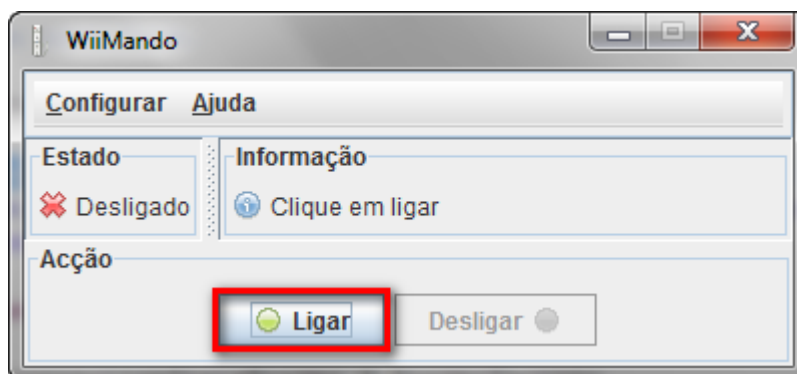


Figura 13 – Interface principal que apresenta o botão ligar ao utilizador.

Ao botão foi adicionado um `ActionListener`, quando este é pressionado o botão é desactivado e é iniciada uma *thread* para garantir que a execução do método `connect(MainFrame mainFrame)` é feita sem interferências. Para chamar este método foi criado na classe `MainFrame` o objecto `wiimando` da classe `WiiMando`.

O método de ligação foi feito com base no código fornecido na classe de testes `WRLImpl`.

O método tem como parâmetro um objecto da classe `MainFrame` (*package* `wiimando.gui`) para permitir a actualização da interface, à medida que é estabelecida a ligação, utilizando para esse efeito os métodos `void connecting()` e `void updateConnection()`.

Neste método foi implementada uma condição de *timeout*. É definido o tempo em que se iniciou a busca do comando na variável `long timeout` em milissegundos com a função `System.currentTimeMillis()`. Quando é excedido o tempo definido é chamada a função `updateConnection()` da classe `MainFrame` através do objecto `mainframe`. Este método interrompe a *thread* e consequentemente a tentativa de ligação por ter excedido o tempo, informando o utilizador e activando de novo o botão “Ligar” na interface gráfica.

Este método tem como objectivo a inicialização do objecto `remote` da classe `WiiRemote` que inicialmente se encontra a `null` e equivale ao estado desligado.

Sabendo da possibilidade do comando não ser encontrado, e essa ser uma excepção, foi criado o bloco `try`. Neste bloco é feita a busca do comando utilizando o método `static WiiRemote findRemote()`. Este método da classe `WiiRemoteJ`, pertencente à biblioteca `WiiRemoteJ`, procura o comando e caso o encontre emparelha-o com o computador via *Bluetooth*. Ao encontrar o comando, o objecto `remote` é inicializado e são activadas funções fundamentais para o funcionamento como o acelerómetro através do método `void setAccelerometerEnabled(boolean enabled)` e a captura de eventos através do método `void addWiiRemoteListener(WiiRemoteListener listener)` que posteriormente é utilizado para capturar os eventos dos botões com os eventos do tipo `WRButtonEvent`.

Caso ocorra uma excepção o objecto `remote` continua com o valor `null`, o comando não fica emparelhado com o computador e nenhuma das funções do sistema funcionam.

Caso a biblioteca `bluecove` verifique que o receptor *Bluetooth* não está operacional ou não está instalado, o utilizador é informado através de um `JOptionPane`.

## 4.3 Leitura da Configuração dos Botões

Para que o sistema fosse personalizável e não houvesse a necessidade de alterar as suas configurações sempre que este é iniciado, a sua configuração é guardada em documentos XML que funcionam como a base dados do sistema. A sintaxe restrita e requerimentos de *parsing* destes documentos tornam a sua leitura mais eficiente e consistente.

Com estes ficheiros de configuração é possível ao utilizador ter várias configurações para os botões do comando por exemplo. Criaram-se ficheiros de configuração para as seguintes funcionalidades: último ficheiro de configuração carregado pelo sistema, para que o sistema se inicie da próxima vez com as últimas configurações definidas pelo utilizador; funcionalidades de cada botão; a calibração do comando; página de entrada na abertura do *browser*.

Veja-se o exemplo da estrutura do documento “config.xml”:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<configuration>
  <combo>WRButtonEvent.B</combo>
  <button>
    <name>close</name>
    <event>WRButtonEvent.ONE</event>
    <combination>true</combination>
  </button>
  <button>
    <name>back</name>
    <event>WRButtonEvent.LEFT</event>
    <combination>false</combination>
  </button>
  (...)
</configuration>
```

A leitura só é feita quando o comando está ligado e a configuração não está a ser alterada, enquanto a configuração está a ser alterada, o comando é desligado e não é feita a leitura. É necessário desligar o comando porque nas primeiras versões do sistema verificou-se que por vezes o ficheiro de configuração era lido quando ainda não estava completamente formado, provocando o término do programa devido a um erro de *parsing*.

Como a leitura se processa da mesma forma para todos os ficheiros de configuração será explicado o funcionamento daquele que é o mais complexo, o ficheiro de configuração dos botões.

A leitura dos ficheiros de configuração é feita na classe `wiimando.model.XMLReader`, através do método `void getXMLConfiguration(String filename)`. Este método recebe como parâmetro o nome do ficheiro de configuração que vai ser lido. Este método é chamado pelo método `void handleButtons(evt)` da classe `wiimando.model.Wiimando` cujo funcionamento é explicado no subtítulo seguinte.



A leitura é feita elemento a elemento recorrendo ao *parser* da *package* `javax.xml.parsers`, conseguindo este entender a estrutura do documento e obter os elementos pelo nome:

```
(...)  
NodeList buttonNodeLst = doc.getElementsByTagName("button");  
NodeList comboNodeLst = doc.getElementsByTagName("combo");  
(...)
```

Os valores obtidos são *strings* que necessitam de ser convertidas para o tipo *Integer* para que possam ser interpretados como eventos de botões (`WRButtonEvent`). Essa conversão foi feita através do seu mapeamento com o *HashMap* `<String, Integer> eventMapping`. O *HashMap* devolve o valor inteiro da chave do tipo *string* obtida do documento e guarda-o na variável `data`. Este valor é depois atribuído a uma das variáveis inteiras cujo nome representa a função. A atribuição é feita através do método `void setButtonValue(String function, int value)`:

```
public void setButtonValue(String function, int value) {  
    if (function.equals("left_click")) {  
        this.leftClickBtn = value;  
    } else if (function.equals("down")) {  
(...)  
    }
```

Se o valor do elemento for, neste caso, a função `left_click`, a variável representativa do botão que desempenha essa função, fica com o valor da variável `data`. Por exemplo:

```
this.leftClickBtn = WRButtonEvent.A;
```

Ou seja, o clique esquerdo ficou associado ao botão A.

De seguida o *HashMap* `<String, String> combination` guarda o nome da função, obtido através da leitura, e se esta função é executada através da sua combinação com outro botão. Por exemplo `("left_click", "false")`.

Sabendo o valor do evento (`evt`) de cada botão e se este é combinação, estes dados são utilizados nas classes das funcionalidades, presentes na *package* `wiimando.model`, mais precisamente no método `void handleButtonEvent(WRButtonEvent evt, Robot robot, XMLReader xmlReader)`, comum a todas as classes e herdado da classe `wiimando.model.ButtonEvent`, veja-se mais uma vez o exemplo do clique esquerdo:

```
public void handleButtonEvent(WRButtonEvent evt, Robot robot,  
    XMLReader xmlReader) {  
    if (!xmlReader.getLastLoadedFile().equals("")) {  
        if ((xmlReader.getCombination("left_click").equals("true"))) {  
            {  
                (...)  
            }  
        }  
    }  
    (...)  
}
```

O funcionamento deste método é explicado no subtítulo seguinte de detecção de eventos.

## 4.4 Detecção de eventos

Antes de poder haver qualquer tipo de detecção de eventos, é necessário que o comando esteja ligado. Ao ser estabelecida a ligação do comando é executado o método `remote.addWiiRemoteListener(this)` que fica a aguardar que os botões do *Wii Remote* sejam pressionados, transmitindo esse evento para a aplicação.

Juntamente com a biblioteca e a classe de testes vinha o método `public void buttonInputReceived(WRButtonEvent evt)` que na sua versão original enviava mensagens de texto para a consola à medida que os botões no comando eram pressionados. Este método foi alterado de modo a que o mesmo chamasse um outro implementado por nós, o método `handleButtons(evt)` que fará a detecção de eventos.

Os eventos são detectados através do método `void handleButtons(WRButtonEvent evt)` da classe `WiiMando` que está localizada no *package* `wiimando.model`. Dentro do método existe o seguinte código:

