Escalabilidade do jogo WebRun: Versão Cliente-Servidor versus Versão Peer-to-Peer

Paula Prata
Instituto de Telecomunicações
Universidade da Beira Interior
Covilhã, Portugal
pprata@di.ubi.pt

Rui Cunha
Instituto de Telecomunicações
Universidade da Beira Interior
Covilhã, Portugal
ruimfcunha@gmail.com

Pedro Araújo
Instituto de Telecomunicações
Universidade da Beira Interior
Covilhã, Portugal
paraujo@di.ubi.pt

ABSTRACT
Os jogos *online* são cada vez mais importantes, representando um mercado que envolve biliões de dólares. A maioria dos jogos segue um modelo cliente-servidor, arquitetura que facilita o desenvolvimento e o controlo do jogo, por exemplo, em termos de evitar a utilização indevida por jogadores mal-intencionados. Esta arquitetura tem, no entanto, problemas em termos de escalabilidade. Com o aumento do número de utilizadores, diminui a qualidade do serviço do ponto de vista dos jogadores. Neste artigo propomos uma arquitetura *peer-to-peer* para um jogo que permite promover a atividade física e que inicialmente foi desenvolvido como cliente-servidor. Apresenta-se um estudo do desempenho das duas versões do jogo quando o número de jogadores aumenta. Os resultados obtidos mostram que a arquitetura peer-to-peer permite aumentar significativamente a escalabilidade do jogo, sendo uma alternativa promissora para áreas em que o problema da fraude no jogo não seja crítico.

Author Keywords
Exergames; *peer-to-peer*; cliente-servidor, escalabilidade.

ACM Classification Keywords
C.4. Performance of Systems. K.8.0. General Games.

INTRODUCÃO

Os jogos *online* multiutilizadores enfrentam a questão de como tornar escalável a arquitetura centralizada, tipo cliente-servidor (C/S), na qual é baseada a maioria dos jogos. Para enfrentar o crescente número de utilizadores deste tipo de jogos, a arquitetura *peer-to-peer* (P2P) é cada vez mais objeto de investigação.

O jogo WebRun é um jogo *online* de arquitetura C/S que pretende promover a atividade física, no conforto da própria casa, tornando-a um divertimento [1]. A partir do protótipo deste jogo disponibilizado pelos seus autores, foi criada uma nova versão seguindo uma arquitetura P2P e estudou-se o desempenho das duas versões quando o número de utilizadores aumenta.

A ideia principal do jogo consiste em permitir a realização de corridas de bicicleta em grupo usando bicicletas de manutenção fixas e cada jogador poder visualizar num ecrã a sua evolução no percurso assim como a evolução dos jogadores que aceitaram correr com ele. Cada jogador deve ter acesso a um computador onde está instalado o cliente do jogo e simultaneamente dispor de uma bicicleta fixa, à qual está associada uma unidade de controlo que recolhe da bicicleta a velocidade com que o jogador pedala. Esta recolha pode ser feita através de sinais disponibilizados pela maioria das bicicletas existentes ou pela colocação de sensores externos.

Cada jogador que propõe uma corrida, escolhe num mapa o percurso que pretende simular, convida outros jogadores a participar na corrida, e à medida que a corrida decorre os vários jogadores podem visualizar a evolução dos jogadores do grupo no percurso.

A versão P2P implementada neste trabalho, é uma versão híbrida, em que o jogo é C/S na fase de autenticação, mas mais tarde, durante o jogo, o servidor de uma nova corrrida passará a ser o jogador que propõs a corrida.

O artigo começa por apresentar brevemente a área dos jogos que promovem a atividade física, seguida do background em arquiteturas de jogos *online* e desafios associados. De seguida, descreve-se a atual implementação C/S do jogo WebRun e apresenta-se a interface para aquisição de dados da bicicleta para a aplicação cliente do jogo. Na secção seguinte apresenta-se a nova arquitetura proposta. Finalmente descreve-se a validação de desempenho realizada para as duas versões do jogo em termos de tempo médio de resposta ao utilizador e da análise do número de mensagens trocadas entre os processos do jogo. O artigo termina com a conclusão e sugestões de trabalho futuro.

JOGOS QUE PROMOVEM A ACTIVIDADE FÍSICA

Os jogos que incluem uma vertente de exercício físico são geralmente designados por *exergames*. Yoonsin Oh [2] define *exergame* como uma combinação de jogos de video e exercício incluindo atividades como treino de força, equilibrio e exercícios de flexibilidade. Numa sociedade em que o combate ao sedentarismo é cada vez mais importante, os jogos que promovem o exercíco físico surgem como ferramentas para chegar a indivíduos que são inativos por razões como falta de interesse, restrições de tempo, de custo, ou até limitações físicas [3]. O mercado dos jogos de video é liderado pelas empresas Microsoft, Sony e Nintendo que proporcionam consolas com sofisticada tecnologia para detetar o movimento do corpo humano e têm criado *exergames* de grande sucesso. Vários estudos confirmam que este tipo de jogos pode ter um impacto benéfico em aspectos fisiológicos [4], em aspetos sócio-psicológicos e motivacionais [5], e até na área de reabilitação física [6] [7].

No estudo sobre o desenho de *exergames* apresentado em [8] conclui-se que o sucesso de um *exergame* depende por um lado da capacidade do jogo proporcionar exercício físico e por outro lado da sua atratividade. Jogar *exergames online* pode ser uma forma de aumentar a atratividade do jogo ao combinar o exercício físico com uma maior interação social que pode, por exemplo, ajudar a combater a solidão dos mais idosos [9].

O jogo WebRun é um jogo que pretende motivar os jogadores pelo fato de associar à atividade física como o pedalar, a visualização de um percurso real a ser percorrido individualmente ou em competição dentro de um grupo. Através da API disponibilizada pelo Google Maps cada jogador ou grupo de jogadores pode visualizar as paisagens reais do percurso escolhido para a corrida e conhecer a posição relativa dos outros elementos do grupo. Espera-se que a competição com outros jogadores e as paisagens dos locais por onde a corrida conduz o jogador sejam aliciantes à utilização do jogo e logo à prática de exercício.

