Instituto Superior de Engenharia do Porto

Acesso a Laboratórios Remotos via Ambientes Imersivos

Acesso Remoto a um Multímetro

David Tomé Oliveira Costa

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
Área de Especialização em
Arquitecturas, Sistemas e Redes

Orientador: Gustavo R. Alves
Co-Orientador: Paulo D. Ferreira

Júri:
Presidente:
Doutora Maria de Fátima Coutinho Rodrigues, Instituto Superior de Engenharia do Porto

Vogais:
Doutor Juarez Bento da Silva, Universidade Federal de Santa Catarina
Doutor Gustavo Ribeiro da Costa Alves, Instituto Superior de Engenharia do Porto
Eng. Paulo Alexandre Duarte Ferreira, Instituto Superior de Engenharia do Porto

Porto, Julho 2011

Ao meu Pai, Mãe e Avó

Agradecimentos

Ao Prof. Gustavo Alves, pela oportunidade oferecida, e apoio prestado no desenvolvimento da solução, e na elaboração da dissertação.

Ao Prof. Paulo Ferreira, pelo apoio prestado na resolução de problemas relacionados com redes de computadores, e na elaboração da dissertação.

Aos Profs. Juarez Bento da Silva (UFSC) e Roderval Marcelino (SATC), pela sua disponibilidade para testar a solução final.

Aos meus colegas Bruno Bichels e Marcelo Andriolli, pela sua disponibilidade para me ajudar em testes, e na resolução de problemas da solução final.

Ao meu colega Vasco Sousa, pelo apoio prestado em tarefas operacionais, e em testes à solução final.

Ao meu colega Tiago Pinto, pela sua disponibilidade para testar a solução final.

A todos os docentes do MEI-ISEP, que me ajudaram a melhorar as minhas competências.

A todos os meus colegas e amigos, que me incentivaram a ingressar no MEI-ISEP.

Ao meu Pai, Mãe e Avó, por me incentivarem a ingressar no MEI-ISEP, e pelo apoio prestado nos momentos mais difíceis.

Resumo

Recentemente, tem-se assistido à utilização de ambientes imersivos 3D em vários domínios tais como: actividades empresariais, educativas, lúdicas, entre outras devido à expansão do Second Life. A finalidade deste conceito é oferecer aos utilizadores um acesso alternativo a valências existentes no mundo real, a partir de um computador ligado à Internet. Uma aplicação prática pode ser a sua utilização em laboratórios remotos, com a finalidade de controlar remotamente instrumentos de medição, a partir de um ambiente imersivo. Para isso, o mesmo deve permitir a construção de um laboratório virtual e respectivos instrumentos, também virtuais. Este tipo de solução é viável, devido a existirem dispositivos com interfaces de acesso remoto, e ambientes 3D desenvolvidos em linguagens de programação que possuem bibliotecas de código para protocolos de redes de computadores. A finalidade deste trabalho é desenvolver uma metodologia de acesso remoto, a instrumentos de medição em laboratórios de electricidade e electrónica, usando ambientes imersivos 3D. Como caso de estudo, o instrumento utilizado é um multímetro, controlado remotamente a partir de uma reprodução num mundo virtual, construído no ambiente 3D Open Wonderland. Nessa reprodução virtual, numa primeira fase, só serão disponibilizadas para medição, um conjunto limitado das variáveis eléctricas passíveis de medir através do multímetro seleccionado.

Palavras-chave: Acesso remoto, ambientes imersivos, multímetro.

Abstract

Recently, 3D immersive environments have been spreading through several domains, such as: enterprise activities, education, entertaining, etc., due to the growth of Second Life. This concept goal is provide an alternative access to users in order to use existing facilities in real world, from a computer connected to the Internet. It can be applied to make remote laboratories accessible from immersive environments, in order to remotely control measuring instruments. The 3D environment should allow the virtual laboratory and its virtual instruments developing. This type of solution is practicable, due to measuring devices with remote access interfaces availability, and 3D environments developed in programming languages which provide libraries including code for network protocols features. This work goal is to develop a remote access methodology to measuring instruments in an electricity and electronics laboratory, using 3D immersive environments. A multimeter is used as a case study. It is remotely controlled from a virtual reproduction in a virtual world, built in the Open Wonderland 3D environment. In that reproduction and in a first stage, it will only be available for measuring, a limited set of electrical variables provided by the selected multimeter.