```
this.xmlReader.getXMLConfiguration(this.xmlReader.getXMLLastLoadedFile());

this.close.handleButtonEvent(evt, this.robot, xmlReader);
this.back.handleButtonEvent(evt, this.robot, xmlReader);
this.closeTab.handleButtonEvent(evt, this.robot, xmlReader);
this.cLeftClick.handleButtonEvent(evt, this.robot, xmlReader);
this.dLetterSize.handleButtonEvent(evt, this.robot, xmlReader);
this.down.handleButtonEvent(evt, this.robot, xmlReader);
this.end.handleButtonEvent(evt, this.robot, xmlReader);
this.enter.handleButtonEvent(evt, this.robot, xmlReader);
this.favorites.handleButtonEvent(evt, this.robot, xmlReader);
this.forward.handleButtonEvent(evt, this.robot, xmlReader);
this.home.handleButtonEvent(evt, this.robot, xmlReader);
this.iLetterSize.handleButtonEvent(evt, this.robot, xmlReader);
this.left.handleButtonEvent(evt, this.robot, xmlReader);
this.leftClick.handleButtonEvent(evt, this.robot, xmlReader);
this.openBrowser.handleButtonEvent(evt, this.robot, xmlReader);
this.openTab.handleButtonEvent(evt, this.robot, xmlReader);
this.right.handleButtonEvent(evt, this.robot, xmlReader);
this.rightClick.handleButtonEvent(evt, this.robot, xmlReader);
this.tab.handleButtonEvent(evt, this.robot, xmlReader);
this.up.handleButtonEvent(evt, this.robot, xmlReader);
```

A primeira coisa a ser feita é a chamada do método `getXMLConfiguration` que irá carregar a actual configuração de botões. Posteriormente são chamados todos os objectos de funcionalidades para que seja verificado qual a função associado ao botão, ou botões pressionados e assim executada a opção pretendida. A forma como isso acontece é através da chamada do método `handleButtonEvent` de cada objecto. Este método é herdado da classe abstracta `ButtonEvent` e aplica-se a todos os objectos das funcionalidades, sendo portanto o processo igual em todos. Tendo por exemplo o clique esquerdo, é executado o seguinte código:

```

public void handleButtonEvent(WRButtonEvent evt, Robot robot,
    XMLReader xmlReader)
{
    if (!xmlReader.getLastLoadedFile().equals(""))
    {
        if (xmlReader.getCombination("left_click").equals("true"))
        {
            if (evt.wasPressed(xmlReader.getLeftClickBtn())
                && (evt.isPressed(xmlReader.getComboBtn())))
            {
                robot.mousePress(InputEvent.BUTTON1_MASK);
            }
            if (evt.wasReleased(xmlReader.getLeftClickBtn())
                && (evt.isPressed(xmlReader.getComboBtn())))
            {
                robot.mouseRelease(InputEvent.BUTTON1_MASK);
            }
        }
        else if ((xmlReader.getCombination("left_click").equals("false")))
        {
            if (evt.wasPressed(xmlReader.getLeftClickBtn())
                && !(evt.isPressed(xmlReader.getComboBtn())))
            {
                robot.mousePress(InputEvent.BUTTON1_MASK);
            }
            if (evt.wasReleased(xmlReader.getLeftClickBtn())
                && !(evt.isPressed(xmlReader.getComboBtn())))
            {
                robot.mouseRelease(InputEvent.BUTTON1_MASK);
            }
        }
    }
    else
    {
        if (evt.wasReleased(WRButtonEvent.A)
            && !(evt.isPressed(WRButtonEvent.B)))
        {
            if (evt.wasPressed(WRButtonEvent.A)
                && (evt.isPressed(WRButtonEvent.B)))
            {
                robot.mousePress(InputEvent.BUTTON1_MASK);
            }
            if (evt.wasReleased(WRButtonEvent.A)
                && (evt.isPressed(WRButtonEvent.B)))
            {
                robot.mouseRelease(InputEvent.BUTTON1_MASK);
            }
        }
    }
}

```

Primeiramente é feita uma chamada do método `xmlReader.getLastLoadedFile()` de modo a se saber se existe actualmente um ficheiro de configuração carregado ou não. Na eventualidade de existir, é feita uma sequência de *if's* onde se executa o método `xmlReader.getCombination("left_click")` para que se devolva o valor de combinação para o *left\_click*. Isto é feito para que no caso de ser verdadeiro, significa que a função de clique esquerdo funciona através de uma combinação de botões, se for falso, é porque não é feito através de combinação. Seguidamente ocorrem as chamadas dos métodos `xmlReader.getLeftClickBtn()` e `xmlReader.getComboBtn()` para que se verifique quais os botões de comando associadas às funções de clique esquerdo e de combinação e consoante a

necessidade de combinação ou não, executa-se o `robot.mousePress(InputEvent.BUTTON1_MASK)` seguido do `robot.mouseRelease(InputEvent.BUTTON1_MASK)` causando o evento de clique esquerdo no sistema operativo.

Na eventualidade do ficheiro de configuração não estar presente, existe uma feita no próprio código onde o botão A do *Wii Mote* funciona como clique esquerdo e o botão B como combinação.

## 4.5 Movimento do cursor

O movimento do rato é feito com os valores da aceleração captados pelo método `void accelerationInputReceived(WRAccelerationEvent evt)` herdado da classe `WiiRemoteAdapter`, pertencente à biblioteca `WiiRemoteJ`. Este método permite captar as acelerações de X (`evt.getXAcceleration()`) e Y (`evt.getYAcceleration()`) através do objecto `evt`, um evento da aceleração do comando. Estes eventos são captados pelo método referido e servem de parâmetro ao método `void mouseMovement(WRAccelerationEvent evt)`:

```
public void accelerationInputReceived(WRAccelerationEvent evt) {  
  
    if (accelerometerSource) {  
  
        this.mouseMovement(evt);  
    }  
}
```

Após obter os valores da aceleração, o movimento do rato é determinado pelo método `mouseMovement`:

```
public void mouseMovement(WRAccelerationEvent evt){  
  
    Point p = java.awt.MouseInfo.getPointerInfo().getLocation();  
    double xAccel = (evt.getXAcceleration()  
        + this.getXCalibration())  
        * this.getHorizontalDirection();  
    double yAccel = (evt.getYAcceleration()  
        + this.getYCalibration())  
        * this.getVerticalDirection();  
  
    try {  
        this.robot = new Robot();  
        if (Math.abs(xAccel) > 0.6) {  
            p.x += xAccel * this.getPointerFastSpeed();  
        } else if (Math.abs(xAccel) < 0.6) {  
            p.x += xAccel * this.getPointerSlowSpeed();  
        }  
        if (Math.abs(yAccel) > 0.55) {  
            p.y += yAccel * this.getPointerFastSpeed();  
        } else if (Math.abs(yAccel) < 0.55) {  
            p.y += yAccel * this.getPointerSlowSpeed();  
        }  
        this.robot.mouseMove(p.x, p.y);  
    } catch (AWTException e) {  
        e.printStackTrace();  
    }  
}
```

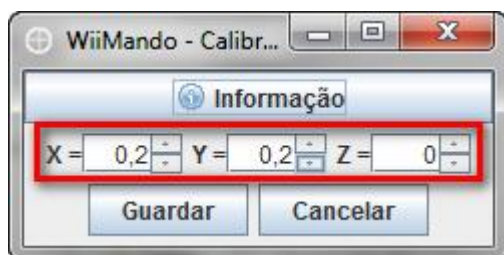
O movimento do rato deve ser feito a partir da posição em que este se encontra. Esta posição é obtida a partir das coordenadas do ponto *p*.

Foram definidas duas variáveis, *xAccel* e *yAccel*, cujo valor é o da aceleração para cada eixo, somado com o valor da calibração e multiplicado pelo sentido do movimento vertical e horizontal:

```
double xAccel = (evt.getXAcceleration()
                + this.getXCalibration())
                * this.getHorizontalDirection();