ARQUITETURAS DE JOGOS ONLINE

Como referido a maioria dos jogos *online* são construídos segundo a arquitetura C/S. Esta arquitetura permite controlar com facilidade o acesso dos jogadores, quer em termos de restrições de acesso quer para fins de faturação, facilitando a gestão do estado do jogo assim como a sincronização dos jogadores [10]. Tem no entanto problemas como falta de escalabilidade, pouca tolerância a falhas, pois o servidor constitui um ponto único de falha, e eventualmente problemas de atrasos no tempo de resposta aos jogadores quando por exemplo, o número de jogadores cresce muito rapidamente. A escalabilidade é conseguida usando *clusters* de servidores ou ainda dividindo o universo do jogo em múltiplos sub-mundos paralelos e distribuindo os jogadores por eles. De qualquer forma implica um dimensionamento de acordo com a previsão do número máximo de jogadores [10].

A primeira proposta de usar a arquitetura P2P em jogos *online* com milhares de utilizadores, os chamados MMOGS (Massively Multiplayer Online Games), foi publicada em 2004 por Knutssonn [11]. Neste modelo a ideia base é distribuir a carga de processamento por todos os intervenientes do jogo (isto é, pelos vários *peers*). Cada *peer* terá recursos para se manter a si próprio. Isto significa que as operações que no modelo C/S estão centralizadas no servidor vão agora ser divididas pelos clientes do jogo. Esta divisão torna o jogo escalável, pois mais jogadores implica automaticamente mais recursos, e torna o jogo mais tolerante a falhas pois deixa de haver um ponto único de falha. Mas o modelo também levanta questões difíceis de resolver, nomeadamente, como gerir, manter consistente e armazenar de forma persistente o estado do jogo [12] [13]. Ou ainda como proteger o jogo contra jogadores mal-intencionados numa arquitetura em que cada jogador também pode ser servidor do jogo, e portanto ter o controlo sobre parte do estado do jogo [10].

Ao construir a versão P2P do WebRun, optou-se por deixar centralizado no servidor o registo dos jogadores e a sua autenticação, além da criação de novos jogos. Sempre que um jogador cria um novo jogo, passará a funcionar como servidor para os jogadores do seu jogo, libertando o servidor central da comunicação com esses jogadores.

WEBRUN - VERSÃO CLIENTE-SERVIDOR

Para jogar WebRun cada jogador deve ter acesso a uma bicicleta de manutenção, que deverá dispor de um sistema de comunicação com um computador designado por controlador. O computador tem de estar preparado para receber a informação desse controlador e possuir instalada a aplicação Cliente do WebRun. Através da ligação à aplicação Servidor do WebRun o jogador interage com outros jogadores. A interação da aplicação Cliente com o Google Maps [14] obtém o mapa do percurso com o caminho a percorrer assinalado, e finalmente a interação da aplicação cliente com o Google Street View [15] permite visualizar algumas regiões ao nível do solo através de fotografias panorâmicas de 360º na horizontal e 290º na vertical. A arquitetura do jogo é esquematizada na Figura 1.

O jogo possui uma base de dados (BD), com informação referente aos utilizadores, às corridas e aos treinos. Uma corrida é um jogo entre vários jogadores. Um treino é um jogo realizado por um jogador sozinho, para praticar quando não tem adversários para convidar, ou simplesmente porque quer disfrutar do jogo sem concorrência.

A aplicação foi desenvolvida para Windows, usando C#, HTML e Java Script. Acede a uma base dados MySQL, a um servidor web Apache, e ao servidor do Google.

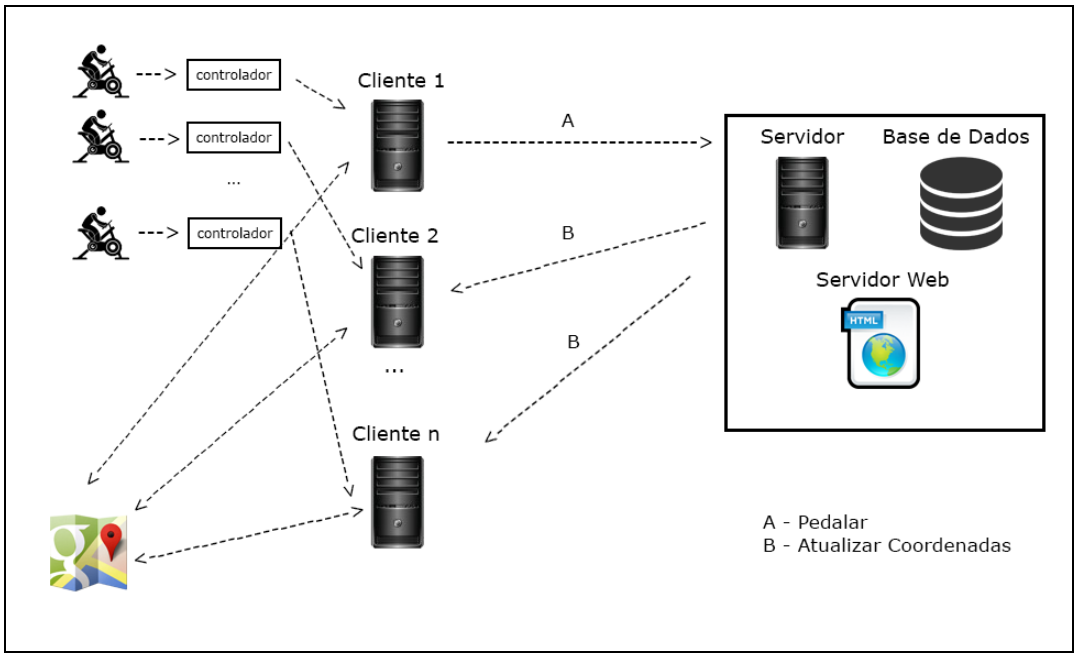

Figura 1. Arquitetura C/S do jogo WebRun.

Aplicação Cliente

Depois de fazer o *download* da aplicação Cliente, para aceder ao jogo os jogadores têm de primeiro efetuar um registo no jogo. Cada jogador é identificado por um *nickname* e autenticado por uma *password* com os quais pode depois aceder ao jogo. Quando o jogador efetua o *login* é establecida a ligação com o servidor e após um processo de autenticação é apresentada a janela principal do jogo *WebRun* (ver Figura 2). Nesta janela é permitido ao jogador ver a lista de utilizadores ligados ao *WebRun*, e para cada um ver o estado em que se encontra. O estado de cada jogador será um de entre quatro possíveis [16]:

- A aguardar o inicio de uma corrida: de cor amarela, representa um jogador que aceitou entrar numa corrida e está à espera que a mesma seja iniciada.
- A treinar: de cor vermelha, representa alguém que está a praticar um treino.
- A participar numa corrida: de cor verde, representa alguém que está a participar numa corrida com outros utilizadores.
- Em espera: de cor preta, representa um jogador que não está pendente de nenhuma ação, podendo ser convidado para uma corrida.