Keywords: Remote access, immersive environments, multimeter.

Índice

Lista de Figuras

Lista de Tabelas

Lista de siglas e abreviaturas

2D	Duas dimensões
3D	Três dimensões
API	Application Programming Interface
BES	Banco Espírito Santo
CIETI-LABORIS	Centro de Inovação em Engenharia e Tecnologia Industrial – Laboratório de Investigação em Sistemas
DSL	Digital Subscriber Line
GMT	Greenwich Mean Time
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
IDE	Integrated Development Environment
IP	Internet Protocol
IRC	Internet Relay Chat
JAR	Java Archive
LAN	Local Area Network
MIT	Massachusetts Institute of Technology
PC	Personal Computer
PDF	Portable Document Format
PHP	PHP: Hypertext Preprocessor
RAM	Random Access Memory
RS-232	Recommended Standard 232
SCPI	Standard Commands for Programmable Instruments
SDK	Software Development Kit
SIP	Session Initiation Protocol
SSH	Secure Shell
TCP	Transmission Control Protocol
UPS	Uninterruptible Power Supply
URL	Uniform Resource Locator
USB	Universal Serial Bus
USD	United States Dollar
VIP	Very Important Person
VOIP	Voice over Internet Protocol
XML	eXtensible Markup Language

1. Introdução

O presente documento descreve um trabalho de mestrado elaborado para o grupo CIETI-LABORIS, do ISEP. Actualmente este grupo tem em curso um projecto, denominado Physics LabFARM, que pretende dotar as unidades curriculares do ISEP, que incluem aulas laboratoriais na área de Física, com laboratórios ou recursos remotos, que complementem as aulas presenciais [VISIR@ISEP, 2010].

Um laboratório remoto consiste num laboratório real com dispositivos que podem ser acedidos e controlados remotamente. No caso desta tese pretende-se que um instrumento de medição ou dispositivo electrónico, denominado multímetro[1], seja acedido e controlado remotamente. O laboratório tem de ser integrado num ambiente imersivo 3D[2], do estilo Second Life e afins, sendo acessível aos utilizadores directamente ou por intermédio de um *browser*[3].

Como resultado final deste trabalho, o utilizador terá acesso a uma reprodução virtual do multímetro real, e através da mesma poderá operá-lo no ambiente 3D. Os valores apresentados no visor do multímetro real devem ser visíveis nesse ambiente em tempo real. Um clique num botão na reprodução virtual do dispositivo no ambiente 3D, deve ter a acção repercutida no multímetro situado no laboratório real. Para toda a solução funcionar correctamente, os terminais do dispositivo têm de estar previamente ligados a um circuito eléctrico ou electrónico, e o mesmo dispositivo deve estar ligado a uma rede de computadores TCP/IP[4], por exemplo, através de uma interface *Ethernet*[5]. A solução final deste trabalho estará fisicamente montada num laboratório real do grupo de investigação CIETI-LABORIS.

Neste trabalho, o acesso remoto ao multímetro serve de caso de estudo para encontrar uma possível metodologia de como ligar/aceder remotamente a outros instrumentos de medição, a partir de um ambiente 3D.

1.1 Objectivos

Os objectivos deste trabalho dividem-se em gerais e específicos. Os objectivos gerais são:

- Instalar/configurar um ambiente 3D.
- Construir/desenhar um cenário virtual.

[1] Multímetro: dispositivo electrónico que mede vários valores desde tensão ou corrente eléctrica num circuito eléctrico ou electrónico, é usado regularmente em laboratórios de Engenharia Electrotécnica e Electrónica.
[2] Designação usada em computação gráfica para caracterizar objectos/entidades com três dimensões, que constituem largura, altura e profundidade.
[3] *Browser*: aplicação informática que permite a um utilizador no computador, interagir ou visualizar páginas *Web* da Internet.
[4] TCP/IP: o protocolo TCP funciona sobre o protocolo IP, ambos os protocolos de comunicação asseguram a transmissão da informação em redes de computadores, sendo o seu uso generalizado na Internet. A função do protocolo TCP é garantir a fiabilidade da transmissão, o protocolo IP é responsável pelo endereçamento dos diferentes nós da rede de computadores.
[5] *Ethernet*: tecnologia que estabelece a interligação física numa rede de computadores.

- Construir uma reprodução virtual do multímetro.
- Efectuar remotamente medições de variáveis eléctricas.
- Visualizar remotamente o resultado das medições.