double yAccel = (evt.getYAcceleration()
                + this.getYCalibration())
                * this.getVerticalDirection();
```

A biblioteca WiiRemoteJ já implementa um método para calibração, no entanto verificou-se que este não é perfeito e mesmo com o comando em repouso é registado um valor de aceleração diferente de zero. Surgiu então a necessidade de implementar um método que calibre o comando de forma perfeita. Tentou-se implementar um método que calibrasse o comando automaticamente mas como não foi conseguido, a calibração é feita de forma manual, pelo utilizador como se pode observar na Figura 14:

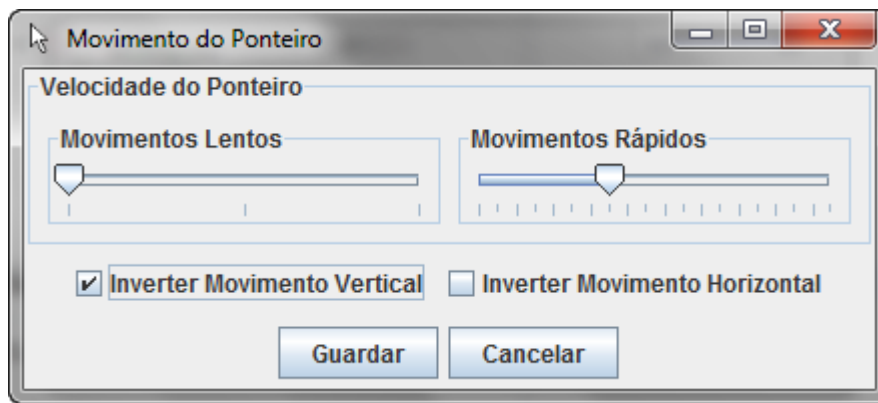


**Figura 14 – Calibração manual do comando.**

O utilizador deve testar valores até que o comando se mantenha em repouso. O repouso acontece quando os valores de aceleração somados aos da calibração são zero.

Estes valores são gravados no ficheiro XML “calibration.xml” pelo método `void saveCalibrationValues(double x, double y, double z)` e lidos pelos métodos `double getXCalibration()` e `double getYCalibration()` da classe `wiimando.gui.XMLReader`.

Para completar o valor do movimento resta definir o valor da inversão. Este é definido pelo utilizador na configuração como se pode observar na Figura 15:



**Figura 15 – Configuração da inversão do sentido do movimento do cursor do rato.**

Os valores definidos pelo utilizador são gravados no ficheiro XML “inversion.xml” e utilizados nos métodos `getHorizontalDirection()` e `getVerticalDirection()` da classe `WiiMando`. A escrita e leitura dos ficheiros XML são feitas na classe `XMLReader`. Os métodos referidos lêem o valor booleano da inversão, se for `true` o valor é -1 e ao multiplicar pelo valor da aceleração inverte o movimento.

Foram definidas 2 velocidades, uma lenta para movimentos lentos e precisos e outra rápida para deslocações rápidas do cursor ao longo do ecrã.

Estas duas velocidades são factores que são multiplicados aos valores de `xAccel` e `yAcel` calculados anteriormente e são definidos pelo utilizador na interface como se pode observar na Figura 15.

Se o valor absoluto da aceleração de X, calculado da forma referida anteriormente, for superior a 0.6, o movimento realizado é rápido e o valor do factor é o definido pelo utilizador no `JSlider fastSpeedSlider` em “Movimentos Rápidos”, caso contrário é um movimento lento e o factor é o valor definido no `JSlider slowSpeedSlider` em “Movimentos Lentos”.

O valor mínimo definido para movimentos lentos foi 3 e o máximo 5. Definiu-se 3 após serem efectuados alguns testes que permitiram concluir que o comportamento do cursor era muito errático.

Foi considerado que o valor mínimo para movimentos rápidos deveria ser 6 e no máximo 20, dado que é um valor onde a velocidade do cursor já fica perto de um limiar de controlo.

Estes valores, lentos ou rápidos, vão sendo incrementados e utilizados para mover o cursor do rato, recorrendo à classe `java.awt.Robot` através do método `mouseMove(p.x, p.y)`.

No próximo capítulo serão abordados os testes efectuados com utilizadores, especificamente a fase onde já foi usada uma versão final do programa e onde foram efectuadas algumas tarefas com grupos heterogéneos de utilizadores de modo a se verificarem os erros dados e haver uma medição do tempo levado. No final, os resultados são analisados.

## 5 Testes com utilizadores

Neste capítulo serão abordados os testes efectuados com utilizadores, especificamente a fase onde já foi usada uma versão final do programa e onde foram efectuadas algumas tarefas com grupos heterogéneos de utilizadores de modo a se verificarem os erros dados e haver uma medição do tempo levado. No final, os resultados são analisados.

Esta fase de testes ficou caracterizada por já ser usada uma versão final do programa. Ao contrário da primeira fase de testes com utilizadores, esta fase não serviu para tomar decisões e adicionar funcionalidades, mas para medir o desempenho do sistema.

O grande objectivo desta fase passou por testar a precisão dos utilizadores na utilização do comando como ponteiro do rato. Durante essas tarefas foram medidos os erros e os tempos na execução das tarefas para posterior análise.

### 5.1 Descrição do ambiente em que se realizaram as tarefas

Devido à impossibilidade de utilizar um projector e uma tela na realização dos testes, o ambiente em que se pretende utilizar o sistema, a alternativa foi ligar o computador a uma televisão LCD de 82 centímetros.

A ligação entre o computador e a televisão foi feita por HDMI, possibilitando aos utilizadores uma melhor definição de imagem. Isto permitiu aproximar a televisão LCD ao projector em termos de qualidade, de modo que o utilizador não perdesse tempo devido a dificuldades de visualização do ecrã.

Os testes foram feitos numa sala onde apenas se encontravam os coordenadores do teste e o utilizador.

### 5.2 Características dos participantes

Os testes contaram com 14 utilizadores de vários níveis de conhecimento, divididos de forma heterogénea em 3 grupos. Dois grupos de cinco utilizadores, grupo 1 (G1) e grupo 2 (G2) utilizaram o comando na realização das tarefas e o terceiro grupo, o grupo de controlo (GC), composto por quatro elementos utilizou o rato.

### 5.3 Metodologia

1. Antes da realização dos testes, os utilizadores dos grupos 1 e 2 tiveram 5 minutos de ambientação.

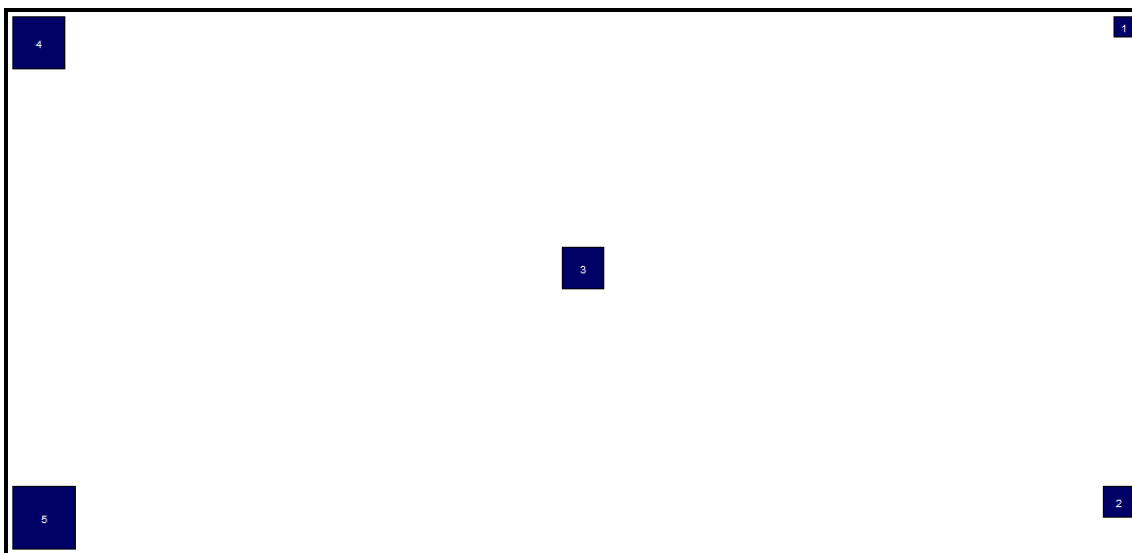


2. Seguidamente foi efectuada uma breve descrição do propósito dos testes e estes foram realizados. Durante a realização dos mesmos, foi feita a captura do ecrã em formato vídeo.
3. Após a realização dos testes foi fornecido um inquérito ao utilizador para que este pudesse avaliar o sistema e apresentar sugestões.

## 5.4 Tarefas

Os testes consistiram em quatro tarefas, todas realizadas no *browser* Firefox.

A tarefa 1 (T1) consiste em acertar em cinco quadrados de diferentes tamanhos, dispostos aleatoriamente como é possível constatar na Figura 16. Os utilizadores do G1 clicaram em cima dos quadrados por ordem crescente de tamanho e os do G2 por ordem decrescente. Este teste permitiu avaliar qual a ordem que permite uma mais rápida aprendizagem.



**Figura 16 – Imagem da T1**

A tarefa 2 (T2) consiste em preencher alguns parâmetros de um formulário composto por diversos elementos como caixas de selecção, listas e botões, como disposto na Figura 17. Esta tarefa serviria para verificar a precisão dos utilizadores quando deparados com formulários cujos elementos requerem uma maior precisão. Os 3 grupos realizaram as tarefas de forma igual.