É também nesta janela que é permitido ao jogador criar um treino ou uma corrida e convidar outros jogadores para entrar num jogo. Ao escolher criar um jogo é enviada uma mensagem ao servidor com o percurso a realizar e com quantos adversários será realizado (em caso de treino será sem adversários, no caso de uma corrida poderá ter de 1 a 5 adversários). Após a criação do jogo, na janela principal da aplicação é ativada a opção de convidar os adversários e é enviada uma mensagem ao servidor com o identificador do adversário a convidar e a respetiva corrida. Se o adversário aceitar o desafio, é adicionado ao jogo pelo servidor que informa o criador do jogo dessa opção.

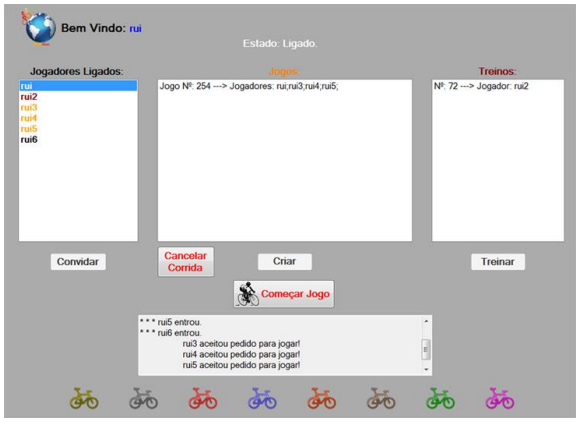

Figura 2 – Janela principal do jogo WebRun (aplicação Cliente).

Depois de todos os adversários aceitarem juntar-se ao jogo, o mesmo é iniciado pelo jogador que procedeu à sua criação. O servidor comunica aos jogadores o início do jogo e cada jogador carrega a janela do jogo (ver Figura 3). Esta janela para além de alguns botões para controlo das imagens obtidas do Google Maps, possui cinco *frames* que se descrevem de seguida e que na Figura 3 estão numeradas de i) a v).

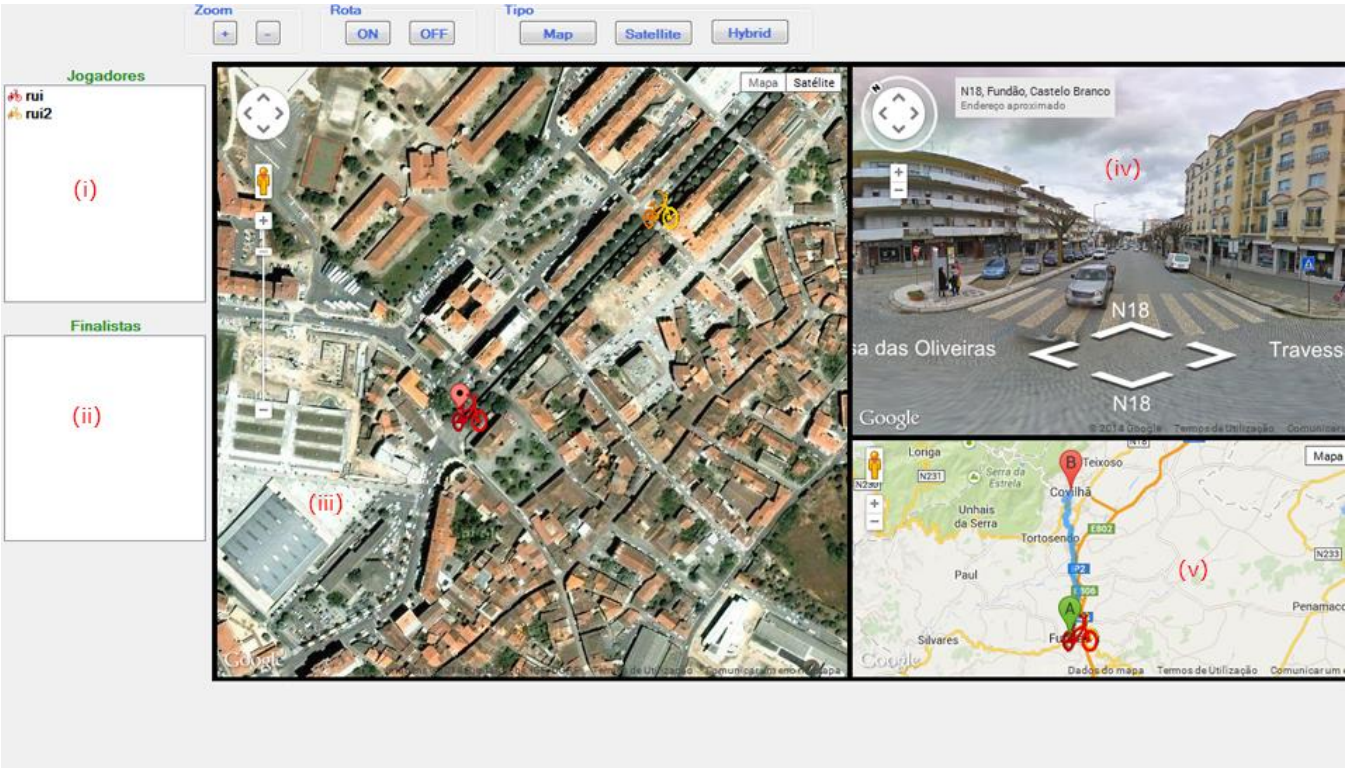

Figura 3 – Janela do jogo.

i) *Frame* Jogadores que mostra a lista de todos os jogadores na corrida (não necessário para o treino, caso em que apenas aparece o próprio jogador); ii) *Frame* Finalistas que vai mostrando os jogadores que já terminaram a corrida; iii) *Frame* principal, ao centro, onde é apresentada uma secção do mapa do percurso visto de cima, com o percurso assinalado e as bicicletas nele posicionadas consoante o percurso já efetuado. Cada jogador é identificado por uma bicicleta com a sua cor; iv) *Frame* que, quando possível, mostra o Google *Street View* da zona onde o jogador se encontra; v) *Frame* que mostra o mapa do percurso com a totalidade do caminho a percorrer assinalado;

Depois de carregada a janela do jogo, cada jogador pode começar a pedalar. Sempre que um jogador pedala, atualiza a sua posição no mapa, e é enviada para o servidor uma mensagem com o nickname do jogador, a identificação da corrida (ou jogo) em que participa e as coordenadas da sua posição.