Estes objectivos são detalhados nos seguintes objectivos operacionais ou específicos:

- A solução final deverá ter um ambiente 3D, devidamente instalado e configurado numa máquina (PC), acessível/utilizável por todos os utilizadores com acesso via Internet.
- O ambiente 3D da solução final deverá disponibilizar aos utilizadores um cenário virtual onde estes possam navegar. Esse cenário deverá incluir pelo menos uma reprodução virtual do multímetro com funcionalidades de interacção com os utilizadores.
- Todas as acções de interacção do utilizador efectuadas no dispositivo virtual, devem-se repercutir no multímetro real.
- Na solução final deverá ser possível medir variáveis eléctricas com o multímetro, remotamente, a partir do ambiente virtual.
- O valor medido, apresentado no visor do multímetro, deverá ser também apresentado no dispositivo virtual da solução final.
- Na solução final, a utilização do dispositivo virtual, deverá ser controlada por um mecanismo de acessos concorrentes.

1.2 Plano de trabalhos

A divisão do trabalho desta tese foi planeada de acordo com a elaboração sequencial das seguintes tarefas:

- Análise das potencialidades dos ambientes 3D mais conhecidos.
- Selecção do ambiente 3D ou mundo virtual que serve de suporte ao laboratório virtual.
- Instalação do servidor de ambientes 3D na máquina adequada.
- Configuração do servidor de ambientes 3D.
- Construção/desenho do dispositivo virtual.
- Ligação do multímetro ao computador onde executa o servidor de ambientes 3D.
- Programação das funcionalidades de comunicação remota no dispositivo virtual, para efectuar medições no homólogo real.
- Programação de um mecanismo de acessos concorrentes no dispositivo virtual.
- Implementação de um mecanismo de arranque automático do servidor de ambientes 3D.

- Instalação e configuração de uma UPS[6] e seu software, para um mecanismo de encerramento automático do sistema operativo (SO) em caso de falha de energia eléctrica.

1.3 Estrutura da tese

Depois deste capítulo de introdução, é feita uma revisão de vários ambientes 3D ou mundos virtuais no capítulo 2, onde se levantam, analisam e comparam as suas características mais importantes, e possíveis laboratórios virtuais/remotos já implementados. O capítulo 3 apresenta a solução a desenvolver em termos genéricos, com a explicação da selecção do ambiente 3D, e das funcionalidades do instrumento de medição seleccionado. Nesse capítulo a exposição das arquitecturas física e lógica da solução final mostra como os seus diferentes elementos se interligam. Toda a implementação do ambiente 3D é abordada no capítulo 4, desde a instalação/configuração até à administração do servidor do ambiente 3D. Também fazem parte do capítulo 4 temáticas de como um utilizador acede a um mundo virtual, como implementa/aplica segurança nos seus objectos 3D, e quais as funcionalidades e mecanismos adicionais que tornam a solução final menos vulnerável a falhas. O capítulo 5 trata a construção do dispositivo virtual. Na metodologia de construção interessa saber que funcionalidades implementar, como programar o seu aspecto gráfico, como implementar a comunicação com o multímetro e como programar um mecanismo de acessos concorrentes para o caso do dispositivo virtual ser usado num ambiente multi-utilizador. Para testar a solução final, são descritos vários cenários de teste no capítulo 6, onde se referem a metodologia, seguida dos resultados obtidos. Um balanço final do trabalho e o potencial da solução são analisados no capítulo 7, de conclusão e perspectivas futuras.

[6] UPS: fonte de alimentação eléctrica ininterruptível que assegura o funcionamento de dispositivos eléctricos em caso de falha de energia.

2. Revisão dos ambientes 3D

Antes de descrever o desenvolvimento da solução proposta, apresentam-se as tecnologias existentes de ambientes imersivos/3D, que permitem a construção ou utilização de mundos virtuais. O objectivo deste capítulo é rever os vários ambientes 3D, analisando as suas características e possíveis laboratórios virtuais/remotos já implementados, nas várias secções. Esta revisão serve de ponto de partida para a selecção do ambiente 3D que suportará o cenário do laboratório virtual. Factores como a facilidade de construção/programação de conteúdos adicionais, custo, possibilidade de comunicação entre utilizadores serão decisivos nessa selecção.