E-MAIL *	UTILIZADOR *	NOME *	
<input type="text"/>	<input type="text"/>	<input type="text"/>	
PASSWORD *	REPITA PASSWORD *	DATA DE NASCIMENTO *	SEXO *
<input type="text"/>	<input type="text"/>	<input type="text"/> <input type="text"/> <input type="text"/>	M <input type="button" value="v"/>
PERGUNTA DE RECUPERAÇÃO *		RESPOSTA DE RECUPERAÇÃO *	
<input type="text"/>		<input type="text"/>	
MORADA	LOCALIDADE	PAÍS	
<input type="text"/>	<input type="text"/>	Portugal <input type="button" value="v"/>	
TELEFONE	PROFISSÃO	DESPORTO FAVORITO	
<input type="text"/>	Escolha <input type="button" value="v"/>	Escolha <input type="button" value="v"/>	
CLUBE	SÓCIO?	FREQUÊNCIA NOS COSTUMA VISITAR	
- Outro - <input type="button" value="v"/>	<input type="checkbox"/>	Escolha <input type="button" value="v"/>	
<input type="checkbox"/> Autorizo que os meus dados sejam facultados a terceiros.			
<input type="button" value="REGISTAR"/>			

Figura 17 – Imagem da T2. Formulário extraído da página de registo do sítio <http://www.abola.pt>.

A tarefa 3 (T3) e tarefa 4 (T4) são similares. Ambas têm como objectivo a navegação por algumas hiperligações com origem em <http://www.estig.ipbeja.pt> até se atingir um destino estipulado. Mais uma vez foi feita a divisão do grupo: os utilizadores do G1 realizaram as tarefas utilizando o clique esquerdo normal, enquanto os utilizadores do G2 realizaram as tarefas utilizando o clique esquerdo contínuo. O objectivo destas duas tarefas é concluir qual dos dois cliques é mais eficaz e eficiente.

Em apêndice está disponível o guião completo das tarefas executadas.

## 5.5 Testes e medidas

É importante referir a dificuldade de conseguir que os testes corressem de forma perfeita e estes não fugiram à regra. Durante a realização dos testes com os 2 últimos participantes de cada grupo, considerados utilizadores com menos experiência, por lapso as gravações dos testes não foram feitas de forma correcta impossibilitando a contagem dos resultados destes 2 utilizadores. Como tal, em vez dos 5 utilizadores nos G1 e G2 apenas foram contabilizados os resultados de 4 utilizadores em cada grupo.

Na Tabela 3 é possível observar a média e o desvio padrão dos resultados dos testes com os três grupos.

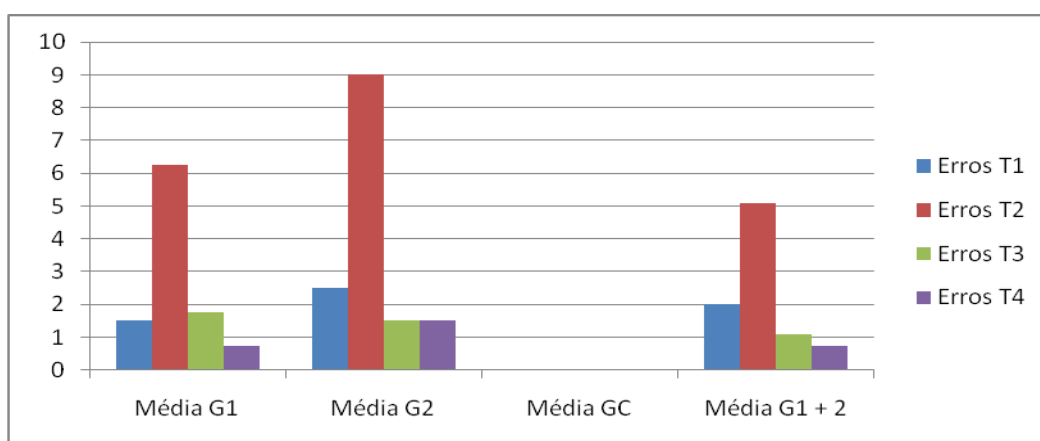
	Erros T1	Tempo T1	Erros T2	Tempo T2	Erros T3	Tempo T3	Erros T4	Tempo T4
<b>Média G1</b>	1,50	31,50	6,25	182,25	1,75	46,00	0,75	32,50
<b>Média G2</b>	2,50	54,50	9,00	270,25	1,50	65,75	1,50	60,00
<b>Média GC</b>	0,00	17,33	0,00	43,67	0,00	18,00	0,00	13,67
<b>Média G1 + 2</b>	2,00	34,44	5,08	165,39	1,08	43,25	0,75	35,39
<b>Desvio Padrão G1</b>	0,58	6,86	4,03	80,58	1,50	15,30	0,50	16,94
<b>Desvio Padrão G2</b>	2,65	46,67	3,56	347,86	1,73	20,56	1,73	40,52
<b>Desvio Padrão GC</b>	0,00	11,02	0,00	23,82	0,00	1,73	0,00	1,53
<b>Desvio Padrão G1 + 2</b>	0,71	16,26	1,94	62,23	0,18	13,97	0,53	19,45

**Tabela 3 – Tabela com resultados dos testes de todos os grupos. Tempo é apresentado em segundos.**

Em apêndice encontram-se as tabelas com os resultados individuais de cada grupo.

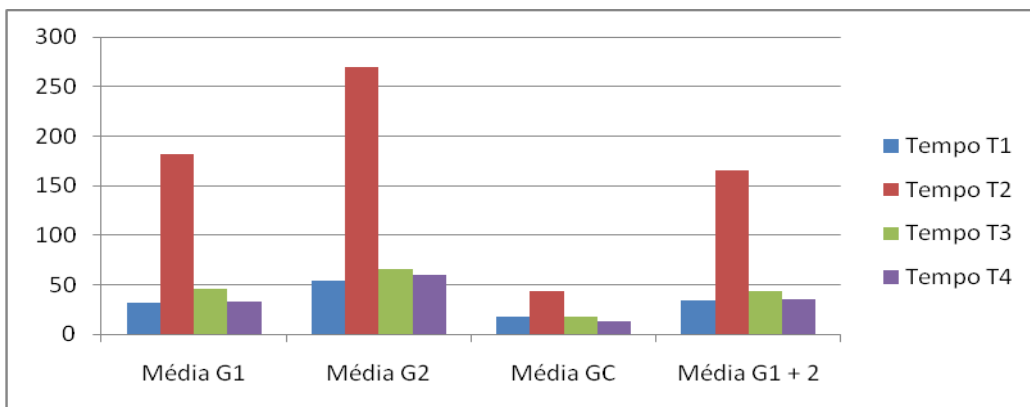
Em seguida serão mostrados os gráficos gerados pelos dados resultantes da tabela. Haverá uma breve introdução aos mesmos e a análise mais profunda a estes resultados será feita no ponto 5.6 – Análise dos Resultados.

No Gráfico 1 - Média de erros cometidos pelos vários grupos na realização das tarefas. Foi calculada a média dos grupos 1 + 2 para comparação com o grupo de controlo. Gráfico 1 pode ser vista a média dos erros cometidos, em cada tarefa, por cada grupo, sendo que é também mostrada a média conjunta do G1 e G2.



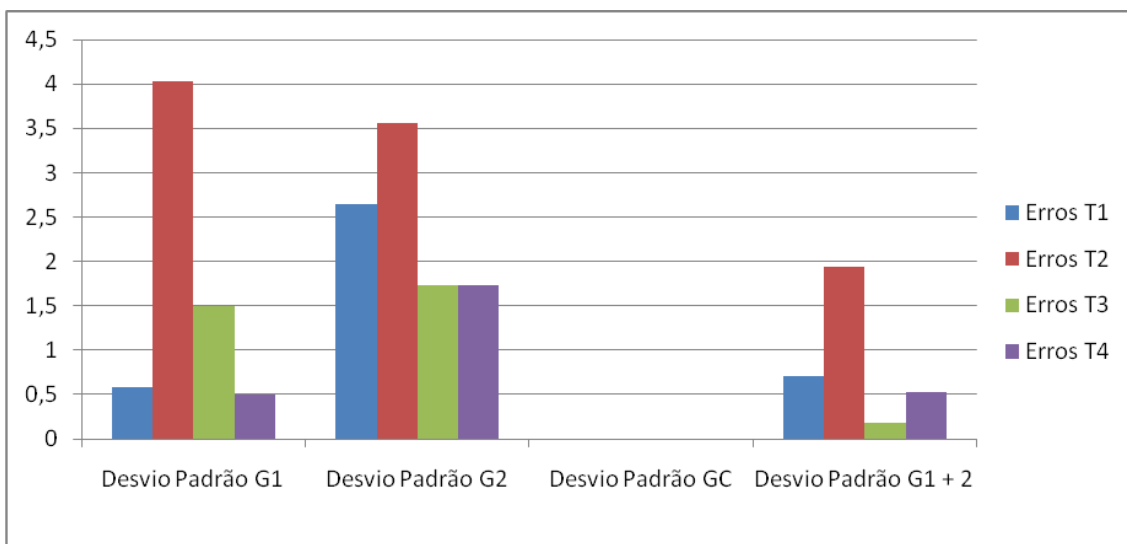
**Gráfico 1 - Média de erros cometidos pelos vários grupos na realização das tarefas. Foi calculada a média dos grupos 1 + 2 para comparação com o grupo de controlo.**

Observando o Gráfico 2 pode-se verificar a média do tempo (em segundos) demorado por cada grupo para concluir cada tarefa. Adicionalmente, é mostrada uma média conjunta do G1+2.



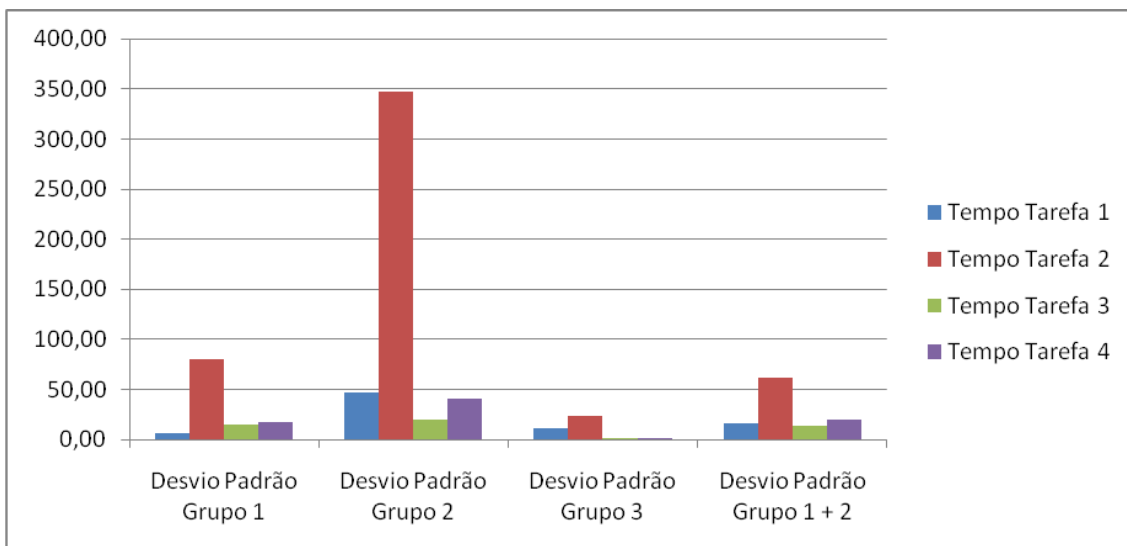
**Gráfico 2 - Média de tempos realizados pelos vários grupos na realização das tarefas. Foi calculada a média dos grupos 1 + 2 para comparação com o grupo de controlo.**

No Gráfico 3 é possível consultar o desvio padrão de cada grupo bem como o do G1+2. Isto, para os erros cometidos em cada tarefa.



**Gráfico 3 - Desvio padrão dos erros cometidos pelos vários grupos na realização das tarefas. Foi calculado o desvio padrão dos grupos 1 + 2 para comparação com o grupo de controlo.**

No Gráfico 4 é possível consultar o desvio padrão de cada grupo bem como o do G1+2. Isto, para o tempo demorado (em segundos) para cada tarefa.



**Gráfico 4 - Desvio padrão dos tempos realizados pelos vários grupos na realização das tarefas. Foi calculado o desvio padrão dos grupos 1 + 2 para comparação com o grupo de controlo.**

## 5.6 Análise dos resultados

Na análise dos resultados foram consideradas duas análises, a análise com base nas anotações provenientes da observação durante a fase de ambientação e a análise estatística com base no tempo realizado e nos erros cometidos.

### 5.6.1 Análise à fase de ambientação

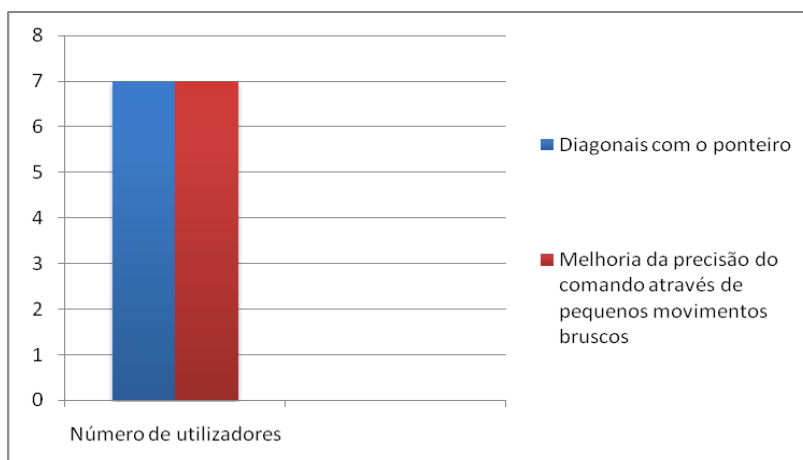
Verificou-se que cada utilizador se ambientava de forma diferente ao comando. Enquanto uns apenas movimentavam o cursor pelo ambiente de trabalho, outros foram mais pró-activos e abriram algumas aplicações, como por exemplo o jogo Solitário. Outros utilizadores abriram alguns documentos e directorias.

Tendo conhecimento de que pequenos movimentos bruscos aumentam bastante a precisão pretendia-se saber a rapidez de aprendizagem desses movimentos de forma autónoma. Além disso pretendeu-se saber se num curto espaço de tempo os utilizadores aprendem a fazer diagonais da mesma forma que é efectuado o movimento do rato. Na tabela 6 é possível observar o número de utilizadores que realizaram os movimentos de forma autónoma:

Movimento	Número de utilizadores
Diagonais com o ponteiro	7
Melhoria da precisão do comando através de pequenos movimentos bruscos	7

**Tabela 6 – Número de utilizadores que realizaram movimentos esperados de forma autónoma.**

Com base na tabela 6 e no gráfico 5 é possível observar que 7 em 8 dos utilizadores, ou seja 87,5%, apreenderam os esperados movimentos de forma autónoma em apenas 5 minutos. Estes dados permitem concluir que a auto-aprendizagem destes movimentos foi relativamente rápida.