O servidor ao receber esta mensagem vai reenviá-la para todos os outros jogadores de mesma corrida com exceção do jogador que enviou a mensagem. Assim, cada jogador (isto é, cada aplicação cliente) do mesmo jogo irá atualizar a posição dos jogadores concorrentes, acedendo ao servidor do Google para redesenhar o mapa.

Quando o primeiro jogador chega às coordenadas finais é anunciado o vencedor na *frame* Finalistas. Sempre que um novo jogador termina, é adicionado o seu identificador à mesma *frame*. Terminado o jogo, todos os jogadores da corrida voltam para a janela principal do jogo.

Aplicação Servidor

A aplicação Servidor contém o motor do jogo. É o servidor que regista os clientes na BD, faz a sua autenticação quando acedem ao jogo e faz a gestão de cada jogo (corrida ou treino) criado. Ao ser executada apresenta uma janela de administração com um botão para iniciar e outro para desligar a aplicação. A janela contém informação sobre a data e hora em que o servidor foi ligado, quais os jogadores que estão ligados, quais os jogos e quais os treinos em curso. Contém ainda uma *frame* que funciona como um *log* de todas as ações realizadas no jogo.

O Servidor contém uma *thread* principal que espera por ligações de clientes. Quando um jogador registado faz login é criada uma *thread* para comunicar com esse jogador. A comunicação entre a aplicação Cliente e a aplicação Servidor é centralizada no servidor, todo o trafego tem obrigatoriamente de ser encaminhado pelo servidor para os clientes. Se o número de jogadores aumentar consideravelmente o servidor irá ser um ponto de estrangulamento do jogo diminuindo o tempo de resposta aos clientes. No entanto, a comunicação com o Google Maps para visualização da posição no mapa é feita diretamente pela aplicação Cliente. Finalmente, a comunicação entre a bicicleta e o cliente, é feita através do controlador que se descreve na próxima secção.

LIGAÇÃO HARDWARE-SOFTWARE

O controlador efetua a interface hardware-software entre o dispositivo de treino (bicicleta fixa ou outro) e a aplicação Cliente do jogo. Tem como função principal detetar que o jogador pedalou, informando a aplicação desse facto e de qual a velocidade (ou cadência) desse pedalar. Eventualmente, numa futura versão pode receber do processo cliente o nível de dificuldade do jogo, usando-o para controlar a resistência da bicicleta de modo a simular as dificuldades do percurso (como por exemplo, uma subida acentuada).

A Figura 4 apresenta o diagrama de blocos do controlador. Trata-se de um circuito condicionador de sinal, que adapta os sinais elétricos de/para a bicicleta obtendo-os de uma ligação USB ao computador que executa o jogo. Baseia-se numa placa Arduíno [17], usando um protocolo de comunicações muito simples com o computador. Basicamente cada ação é representada pelo envio de um carácter, por exemplo, quando o utilizador pedala é enviada a letra 'P' ao Cliente, via porta USB. O controlador pode assumir diversas formas, quer em função da bicicleta utilizada, quer em função dos objetivos pretendidos para o treino físico.

Numa primeira forma a indicação de que o jogador pedalou pode provir de um sinal fornecido pela própria bicicleta nos casos em que esta disponha desse sinal. Regra geral trata-se de um sensor de proximidade indutivo, que deteta a rotação do pedal. As modernas bicicletas fixas de treino fornecem esse sinal, uma vez que se encontram equipadas de origem com controladores que indicam as distâncias percorridas, as calorias gastas, etc. A maior dificuldade nesta alternativa é ter acesso à ficha que veicula esse sinal, pois os fabricantes nem sempre fornecem o respetivo esquema elétrico. Nos testes do protótipo WebRun, as bicicletas utilizadas dispunham de origem desse sinal (Figura 5) [18].

Nos casos de bicicletas ou de outros dispositivos que não disponham desse sinal, ele pode ser obtido instalando na estrutura um sensor de proximidade indutivo que pode assumir a forma de um parafuso colocado perto do pedal de modo a detetar a sua rotação. Este tipo de sensor fornece um sinal digital cujo estado (0 ou 1) depende de estar ou não nas proximidades de um objeto metálico, no caso o pedal da bicicleta (Figura 6) [19].

Uma terceira versão para o controlador é captar o sinal de Eletromiografia (EMG) [20], diretamente dos músculos do jogador, para detetar quando este pedala. Este tipo de sinal é gerado pelos impulsos nervosos que acionam os músculos e que pode ser captado por elétrodos colocados na superfície da pele. Exige um sistema de amplificação e filtragem dos sinais mais aperfeiçoado, mas permite que o sistema possa ser utilizado para avaliar a condição física do jogador. Os elétrodos usados podem ser do tipo descartável ou estarem embutidos no equipamento que o jogador veste (por exemplo nos calções). Este tipo de controladores pode captar outros sinais fisiológicos do jogador, como a frequência cardíaca, a temperatura ou mesmo a tensão arterial e com base nesses sinais pode condicionar o evoluir do jogo. Por exemplo, se a frequência cardíaca atingir um valor demasiado alto, o jogador pode ser avisado que deve descansar ou o jogo pode mesmo ser terminado. Para a captação de diferentes sinais fisiológicos pode ser usada uma placa de aquisição do tipo Bitalino [21]

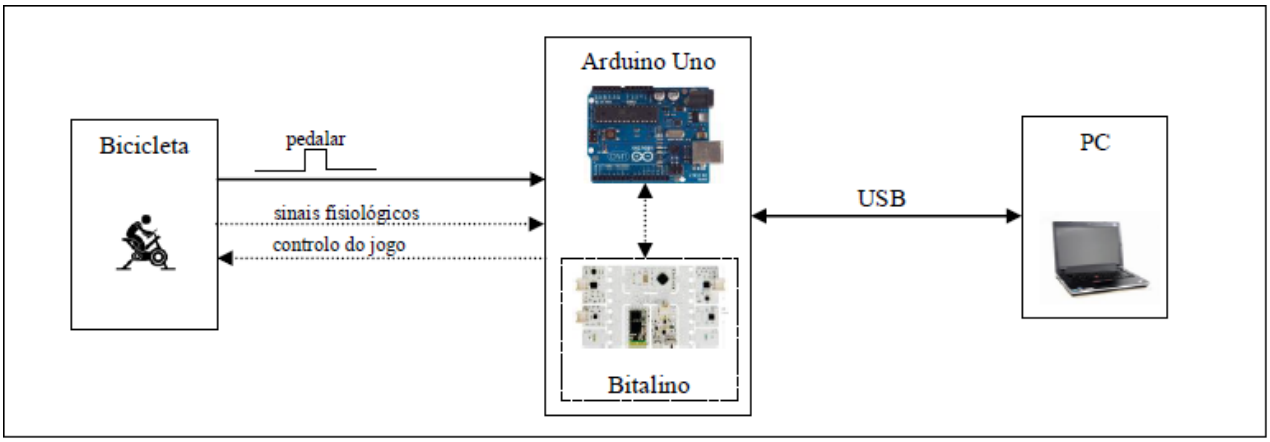

Figura 4 – Bloco Controlador

Figura 5 – Bicicleta fixa Domyos VM790

Figura 6 – Sensor de proximidade indutivo e pormenor de aplicação.