Pode-se considerar como um ambiente 3D qualquer percepção simultânea de altura, largura e profundidade, por parte do ser humano. O universo em si já é um ambiente 3D. No que diz respeito às aplicações informáticas existem as que usam interfaces 2D[7] ou 3D, com o utilizador. Nas 2D apenas existe percepção de altura e largura, enquanto as 3D acrescentam a noção de profundidade em relação àquelas. Note-se que apesar de existirem aplicações com interfaces 3D, estas são projectadas em ecrãs 2D através do processo de renderização[8] [Franklin, C., 2009]. Actualmente, já existem ecrãs 3D que possibilitam a visualização de imagens a 360º. Tem-se como exemplo de ecrã 3D, o Perspecta da Actuality Systems,Inc. utilizado para auxílio a pessoal médico [Actuality Systems, Inc., 2009].

São usadas interfaces 3D nas mais variadas aplicações informáticas, desde jogos, aplicações médicas, mundos virtuais, etc. O desenvolvimento de bibliotecas gráficas tais como o OpenGL[9] e o DirectX[10] permitiram de modo considerável a expansão de aplicações 3D.

O conceito de ambiente imersivo consiste num ambiente/mundo virtual interactivo simulado, acedido simultaneamente por vários utilizadores, em que a comunicação é estabelecida através de uma interface de rede (Internet). Os mundos virtuais *online* baseiam-se nas seguintes premissas [What is a Virtual World?, 2009]:

- **3D:** ter uma interface 3D é importante para ser o mais próximo possível de um modelo real.
- **Baseados em *avatares*:** *avatar* é uma representação virtual de cada utilizador. Muitos mundos virtuais dispõem das mais variadas configurações para construção de *avatares*, de modo a que sejam o mais exclusivo possível. Diferentes utilizadores podem ser reconhecidos por outros através dos *avatares*. Os utilizadores também podem interagir entre si.
- **Múltiplos meios de comunicação:** num mundo virtual a comunicação entre os utilizadores deve ser possível através de escrita, voz, animações e vídeo.

[7] Designação usada em computação gráfica para caracterizar objectos/entidades com duas dimensões, que constituem a largura e a altura.
[8] O processo de renderização consiste em representar objectos gráficos 3D em ecrãs 2D.
[9] OpenGL: biblioteca para construção de aplicações informáticas gráficas 2D ou 3D.
[10] DirectX: conjunto de bibliotecas de código, desenvolvido pela Microsoft, usado para programação de jogos ou objectos gráficos.

- **Conteúdos criados por utilizadores:** os mundos virtuais permitem certas funcionalidades como: construir edifícios, configurar veículos, comprar terrenos, etc., tudo criado pelos utilizadores. Em alguns casos pode ser necessário ter dinheiro para o efeito. Alguns mundos virtuais possuem uma economia dita interior, onde o utilizador precisa de aplicar dinheiro real, para depois obter dinheiro virtual, desfrutando assim da economia do mundo virtual.
- **Objectivos a atingir:** dependendo das funcionalidades disponíveis em cada mundo virtual, estes são concebidos para que um utilizador tenha objectivos a atingir tais como: conhecer pessoas, divulgar produtos ou negócios, jogar os mais diversos jogos, etc.
- **Ferramentas de interface *Web*:** um mundo virtual pode ser acedido através de aplicações cliente, ou através de um *browser*.
- **Ferramentas de comunicação:** para interacção entre os utilizadores devem estar disponíveis funcionalidades tais como: perfis de utilizador, blogues, *wikis*[11], etc.

2.1 Second Life

O Second Life (SL) é porventura o mundo virtual 3D mais conhecido do público em geral. Foi lançado pela empresa Linden Lab em 2003, desenvolvido numa linguagem de programação criada para o efeito, denominada Linden Scripting Language [LSL Portal, 2009]. Cada utilizador pode aceder a este mundo virtual através de uma aplicação cliente denominada Second Life Viewer. O acesso é controlado através das credenciais: primeiro e último nome, e palavra-chave (*password*). Na Tabela 1 encontram-se os requisitos mínimos [System Requirements, 2009] e recomendados de funcionamento do Second Life Viewer, para os sistemas operativos Windows, MacOS e Linux.

Para utilizar o SL, o primeiro passo, é configurar o *avatar* ou a representação do utilizador no mundo virtual, onde existem as mais variadas opções, desde sexo (masculino ou feminino), cor de pele, cor dos olhos, cor do cabelo, formato do cabelo, roupa, altura, largura, entre outras.