**Gráfico 5 – Número de utilizadores que realizaram movimentos esperados de forma autónoma.**

Os pequenos movimentos bruscos permitiram aos utilizadores aumentar a precisão e as diagonais ganhar eficiência, na medida em que permitem poupar tempo.

Não obstante, quando os testes se iniciaram, mesmo alguns utilizadores que tinham revelado algumas facilidades em termos de precisão na ambientação, revelaram depois algumas dificuldades nesse campo, nomeadamente quando tentavam acertar no quadrado mais pequeno durante a primeira tarefa, ou na segunda tarefa quando tentavam acertar nos botões de selecção. Estas dificuldades indicam que poderá ser necessário mais tempo para aprendizagem.

### **5.6.2 Análise à fase de testes**

Como referido na caracterização dos participantes, os que utilizaram o comando foram divididos em dois grupos. Inicialmente será feita a comparação entre os dois primeiros grupos e posteriormente destes dois conjuntamente com o grupo 3.

O grupo 1 realizou a primeira tarefa clicando nos quadrados de forma crescente, enquanto o grupo 2 clicou nos quadrados de forma decrescente. Verificou-se em média que o grupo 1 demorou menos tempo e cometeu menos erros que o grupo 2, cometendo menos 40% erros e demorou menos 42,2 % a realizar as tarefas. Estes valores indicam que o método de aprendizagem utilizado pelo grupo 1 é mais eficaz e mais eficiente.

Dentro do grupo 1 não se verifica praticamente nenhum desvio da média nos erros cometidos e um pequeno desvio no tempo realizado, estes valores do desvio padrão revelam fiabilidade nos valores obtidos. No grupo 2 existe um desvio padrão superior ao grupo 1, embora reduzido, nos

erros cometidos e um desvio padrão elevado (46,67) no tempo realizado. Este desvio no tempo realizado deixa incerteza quanto aos dados deste grupo.

A tarefa 2 foi realizada pelos dois grupos de forma igual e como tal não se considera importante a comparação entre ambos nesta fase da análise. Fica a nota no entanto para o valor muito elevado do desvio padrão registado pelo grupo 2, em particular pelo utilizador 4. Esta tarefa é considerada de dificuldade elevada devido a ter vários elementos Web de pequenas dimensões como *checkboxes* e *comboboxes*. Foi nesta tarefa que se verificou o maior número de erros, indicando que para uma utilização que requer maior precisão há a necessidade de um tempo maior de aprendizagem.

Na realização das tarefas 3 e 4, o grupo 1 usou a função clique esquerdo normal e o grupo 2 utilizou o clique esquerdo contínuo. Apesar de estas tarefas terem diferentes objectivos em termos de guião, ambas visam avaliar a mesma funcionalidade e portanto serão analisadas em conjunto.

Na tarefa 3, o grupo 1 cometeu menos 16% erros que o grupo 2, mas demorou mais 43% a realizar a tarefa. Esta demora poderá estar relacionada com uma necessidade inicial de posicionar a seta do rato antes de passar pela hiperligação. Esta dificuldade foi relatada por alguns dos participantes que utilizaram o clique esquerdo contínuo, ou seja, necessita de mais tempo para aprendizagem. Na realização desta tarefa verificou-se um desvio padrão elevado mas aproximado no tempo realizado, este desvio pode ser responsabilizado pela heterogeneidade dos grupos, revelando alguma incerteza nos resultados obtidos.

Verificou-se um valor reduzido desvio padrão nos erros cometidos, o que revela fiabilidade nos testes.

Na execução da tarefa 4, o grupo 2 cometeu o dobro dos erros, no entanto é importante registar que em média esse valor foi de apenas 1,5 erros. O grupo 2 demorou cerca de 85% mais que o grupo 1, este valor foi afectado pelo tempo que os utilizadores demoram em recuperar dos erros. Questionando a sua utilização de uma forma mais frequente.

Considerando os valores das médias dos dois grupos, podemos concluir que nesta tarefa, que se considera ser de pequena/média dificuldade, ambos os sistemas têm uma eficácia semelhante, no entanto o clique esquerdo contínuo revela ter uma menor eficiência.

Foi feita a junção dos dados dos grupos 1 e 2 para comparar os resultados dos grupos que utilizaram o comando com o grupo 3. Desta forma será possível medir a eficácia e eficiência do sistema em relação ao rato.

Na tarefa 1, o grupo 1 + 2 cometeu 2 erros e demoraram cerca do dobro do tempo em relação ao grupo 3 que não cometeu nenhum erro. Nas tarefas realizadas pelos utilizadores do grupo 1 + 2 o desvio padrão aproxima-se de zero, indicando que todos, com muitos ou poucos conhecimentos informáticos, cometeram os mesmos erros e que os resultados do teste são fiáveis. Conclui-se que o sistema foi eficaz e de fácil aprendizagem na primeira tarefa por apenas terem sido cometidos em média 2 erros. Em relação ao tempo realizado, o valor do desvio é semelhante, provando também que o grupo 3 é heterogéneo.

Na tarefa 2 foi possível observar - como já tinha sido na comparação do grupo 1 e 2 - um aumento nos erros cometidos, passando a média do grupo 1 + 2 para cerca de 5. Nesta tarefa o grupo 3 continuou a não cometer erros e demorou 3,8 vezes menos tempo. É importante lembrar que esta tarefa é de elevada dificuldade e exige mais tempo de aprendizagem. Como tal considera-se que o sistema foi relativamente eficaz e eficiente, mas que pode ser muito melhorado se for dado mais tempo para praticar aos utilizadores.

Na realização das tarefas 3 e 4 verificou-se em média que o grupo 1 + 2 apenas cometeu um erro, enquanto o grupo 3 continuou sem cometer erros. Estes valores revelam a eficácia do sistema, embora os utilizadores que utilizaram o comando tenham demorado 2,58 vezes mais tempo. Os valores do desvio padrão dos utilizadores do rato transmitem confiança. Verifica-se no entanto alguma incerteza, não muita, na fiabilidade dos testes dos utilizadores do comando.

Em suma, conclui-se que o sistema é eficaz e eficiente, necessitando apenas de mais prática por parte dos utilizadores. Nestes testes os utilizadores do comando apenas tiveram 5 minutos de ambientação, enquanto os utilizadores do rato contam com um número bastante elevado de horas de treino.

Não se pode contudo ignorar os valores altos do desvio padrão. Esta situação alerta para a necessidade da existência de grupos maiores para a realização dos testes e diminuição da incerteza.





## Conclusão e trabalho futuro

O trabalho desenvolvido teve como objectivo a criação de uma aplicação que, de modo eficaz e eficiente, permitisse a ligação de um *Wii Remote* a um computador pessoal de uma forma suficientemente simples, de modo que qualquer utilizador comum o conseguisse usar para navegar na Web.