WEBRUN - VERSÃO PEER-TO-PEER

Na versão P2P, o registo dos jogadores mantém-se na aplicação Servidor. O mesmo acontece com a operação de login de um jogador registado e com o processo de criação dos jogos. A partir do momento em que um jogo, do tipo corrida, é criado, o criador do jogo passa também a ser o servidor desse jogo. No caso dos jogos do tipo treino, nada é alterado. Repare-se que neste caso durante o jogo não existe comunicação com o servidor, pois é a aplicação Cliente que acede ao *site* do Google Maps e atualiza o seu mapa não sendo necessário enviar o pedalar do jogador aos outros jogadores.

No caso de um jogo do tipo corrida, o criador do jogo funciona como um *super-peer* e será ele a receber e disseminar as posições de cada jogador do jogo para que cada jogador concorrente da mesma corrida possa atualizar no seu mapa as posições dos outros jogadores concorrentes. Terminada a corrida, é anunciado o vencedor pelo *super-peer* após o que os jogadores voltam para a janela principal, passando novamente a ser controlados pelo servidor central. A arquitetura da nova versão é esquematizada na Figura 7.

Nova Aplicação Cliente

A interface da aplicação para o jogador vai ser a mesma em comparação com a versão C/S. As mudanças são apenas estruturais, e na forma como a comunicação é feita com a aplicação servidor.

Na aplicação Cliente da máquina do jogador que cria uma nova corrida, será instanciada uma nova *thread* que ficará responsável pela gestão da nova corrida. Essa nova *thread*, a que chamamos "gestora da corrida" irá desempenhar o papel de servidor para o novo jogo e a aplicação cliente passa a desempenhar o papel de *super-peer*, isto é, será cliente e servidor ao mesmo tempo. A *thread* gestora da corrida ficará à espera das ligações dos jogadores que depois de convidados aceitem jogar a corrida. Para cada jogador que aceite jogar a corrida, será criada uma *thread* para comunicação entre a aplicação Cliente desse jogador e a *thread* gestora da corrida. Finalmente, a *thread* gestora da corrida será a responsável por receber "o pedalar" de cada jogador e disseminar essas mensagens pelos outros jogadores da corrida, que irão atualizar nos seus mapas as suas posições e as posições dos seus concorrentes. À medida que os jogadores vão chegando ao fim da corrida, o seu identificador aparece na *frame* Finalistas da janela do jogo de cada um dos jogadores dessa corrida.

Nova Aplicação Servidor

Como foi dito atrás, o Servidor vai continuar a ser o responsável pelo registo, autenticação, e criação de corridas, mas vai perder algumas das suas funções. A partir do momento em que todos os convidados para uma corrida a aceitam, a função de gerir o pedalar dos vários concorrentes dessa corrida passa para o *super-peer* que criou a corrida. No momento em que a aplicação Servidor permite que o criador do jogo dê início à corrida, envia na mensagem, para todos os outros jogadores da corrida, o endereço (IP e porto) do cliente que se tornou no *super-peer* do jogo, e que assumirá a gestão do resto do jogo.

Uma vez terminada a corrida, a responsabilidade volta para o Servidor, de forma a gerir os jogadores que continuam ligados à aplicação.

AVALIAÇÃO DE ESCALABILIDADE

Para estudar a escalabilidade das duas versões do jogo foi simulada a execução simultânea de vários jogos em cada versão. Nos testes foram usados 26 computadores ligados em rede, com o sistema operativo *Windows XP Professional*, com o *Service Pack 3*, um processador *Intel Core2 Quad CPU Q6600 @ 2.40GHz*, com 3 *GB* de memória *RAM* e uma placa de rede *Atheros L1 Gigabit Ethernet 10/100/1000Base-T*.

Metodologia

Para simular a execução dos jogos, foi usado um ficheiro onde previamente se armazenaram as coordenadas do percurso de uma corrida exemplo. Esse ficheiro foi colocado em todas as máquinas que contêm a aplicação Cliente e nesta foi criado um *Timer* com uma frequência de 200 milissegundos, para simular 5 pedaladas por segundo. De cada vez que o *Timer* expira é lido um novo par de coordenadas do ficheiro que contém o percurso e que depois é enviado para o servidor do jogo.