Este ambiente oferece aos utilizadores mundos virtuais, que são reproduções das valências que estão disponíveis no mundo real, de acordo com a Figura 1. No caso dessa figura as valências ilustradas aparentam ser uma zona de edifícios comerciais ou residenciais numa área balnear, com o objectivo de disponibilizar aos utilizadores um mundo virtual com a finalidade de publicitar actividades turísticas. Muitas das funcionalidades do SL são usadas para promover as mais variadas actividades, desde negócios, entretenimento, educação, etc.

[11] *Wiki*: termo usado para identificar um determinado conjunto de documentos ou o software que os criou.

	Windows		MacOS		Linux	
	Requisitos mínimos	**Requisitos recomendados**	**Requisitos mínimos**	**Requisitos recomendados**	**Requisitos mínimos**	**Requisitos recomendados**
Ligação Internet	Cabo ou DSL					
Sistema Operativo	XP ou Vista		Mac OS X ≥10.4.11	Mac OS X ≥10.5.4	Qualquer distribuição recente de 32 bits	
Processador	≥800 MHz Pentium III ou Athlon	≥1.5 GHz (XP) ≥2.0 GHz (Vista)	≥ Intel 1.5 GHz	≥ Intel Core 2 Duo 2.0 GHz	≥800 MHz Pentium III ou Athlon	≥1.5 GHz
RAM	≥512 MB	≥1GB	≥512 MB	≥1 GB	≥512 MB	≥1GB
Placa Gráfica	NVIDIA GeForce 6600 (XP/Vista/7) ATI Raedeon: 8500 (XP), 9250 (XP), 9500(Vista/7), Intel 945 chipset[1] (XP/Vista/7)	NVIDIA: 9600(XP/Vista/7), 9800(XP/Vista/7), 275GTX (XP/Vista/7) 295GTX (XP/Vista/7) ATI: 4850(XP/Vista/7), 4870(XP/Vista/7), 4890(XP/Vista/7), 5850(XP/Vista/7), 5870(XP/Vista/7), 5970(XP/Vista/7)	ATI Radeon 9200, NVIDIA: GeForce 2, GeForce 4	ATI: 4850, 4870, NVIDIA: 9800	NVIDIA GeForce 6600, ATI Radeon 8500, 9250	ATI: 4850, 4870 NVIDIA: 9600, 9800
Resolução Ecrã (px)	1024 x 768	≥1024 x 768	1024 x 768	≥1024 x 768	1024 x 768	≥1024 x 768

Tabela 1 - Requisitos mínimos do Second Life Viewer

Figura 1 - Edifícios no SL

Num mundo virtual SL podem existir lojas, cinemas, empresas, bancos, espectáculos de música, escolas, restaurantes, entre outros. Muitas empresas do mundo real fazem publicidade dos seus produtos no SL, como a conhecida IBM[12], com o seu centro de negócios virtual. No caso português, o BES[13] [FORCELLA, T., 2007] tem uma agência bancária nesse mundo virtual, assim como a Universidade de Aveiro possui um *campus* virtual [Universidade de Aveiro inaugura ilha no Second

[12] IBM: empresa fabricante de várias tecnologias de sistemas informáticos.
[13] BES: banco português.

Life, 2007]. Esta instituição planeia disponibilizar actividades curriculares ou complementares às aulas, nesse espaço virtual. A Suécia tornou-se o primeiro país a ter uma embaixada no SL [Simmons, C., 2007]. Muitas bandas e cantores já utilizaram este mundo virtual para darem concertos.

Estas funcionalidades só podem estar ao alcance de qualquer entidade, desde que disponha de uma quantia monetária suficiente, pois o SL tem a sua própria economia interna com moeda própria designada Linden Dollar ($L), com a paridade 1 USD[14] = $L 250 [Currency Exchange: Buy or Sell Linden Dollars (L$), the currency of Second Life, 2009] ou 1 EUR[15] = $L 320 [FRIAS, P., 2007]. A Universidade de Aveiro investiu 3000 EUR só para ter a ilha onde construiu o seu pólo virtual. Existem dois tipos de contas de utilizador: Basic e Premium. Na primeira apenas é possível ter acesso às diferentes funcionalidades do mundo virtual. Um utilizador de conta Basic recebe ao início $L 250, e pode fazer compras em áreas temporárias para o efeito, além de poder construir edifícios nas mesmas. Tanto os edifícios como as áreas podem ser apagados num dado instante. Cada conta básica adicional acarreta um custo. As contas Premium têm um custo mensal, permitindo porém a aquisição "permanente" de bens e serviços, ou assistência técnica da empresa detentora do ambiente virtual. Segundo dados de finais de 2006, o SL contava com cerca de 8 milhões de utilizadores registados, sendo que 40 % destes eram europeus e Portugal era o 12º país com mais utilizadores em todo o mundo, e eram transaccionados cerca de 1,6 milhões USD por dia [Neves, N., 2008].