Em termos de funcionalidades inicialmente propostas, todas a nível de funções do rato foram implementadas com sucesso.

Relativamente às funcionalidades do teclado, nem todas foram introduzidas, nomeadamente a função que iria permitir a escrita de texto, o *page up* e o *page down*. No entanto, foram introduzidas muitas outras, como o caso da movimentação (*scroll*) para cima e para baixo em substituição dos já referidos *page up* e *down*, e das funções de *home* e *end*. Sendo a função de escrita de texto uma mais-valia a ser implementada futuramente.

Sobre as funcionalidades de *browser*, todas foram implementadas com sucesso.

No que concerne às funções no *Wii Remote*, não só foi introduzida a possibilidade de configurar botões, como é possível manter várias configurações guardadas, graças à implementação de uma função que permite gravar e ler ficheiros de configuração.

Abordando os requisitos não funcionais, aqueles considerados operacionais não foram concluídos com sucesso, pois, esta aplicação só funciona em sistemas operativos Microsoft Windows, sendo, claramente, algo a melhorar no futuro.

Em termos de segurança, apenas é possível ligar um só comando, como proposto. Não houve necessidade de implementação deste ponto dado que a própria biblioteca *WiiRemoteJ* já o faz. Certas funcionalidades adicionais permitiram aumentar a robustez e segurança do sistema, como no caso dos ficheiros de configuração da aplicação serem apagados, a própria aplicação, ao ser executada irá criá-los novamente. Outras situações foram tidas em conta, como o caso de apenas ser possível ter uma instância da aplicação aberta ao mesmo tempo de forma a evitar erros e conflitos na ligação com o computador.

Relativamente ao desempenho e apesar de existir alguma incerteza causada pelos valores do desvio padrão nos testes de utilizadores, considera-se que existe precisão e o tempo de resposta aproxima-se do rato.

Foi possível concluir com os inquéritos dos utilizadores que os mesmos consideram este sistema, intuitivo e de fácil utilização, muito embora seja necessário algum tempo de aprendizagem.

Juntamente com a aplicação encontra-se um ficheiro de ajuda que contém toda a informação necessária para um correcto uso do sistema. Incluído nele está, por exemplo, detalhes específicos sobre a configuração original de botões e o seu propósito, a indicação de como manobrar o comando, bem como o calibrar e ainda informação sobre como alterar a configuração original de botões e caso seja necessário, revertê-la à predefinição.

Sobre a compatibilidade do sistema com vários adaptadores *Bluetooth*, não foi cumprida devido às limitações da biblioteca BlueCove, sendo este um dos pontos que também deverá de ser resolvido numa versão futura.

Entende-se que o principal desafio é a precisão do cursor do rato. Apesar dos resultados promissores nos testes de utilizador, consideramos que será sempre a principal prioridade para versões futuras.

Considera-se que o sistema é muito funcional, existindo utilizadores que revelaram entusiasmo em poder utilizar o sistema num ambiente de apresentações, aquele para o qual também foi concebido.

Concluindo, o sistema foi implementado com êxito, com várias funcionalidades adicionais e apesar de existir espaço para melhorias, o resultado final é sem dúvida muito satisfatório.

# Apêndice(s)

## Apêndice 1 – Diagrama de Casos de Uso

A Figura 18 contém o diagrama de casos de uso elaborado no decorrer deste trabalho.

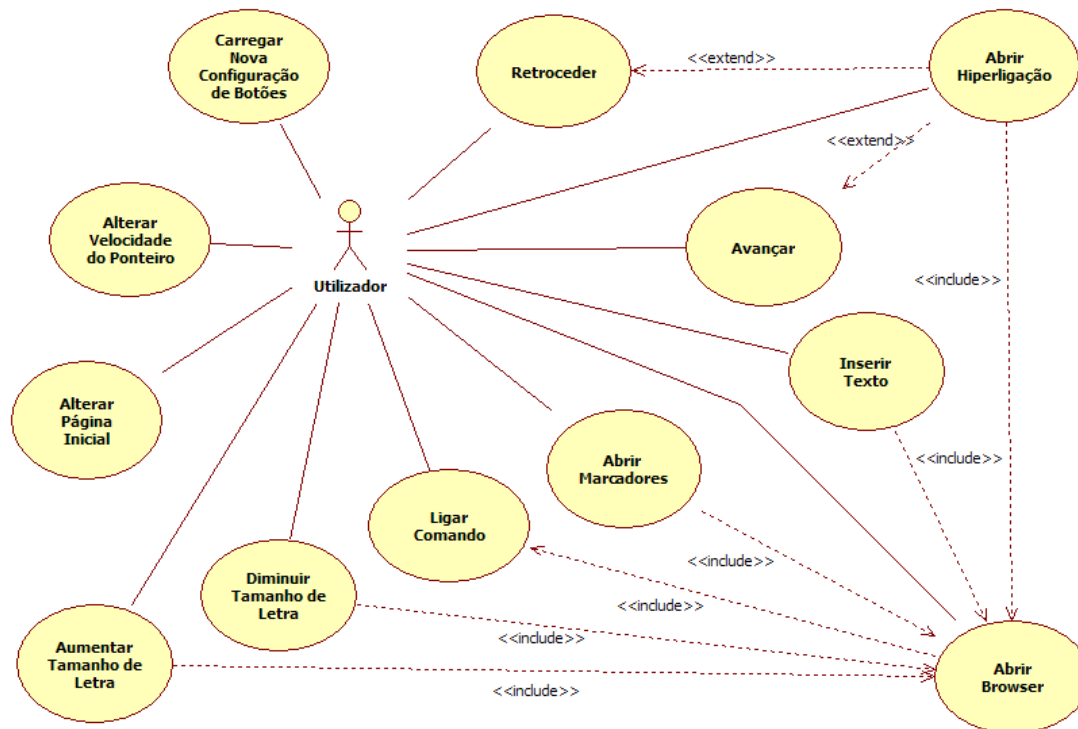


Figura 18 – Diagrama de casos de uso gerado com o programa StarUML.

## Apêndice 2 – Documento de especificação

Seguidamente são apresentados os documentos de especificação elaborados na sequência do diagrama de casos de uso da Figura 18.

Nome	<b>Ligação entre o Wiimote e o computador.</b>
Descrição	Utilizador efectua a ligação entre o <i>Wiimote</i> e o computador.
Pré-condições	<i>Bluetooth</i> estar activo.
Pós-condições	Utilizador fica com o <i>Wiimote</i> ligado ao computador e pronto a funcionar.
Situações de falha	Falha na ligação entre o <i>Wiimote</i> e o computador, <i>Wiimote</i> não ser reconhecido.
Actores	Utilizador.

Cenário principal	(1) Iniciar a aplicação, que faz o sistema iniciar a detecção do <i>Wiimote</i> ; (2) Pressionar os botões 1 e 2 no <i>Wiimote</i> para que o mesmo seja reconhecido pelo sistema; (3) Quando sistema reconhecer o <i>Wiimote</i> , o mesmo ficará sincronizado no sistema e pronto a funcionar.
Extensões ou variações	-

Nome	<b>Aceder ao moodle com autenticação efectuada.</b>
Descrição	Utilizador consegue abrir o browser no sítio web do moodle e autenticar.
Pré-condições	Ligação entre o <i>Wiimote</i> e o computador efectuada com êxito.
Pós-condições	Utilizador com acesso ao conteúdo do moodle de acordo com os seus privilégios de utilizador.
Situações de falha	Falha na ligação entre o <i>Wiimote</i> e o computador. Moodle indisponível.
Actores	Utilizador.
Cenário principal	(1) Utilizador pressiona botão <i>home</i> no <i>Wiimote</i> para que browser seja aberto; (2) Utilizador acede à página do moodle; (3) Utilizador insere os dados de utilizador para que sistema valide a sessão.
Extensões ou variações	Aceder a um recurso no moodle.

Nome	<b>Abrir uma hiperligação.</b>
Descrição	Utilizador abre uma hiperligação no browser.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o <i>Wiimote</i> e o computador efectuada com êxito.
Pós-condições	Utilizador acede ao conteúdo da hiperligação.
Situações de falha	Falha na ligação entre o <i>Wiimote</i> e o computador, <i>Wiimote</i> não ser reconhecido. Hiperligação indisponível.
Actores	Utilizador.
Cenário principal	(1) Abrir o browser; (2) Abrir a hiperligação desejada; (3) Browser redirecciona o utilizador para a hiperligação desejada.
Extensões ou variações	-

Nome	<b>Inserir Texto.</b>
Descrição	Utilizador consegue escrever texto usando o comando.

Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o <i>Wiimote</i> e o computador efectuada com êxito.
Pós-condições	Texto escrito pelo utilizador fica inserido.
Situações de falha	Falha na ligação entre o <i>Wiimote</i> e o computador, <i>Wiimote</i> não ser reconhecido.
Actores	Utilizador.
Cenário principal	(1) Abrir o browser; (2) Seleccionar a caixa de texto desejada; (3) Activar a funcionalidade do comando que permite escrever texto.
Extensões ou variações	-

Nome	<b>Aceder a um recurso no moodle.</b>
Descrição	Utilizador pretende aceder a um recurso disponibilizado no moodle.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o <i>Wiimote</i> e o computador efectuada com êxito. Autenticação no moodle efectuada com êxito.