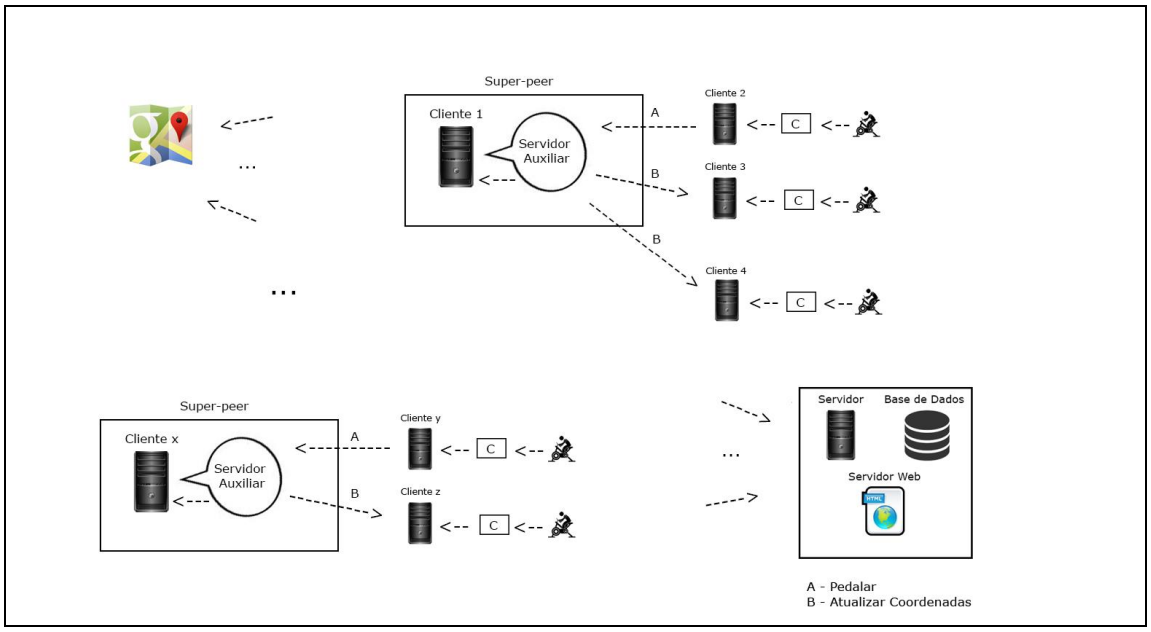

Figura 7 – Arquitetura da versão P2P do WebRun.

O servidor ao receber a mensagem com as coordenadas, e identificando o jogador que a enviou, vai disseminar a nova posição do jogador para os jogadores concorrentes. Na versão P2P o servidor que recebe a mensagem é a *thread* "gestora da corrida" do *super-peer* que criou o jogo. Esta mensagem é equivalente à mensagem enviada pelo controlador aquando do pedalar do jogador.

As mensagens para registo, login de utilizadores, convites para corrida e aceitação de convites, não foram consideradas para medição dos tempos de resposta do jogo. Isto é, supôs-se que os jogos estavam iniciados e mediu-se o tempo de resposta a cada jogador durante o processo em que todos os jogadores pedalam. Também se supôs que apenas se estavam a realizar corridas e nenhum treino.

Para obter o tempo médio de resposta do Servidor a cada jogador, foram guardados no Cliente, para cada mensagem recebida, o identificador do jogador que pedalou e o *timestamp* da máquina ao receber a mensagem. Os dados foram processados de modo a obter, em primeiro lugar, o tempo médio de resposta para cada jogador, considerando todas as mensagens recebidas durante a simulação da corrida.

No caso da versão C/S calculou-se depois o valor médio dos tempos de resposta considerando todos os jogadores em jogo, em todos os jogos, durante a simulação.

Para a versão P2P as mensagens que cada jogador recebe durante a fase de "pedalar" vêm do seu *super-peer*. Assim, calcular o tempo médio de resposta a cada jogador é calcular o tempo médio de resposta a todos os jogadores do mesmo jogo.

Para ambas as versões do jogo foi simulado 1 jogo com 2 jogadores, e *i* jogos com 5 jogadores cada, fazendo variar o valor de i de 1 até 5. Foram assim obtidos os tempos médios de resposta aos processos Clientes quando considerados 2, 5, 10, 15, 20 e 25 jogadores. O processo Servidor do jogo foi executado numa máquina dedicada e cada Cliente foi executado numa das outras máquinas disponíveis para a simulação.

Resultados

O gráfico apresentado na Figura 8, mostra os tempos médios de resposta aos jogadores do jogo WebRun, durante a fase em que os jogadores pedalam, para as duas versões do jogo, C/S e P2P, e considerando um número total de jogadores de 2 (1 jogo * 2 jogadores), 5 (1 jogo * 5 jogadores), 10 (2 jogos * 5 jogadores), 15 (3 jogos * 5 jogadores), 20 (4 jogos * 5 jogadores) e 25 (5 jogos * 5 jogadores).

Observando os valores obtidos para a versão C/S, podemos concluir que o tempo de resposta ao jogador cresce exponencialmente com o aumento do número de jogadores. O tempo obtido para 5 jogos (25 jogadores), cerca 17 segundos, mostra que dificilmente o jogo seria praticável, para um número razoável de jogos, mesmo que o servidor fosse uma máquina mais potente.

Para a versão P2P, podemos observar que o tempo de resposta é independente do número de jogadores, permanecendo perto dos 0,2 segundos. Podemos concluir que o jogo é perfeitamente escalável em relação ao aumento do número de jogadores, na fase de pedalar.

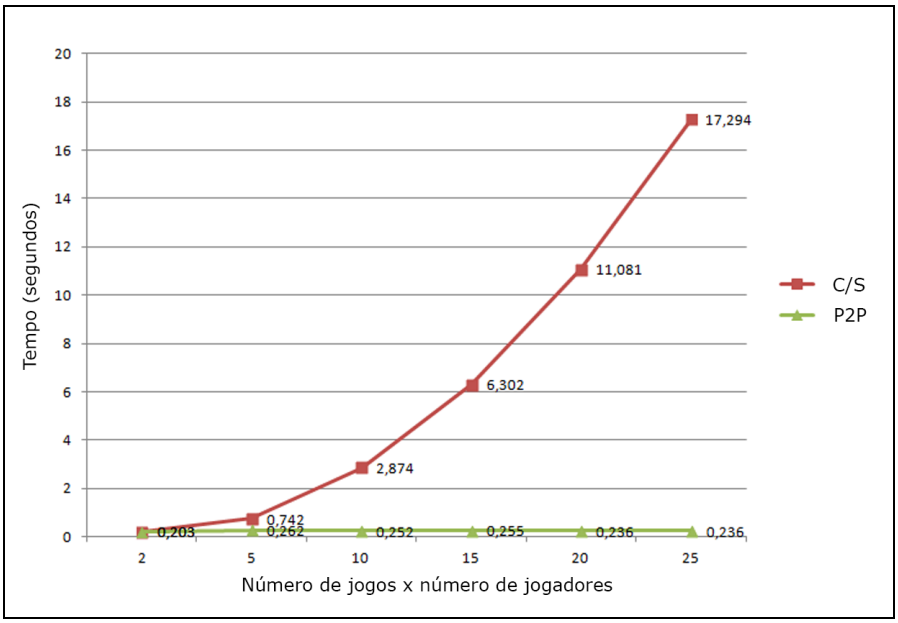

Figura 8. Gráfico da média dos tempos de resposta aos clientes nas arquiteturas C/S e P2P, quando todos os jogadores pedalam, variando o número de jogadores de 2 a 25.

Para melhor se perceber as diferenças entre as duas arquiteturas analisámos as mensagens trocadas entre jogadores e Servidor para a versão C/S e entre jogadores e *super-peer* para a versão P2P. Mais uma vez, não se consideram as fases de registo, autenticação e criação do jogo. Considera-se apenas a fase de pedalar, e são contabilizadas as mensagens supondo que cada jogador pedala uma única vez.

Na versão C/S, considerando um jogo com 2 jogadores, cada jogador envia uma mensagem "pedalar" ao Servidor, e o Servidor envia para o jogador "um" a mensagem de que o jogador "dois" pedalou e envia para o jogador "um" a mensagem de que o jogador "dois pedalou. Portanto, com 2 jogadores, o Servidor recebe 2 mensagens e envia 2. Com 5 jogadores, o servidor recebe 5 mensagens, uma de cada jogador, e por cada mensagem que recebe, vai enviar 4. Isto é, atualiza as coordenadas, dos jogadores concorrentes. Assim, com 5 jogadores, o Servidor recebe 5 mensagens e envia 20. O gráfico da Figura 9 mostra o número de mensagens recebidas e enviadas pelo Servidor quando o número de jogadores varia de 2 a 25, como descrito anteriormente e quando cada jogador pedala uma única vez. Como se pode observar o número total de mensagens recebidas mais enviadas passa de 4 com 2 jogadores, para 125 com 25 jogadores. Mesmo considerando que uma pedalada a cada 200 milissegundos é muito rápido, e que numa situação real duas pedaladas por segundo será uma velocidade razoável, o número de mensagens trocadas entre servidor e jogadores durante o jogo será enorme.

Na versão P2P, o número de mensagens recebidas e enviadas por um *super-peer* apenas depende do número de jogadores do jogo controlado pelo *super-peer*. Executar em simultâneo vários jogos apenas vai ter impacto nas mensagens trocadas com o Servidor principal na fase inicial dos jogos. Na fase de pedalar, o número de mensagens trocadas com o *super-peer* é independente do número de jogos. Analisando o número de mensagens recebidas e enviadas pelo *super-peer* quando aumenta o número de jogadores de um jogo, e considerando também que apenas se pedala uma vez, observa-se o seguinte: num jogo com 2 jogadores o *super-peer* recebe 2 mensagens "pedalar" e envia 2, uma para cada jogador. Num jogo com 3 jogadores, o super-peer recebe 3 mensagens pedalar, e por cada uma que recebe, envia 2 para os jogadores concorrentes, isto é, envia 6 mensagens. No gráfico da Figura 10 podemos observar que quando o número de jogadores de um jogo aumenta de 2 para 6, o número total de mensagens (recebidas mais enviadas) pelo *super-peer* varia de 4 para 36, o que como vimos é aceitável em termos do tempo de resposta que o jogador pode esperar obter, independentemente do número de jogos que venham a decorrer em simultâneo. De notar ainda que nesta versão algumas destas mensagens são locais, nomeadamente as que são trocadas entre a *thread* "gestora da corrida" e o cliente que é agora *super-peer*, facto que reduz ainda mais o impacto no trafego da rede.

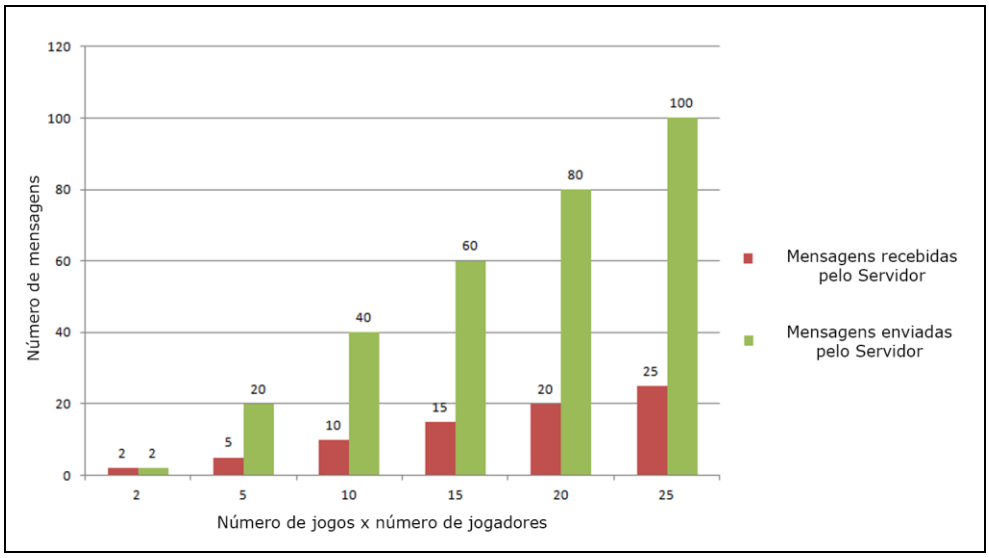

Figura 9. Gráfico com o número de mensagens enviadas e recebidas pelo servidor na arquitetura C/S considerando apenas jogadores em jogo a pedalar uma única vez e variando o número de jogadores de 2 até 25.

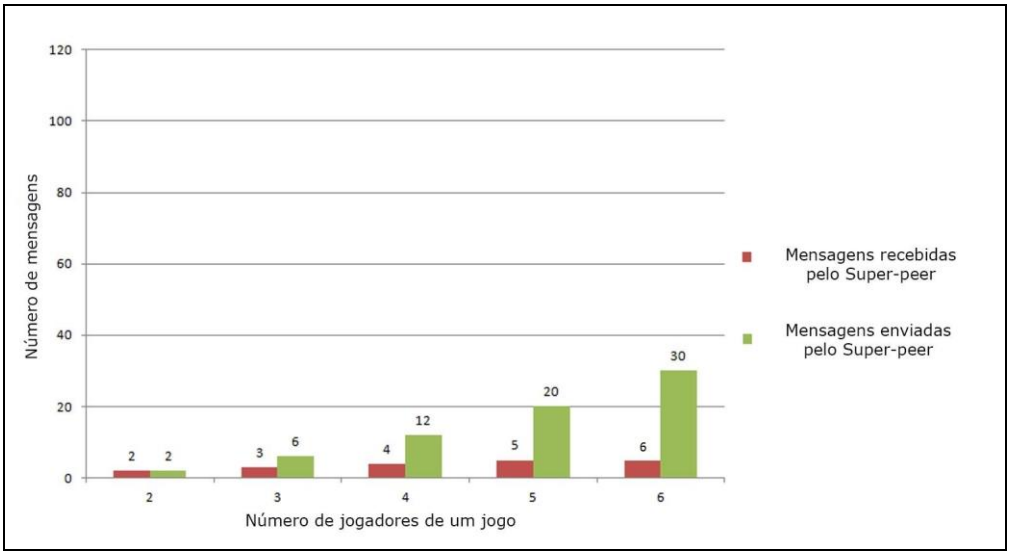

Figura 10. Gráfico com o número de mensagens trocadas pelo Super-peer, na arquitetura P2P, quando cada jogador pedala uma vez, variando o número de jogadores do jogo de 2 a 6.