Pós-condições	Utilizador acede ao conteúdo do recurso.
Situações de falha	Falha na ligação entre o <i>Wiimote</i> e o computador, <i>Wiimote</i> não ser reconhecido. Moodle indisponível. Falhas em hiperligações ou no acesso aos conteúdos.
Actores	Utilizador.
Cenário principal	(1) Utilizador acede à disciplina pretendida; (2) Utilizador escolhe um dos recursos disponíveis; (3) Recurso é aberto pelo browser ou descarregado no PC para que se possa aceder ao seu conteúdo.
Extensões ou variações	-

Nome	<b>Guardar nova configuração de botões</b>
Descrição	Utilizador pretende guardar uma nova configuração dos botões do comando.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o comando e o computador efectuada com êxito.
Pós-condições	Utilizador guarda uma nova configuração.
Situações de falha	Falha na ligação entre o comando e o computador, comando não ser reconhecido.
Actores	Utilizador.
Cenário principal	(1) Utilizador abre a aplicação; (2) Utilizador selecciona o menu “Configurar”; (3) Utilizador escolhe a opção “Botões”;

	<p>(4) É aberta uma interface pelo sistema que possibilita a configuração dos botões;</p> <p>(5) Após as alterações, utilizador clica no botão “Guardar”;</p> <p>(6) Aplicação guarda a nova configuração, ficando a mesma pronta a utilizar.</p>
Extensões ou variações	-

Nome	<b>Carregar nova configuração de botões</b>
Descrição	Utilizador pretende carregar uma nova configuração dos botões do comando.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o comando e o computador efectuada com êxito.
Pós-condições	Utilizador usa uma nova configuração.
Situações de falha	Falha na ligação entre o comando e o computador, comando não ser reconhecido. Não haver ficheiros de configuração.
Actores	Utilizador.
Cenário principal	<p>(1) Utilizador abre a aplicação;</p> <p>(2) Utilizador selecciona o menu “Ficheiro”;</p> <p>(3) Utilizador escolhe a opção “Carregar Configuração”;</p> <p>(4) É aberta uma janela de escolha de ficheiros, onde o utilizador pode seleccionar qual o ficheiro de configuração que pretende carregar;</p> <p>(5) Após carregar o ficheiro, as novas configurações ficam prontas a utilizar.</p>
Extensões ou variações	-

Nome	<b>Alterar página inicial</b>
Descrição	Utilizador pretende alterar a página inicial aberta pelo navegador de internet.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o comando e o computador efectuada com êxito.
Pós-condições	Utilizador fica com uma nova página inicial.
Situações de falha	Falha na ligação entre o comando e o computador, comando não ser reconhecido. URL inserido pelo utilizador não ser correcto.
Actores	Utilizador.
Cenário principal	<p>(1) Utilizador abre a aplicação;</p> <p>(2) Utilizador selecciona o menu “Configurar”;</p> <p>(3) Utilizador escolhe a opção “Página Inicial”;</p> <p>(4) É aberta uma interface pelo sistema que possibilita a configuração da página inicial;</p> <p>(5) Após as alterações, utilizador clica no botão “Guardar”;</p> <p>(6) Aplicação guarda a nova página inicial, ficando a mesma pronta a utilizar.</p>

Extensões ou variações	-
------------------------	---

Nome	<b>Alterar velocidade do ponteiro</b>
Descrição	Utilizador pretende alterar a velocidade do ponteiro controlado pelo comando.
Pré-condições	<i>Bluetooth</i> estar activo, ligação entre o comando e o computador efectuada com êxito.
Pós-condições	Utilizador usa uma nova velocidade no ponteiro.
Situações de falha	Falha na ligação entre o comando e o computador, comando não ser reconhecido.
Actores	Utilizador.
Cenário principal	(1) Utilizador abre a aplicação; (2) Utilizador selecciona o menu “Configurar”; (3) Utilizador escolhe a opção “Velocidade do Ponteiro”; (4) É aberta uma interface pelo sistema que possibilita a configuração da velocidade do ponteiro; (5) Aplicação guarda a nova velocidade, ficando a mesma pronta a utilizar.
Extensões ou variações	-

## Apêndice 3 – Guião de Tarefas

Segue-se uma cópia do guião de tarefas fornecido aos utilizadores.

# Tarefa 1

## Grupo 1

Clique nos quadrados de forma crescente.

## Grupo 2

Clique nos quadrados de forma decrescente.

## Grupo de Controlo

Clique nos quadrados de forma crescente.



## Tarefa 2

1. No campo SEXO seleccione F;
2. No campo PAÍS seleccione MOÇAMBIQUE;
3. No campo PROFISSÃO seleccione ESTUDANTE;
4. No campo DESPORTO FAVORITO seleccione FUTEBOL;
5. No campo CLUBE seleccione BENFICA;
6. Active a caixa de selecção SÓCIO;
7. No campo FREQUÊNCIA NOS COSTUMA VISITAR seleccione QUASE TODOS OS DIAS;
8. Active a caixa de selecção AUTORIZO QUE OS MEUS DADOS SEJAM FACULTADOS A TERCEIROS;
9. Clique em REGISTAR.

## Tarefa 3

### Grupo 1

Usando apenas a função de clique esquerdo, clique na hiperligação CONTACTOS e em seguida DOCENTES. Finalmente, escolha o docente MARGARIDA ISABEL MATOS RAMOS MARTINS DOS SANTOS.

### Grupo 2

Usando apenas a função de clique esquerdo contínuo, clique na hiperligação CONTACTOS e em seguida DOCENTES. Finalmente, escolha o docente MARGARIDA ISABEL MATOS RAMOS MARTINS DOS SANTOS.

### Grupo de Controlo

Clique na hiperligação CONTACTOS e em seguida DOCENTES. Finalmente, escolha o docente MARGARIDA ISABEL MATOS RAMOS MARTINS DOS SANTOS.

# Tarefa 4

## Grupo 1

Usando apenas a função de clique esquerdo, clique na hiperligação ALUNOS e em seguida HORÁRIOS. Escolha HORÁRIOS DE ENGENHARIA DE INFORMÁTICA DO 1º SEMESTRE e finalmente, escolha 4º ANO NOCTURNO.

## Grupo 2

Usando apenas a função de clique esquerdo contínuo, clique na hiperligação ALUNOS e em seguida HORÁRIOS. Escolha HORÁRIOS DE ENGENHARIA DE INFORMÁTICA DO 1º SEMESTRE e finalmente, escolha 4º ANO NOCTURNO.

## Grupo de Controlo

Clique na hiperligação ALUNOS e em seguida HORÁRIOS. Escolha HORÁRIOS DE ENGENHARIA DE INFORMÁTICA DO 1º SEMESTRE e finalmente, escolha 4º ANO NOCTURNO.

## Apêndice 4 – Resultados dos testes de utilizadores

A tabela 4 apresenta os resultados específicos de cada utilizador do grupo 1 ao realizar as tarefas.

Grupo 1	Erros Tarefa 1	Tempo Tarefa 1	Erros Tarefa 2	Tempo Tarefa 2	Erros Tarefa 3	Tempo Tarefa 3	Erros Tarefa 4	Tempo Tarefa 4
Utilizador 1	1	27	4	115	3	31	1	24
Utilizador 2	2	32	8	254	3	67	1	19
Utilizador 3	1	41	11	110	0	40	0	30
Utilizador 4	2	26	2	250	1	46	1	57

**Tabela 4 – Resultados do grupo 1.**

A tabela 5 apresenta os resultados específicos de cada utilizador do grupo 2 ao realizar as tarefas.

Grupo 2	Erros Tarefa 1	Tempo Tarefa 1	Erros Tarefa 2	Tempo Tarefa 2	Erros Tarefa 3	Tempo Tarefa 3	Erros Tarefa 4	Tempo Tarefa 4
Utilizador 1	6	39	7	100	4	80	0	27