CONCLUSÃO

Neste artigo foi proposta uma arquitetura P2P para um jogo *online* que promove a atividade física. Partindo do protótipo do jogo WebRun, construído segundo uma arquitetura C/S, foi desenvolvida uma versão P2P para o mesmo jogo. Sendo cada corrida do WebRun jogada por pequenos grupos de jogadores (2 a 6 elementos) foi proposto passar o controlo do estado de cada corrida, ou jogo, do Servidor central para a máquina do jogador que cria o jogo, criando um *super-peer*. A avaliação do tempo médio de resposta a cada processo cliente mostrou que a nova arquitetura (P2P) é completamente escalável em relação ao número de jogos que decorrem em simultâneo. A mesma avaliação mostrou que a versão C/S não é apropriada para um número elevado de jogadores. Um dos pontos fracos da arquitetura P2P para jogos multiutilizador é a maior possibilidade de fraude. Mantendo centralizado o processo de registo e autenticação, a segurança no acesso ao jogo é preservada. Num tipo de jogo em que o ganho é essencialmente o bem-estar físico e psicológico, a hipótese de fraude no decorrer do jogo não é

um fator crítico. Como trabalho futuro pretende-se estudar outros modelos de gestão do estado do jogo, por exemplo considerando a proximidade geográfica dos percursos de diferentes jogos.

AGRADECIMENTOS

Os autores agradecem a disponibilização do código do protótipo da versão inicial do jogo WebRun a Tiago Dias e José Castanheira.

Agradecem ainda o apoio financeiro à FCT – Fundação para a Ciência e Tecnologia (Projeto com a referência: PEst-OE/EEI/LA0008/2013)

REFERÊNCIAS

1. Araújo, P., Prata, P., Dias, T., Barroso, J. and Castanheira , J. WebRun - Promoting Healthy Sport Activity Through Interactive Online Games. *In Proc. of 6th Iberian Conference of Information Systems and Technologies*, (2011), 1-4.
2. Oh, Y. and Yang, S. Defining exergames and exergaming. *Proc. of Meaningful Play. Michigan State University (2010).*
3. Lieberman, D. A., Chamberlin, B., Medina, E., Franklin, B. A.,Sanner, M. B. and Vafiadis, D. K., The Power of Play: Innovations in Getting Active Summit 2011: A Science Panel Proceedings Report From the American Heart Association. Circulation, Journal of the American Heart Association, (April 2011). http://circ.ahajournals.org/content/early/2011/04/25/CIR.0b013e318219661d.
4. T. B. Stach, T. B., Heart Rate Balancing for Multiplayer Exergames. PhD Thesis, Queen's University, (2012). http://hdl.handle.net/1974/7525.
5. Wollersheim, D., Merkes, M., Shields, N., Liamputtong, P., Wallis, L., Reynolds, F., and Koh, L. Physical and psychosocial effects of Wii video game use among older women. *International Journal of Emerging Technologies and Society, 8, 2*, (2010), pp. 85–98.
6. Durkin, K., Videogames and young people with developmental disorders. *Review of General Psychology*, 14 , 2, (2010) pp. 122-140.
7. Tanaka,K., Parker, J.R., Baradoy,G., Sheehan, D., Holash,J.R.,and Katz, L. A Comparison of Exergaming Interfaces for Use in Rehabilitation Programs and Research. *Loading… The Journal of the Canadian Game Studies Association* 6, 9 (2012) pp. 69-81. http://loading.gamestudies.ca
8. Sinclair, J., Hingston, P., and Masek, M., Considerations for the design of exergames. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia (GRAPHITE '07). ACM, New York, NY, USA*, (2007). Pp. 289-295. DOI=10.1145/1321261.1321313 http://doi.acm.org/10.1145/1321261.1321313
9. Brox, E., Luque, L.F., Evertsen, G.J., Hernandez, J.E.G., Exergames for elderly: Social exergames to persuade seniors to increase physical activity. *In 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, (2011) , pp.546-549.
10. Neumann, C., Prigent, N., Varvello, M. and Suh, K.. Challenges in peer-to-peer gaming. *SIGCOMM Comput. Commun. Rev.* 37, 1 (January 2007), 79-82. DOI=10.1145/1198255.1198269 http://doi.acm.org/10.1145/1198255.1198269.
11. Knutsson, B., Lu, H., Wei Xu, Hopkins, B., Peer-to-peer support for massively multiplayer games. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies* , vol.1, (2004) pp.107- 118. DOI: 10.1109/INFCOM.2004.1354485.
12. Boulanger, J.-S., Kienzle, J., and Verbrugge, C., Comparing Interest Management Algorithms for Massively Multiplayer Games. *Netgames'06, Proc. of the 5th ACM SIGCOMM workshop on Network and System Support for Games*, (2006), pp. 1-12.
13. Gilmore, J. S. and Engebrecht, H.A., A Survey of State Persistency in Peer-to-Peer Massively Multiplayer Online Games. *IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 5*, (May 2012) pp. 818-834.
14. Google Maps, https://developers.google.com/maps/.
15. Google Street View, https://developers.google.com/maps/documentation/streetview/.
16. J. Castanheira, "WebRun: Jogo Web interactivo para promoção da prática desportiva". Dissertação de Mestrado em Engenharia Informática, Universidade da Beira Interior – Departamento de Informática, 2012.
17. Arduíno. http://*www.arduino.cc*
18. Domyos interactive system, VM 790, http://assets.domyos.com/vm790dis_pt.pdf.
19. Sensores indutivos. http://www.weg.net/br/Produtos-e-Servicos/Controls/Sensores-Industriais/Sensores-Indutivos.
20. Guerreiro, T. J. V. and Jorge, J.A., Controlo Miográfico de Dispositivos Móveis para Tetraplégicos. In proceedings of: 1st Workshop on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2006). http://www.di.fc.ul.pt/~tjvg/amc/emgtexting/files/emg-dsai.pdf
21. Placa Bitalino. http://www.bitalino.com/