Utilizador 2	1	29	6	90	1	43	1	50
Utilizador 3	3	26	9	99	0	54	4	119
Utilizador 4	0	124	14	792	1	86	1	44

**Tabela 5 – Resultados do grupo 2.**

A tabela 6 apresenta os resultados específicos de cada utilizador do grupo de controlo ao realizar as tarefas.

Grupo de Controlo	Erros Tarefa 1	Tempo Tarefa 1	Erros Tarefa 2	Tempo Tarefa 2	Erros Tarefa 3	Tempo Tarefa 3	Erros Tarefa 4	Tempo Tarefa 4
Utilizador 1	0	30	0	54	0	19	0	15
Utilizador 2	0	10	0	46	0	19	0	14
Utilizador 3	0	12	0	31	0	16	0	12

**Tabela 6 – Resultados do grupo de controlo.**

## Apêndice 5 – Inquérito aos utilizadores

Segue-se uma cópia do inquérito fornecido aos utilizadores.

1. Descreva os seus conhecimentos informáticos:

- ☐ Baixo (óptica de utilizador)
- ☐ Médio (instalação e configuração de sistemas operativos)
- ☐ Alto (conhecimentos de programação)

2. Como avalia a dificuldade de aprendizagem:

- ☐ Baixa
- ☐ Média
- ☐ Alta

Sugestão:

3. Como avalia o grau de precisão:

- ☐ Baixo
- ☐ Médio
- ☐ Alto

Sugestão:

**Nota:** Caso tenha experimentado o clique esquerdo contínuo responda à pergunta 4, caso contrário avance para a pergunta 5.

4. Qual o método que prefere para abrir hiperligações:

- ☐ Clique esquerdo normal
- ☐ Clique esquerdo contínuo

Justifique

5. Qual o seu grau de satisfação?

- ☐ Baixo
- ☐ Médio
- ☐ Alto

Justifique

6. Na sua opinião o comando seria útil numa apresentação?

- ☐ Sim
- ☐ Não

Justifique



# Bibliografia

- [1] Mozilla, “Firefox”, disponível em <http://www.mozilla-europe.org/pt/firefox/> (consultado em 24/09/2009)
- [2] Nintendo, “Wii”, disponível em <http://wii.com/> (consultado em 24/09/2009)
- [3] Eclipse Foundation, Inc., “Eclipse IDE for Java Developers”, disponível em <http://www.eclipse.org/> (consultado em 24/09/2009)
- [4] “Cha0s”, “WiiRemoteJ”, disponível em <http://www.world-of-cha0s.hostrocket.com/WiiRemoteJ/> (consultado em 04/11/2009)
- [5] BlueCove Team, “BlueCove”, disponível em <http://bluecove.org/> (consultado em 24/09/2009)
- [6] Fritzsche, Volker, “motej”, disponível em <http://motej.sourceforge.net/> (consultado em 04/11/2009)
- [7] Duche, Guilhem, “WiiuseJ”, disponível em <http://code.google.com/p/wiiusej/> (consultado em 04/11/2009)
- [8] Laforest, Michael, “wiiuse”, disponível em <http://www.wiiuse.net/> (consultado em 04/11/2009)
- [9] gl.tter, “Wiiyourself!”, disponível em <http://wiiyourself.gl.tter.org/> (consultado em 04/11/2009)
- [10] Holtmann, Marcel; Hedberg, Johan, “BlueZ”, disponível em <http://www.bluez.org/> (consultado em 04/11/09)
- [11] s.a., “WiiLi”, disponível em [http://www.wiili.com/index.php/Compatible\\_Bluetooth\\_Devices](http://www.wiili.com/index.php/Compatible_Bluetooth_Devices) (consultado em 04/11/2009)
- [12] s.a., “Wii Remote”, disponível em [http://en.wikipedia.org/wiki/Wii\\_Remote#Design](http://en.wikipedia.org/wiki/Wii_Remote#Design) (consultado em 11/11/2009)
- [13] s.a., “WiiLi”, disponível em <http://www.wiili.com/index.php/Wiimote/Hardware> (consultado em 06/11/2009)
- [14] Chen, Jason, “Wii MotionPlus Review”, disponível em <http://gizmodo.com/5326510/wii-motionplus-review> (consultado em 24/09/09)

- [15] oniGamemaster, “USB Sensor Bar”, disponível em <http://www.instructables.com/id/USB-Sensor-Bar/> (consultado em 24/09/09)
- [16] s.a., “StarUML”, disponível em <http://staruml.sourceforge.net/en/> (consulta em 06/11/2009)

# Agradecimentos

Eu, Boris Júnior, quero primeiramente agradecer aos meus pais Jorge António Caldeira Júnior e Maria de Lourdes Caldeira Júnior por todo o esforço, dedicação e sobretudo paciência que tiveram comigo ao longo destes anos, bem como pelos valores e pela educação que me proporcionaram. Sem eles não estaria aqui e só com o apoio deles é que me foi possível alcançar este e outros objectivos.

Agradeço também, a paciência, o apoio e a amizade do meu colega e sobretudo amigo Jorge Pereira, com o qual comecei a fazer todos os trabalhos de grupo, partindo do ponto em que perguntávamos um ao outro se fazíamos um dado trabalho em conjunto, até ao ponto em que simplesmente dizíamos a hora e o local de estudo. Foram muitas, longas, horas passadas em frente a computadores, muitas vezes sob a pressão dos prazos e sempre nos apoiamos mutuamente para atingirmos os nossos objectivos e ao longo desse tempo todo, um laço foi-se formando, uma forte amizade que ficará para sempre.

Eu, Jorge Pereira, quero novamente agradecer aos meus pais, José e Mariana, pelos sacrifícios e pela sua compreensão quando não estive presente em alguns momentos. Pelo exemplo que são para mim enquanto pessoas e por tudo aquilo que só mesmo os pais fazem e que os torna especiais.

Quero agradecer à minha irmã Maria José, que sempre me guiou e sempre foi um verdadeiro exemplo por tudo o que conquistou e me ensinou. Ao meu cunhado Vítor pelo seu apoio, incentivo e conselhos e pela sua ajuda em situações extra-académicas que me permitiram ter mais tempo livre. Aos meus sobrinhos dizer que por eles senti mais responsabilidade e obrigação de ser um exemplo.

Agradeço à minha namorada Sofia pelo apoio que sempre me deu nas alturas em que o cansaço me parecia vencer e pela sua paciência e compreensão pela minha ausência. Quero ser para ela, e espero tê-lo sido até agora, um exemplo.

Agradecer ao meu colega, e sobretudo amigo, Boris Júnior por me ter ajudado ao longo deste percurso difícil. Juntos conseguimos tornar o curso fácil.

Quero pedir desculpa aos meus amigos por não ter estado mais tempo com eles durante estes anos de sacrifício.



Em nome de ambos, queremos também agradecer todo o corpo docente da ESTIG por contribuírem para a nossa formação académica e em especial os nossos orientadores João Paulo Barros e Luís Bruno pela disponibilidade e por todo o apoio prestado durante o decorrer deste trabalho.

Pretendemos também deixar um agradecimento aos nossos colegas e amigos que nos acompanharam ao longo deste percurso e em especial neste último ano como o caso dos colegas Andreia Graça, Joaquim Gomes, Manuel Carapinha, Nelso Ventura, Nuno Varela, Paula Gracinda, Paulo Grade, Paulo Zarcos, Rui Cavaco, Simão Cá, entre muitos outros que são agora demasiados para enumerar.

Finalmente, queremos agradecer todos os elementos do grupo de utilizadores que testou o nosso sistema despendendo do seu tempo livre e pela paciência demonstrada quando as coisas não corriam como planeado, são eles: Ana César, Andreia Graça, Andreia Guerreiro, Ângela Ganhão Jaca, Carlos Lourenço, João Aguiar, Luís Ferro, Marlene Soares, Pedro Imaginário, Rodrigo Aires, Sofia Porto, Tiago Pereira e Vanessa Conceição.