Em relação a desenvolvimento de novas aplicações usando o SL, a Linden Lab disponibiliza uma API[16] proprietária [Second Life Develop, 2009]. A mesma subdivide-se em vários grupos, de acordo com funcionalidades, tais como: integração de funcionalidades do SL em páginas *Web*, câmbio de Linden Dollar para outras moedas, gestão de mapas e de utilizadores, etc. Recentemente e relacionado com o SL, surgiu o Project Snowstorm [Project Snowstorm, 2009], que é uma comunidade cujo princípio é desenvolver código *open source*[17], para melhorar o desempenho, fiabilidade e usabilidade do Second Life Viewer, bem como implementar novas funcionalidades.

Um exemplo de laboratório remoto baseado em SL, é o SecondLab [García-Zubia, J.; et al, 2010], que permite o controlo remoto de *microbots*[18]. Funciona sobre o laboratório remoto da Universidade de Deusto[19], fazendo a interface do utilizador para este laboratório, através do SL.

[14] USD: moeda dos Estados Unidos da América.
[15] EUR: código usado para designar o Euro, moeda adoptada por alguns países da União Europeia.
[16] API: conjunto de especificações/funções desenvolvidas para serem usadas por uma linguagem de programação para construção de aplicações.
[17] *Open source*: corresponde a aplicações informáticas, cujo código fonte é de acesso aberto e gratuito.
[18] *Microbots* são robôs em formato de miniatura.
[19] Universidade espanhola, situada na cidade de Bilbao.

2.2 OpenSim

A designação OpenSim (OS) deriva de Open Simulator e não se trata de um mundo virtual propriamente dito, mas sim de um servidor aplicacional para criação de mundos virtuais. A sua primeira versão data de Abril de 2007, sendo que a mais recente data de Setembro de 2010. É *open source*, logo sem custos no desenvolvimento de aplicações/funcionalidades. Suporta linguagens de programação como C#, JScript ou VB.NET.

Entre as suas vantagens destacam-se as seguintes:

- Criação de vários mundos virtuais numa só aplicação.
- Pode ser acedido por aplicações clientes de outros mundos virtuais, como o Second Life Viewer.
- Pode comunicar com mundos virtuais do SL (Figura 2).
- Possui múltiplas configurações de *avatares* desde cor da pele, roupas e objectos pessoais.
- Contém motores de simulação de física do mundo real.
- Permite criar conteúdo 3D em tempo real.

A utilização de mundos virtuais criados pelo OS implica a criação de uma conta de utilizador no portal OSGrid [Configuration, 2009], que é o principal mundo virtual num servidor OS, ou seja, o mundo virtual por omissão onde os utilizadores se conectam. Um mundo virtual OS executa numa aplicação cliente, como o Second Life Viewer ou Hippo Viewer para aceder ao OSGrid. Também pode ser acedido por um *browser*, usando um *add-on*[20] para o Mozilla Firefox denominado Xenki.

A configuração do OS é feita no ficheiro de texto OpenSim.ini, com a sua própria estrutura. Existem variantes do servidor OS para sistemas Windows, Linux e MacOS. Para armazenamento de dados persistentes, o OS inclui um pequeno motor de base de dados denominado SQLite apenas apropriado para testes, mas não para utilização em larga escala. Outros motores de bases de dados como MySQL, MSSQL e Postgresql são também compatíveis, embora só o MySQL esteja neste momento 100% funcional.

[20] *Add-on*: componente de software que constitui uma nova funcionalidade a ser adicionada a uma aplicação informática.

Figura 2 - Visualização do Second Life a partir do OpenSim

Num servidor OS existe um conjunto de permissões [Permissions (Server), 2009] de gestão de mundos virtuais, das quais se destacam:

- ***Region Master*:** corresponde ao dono de uma determinada região do mundo virtual.
- ***Administrator*:** é aquele que administra um conjunto de regiões.
- ***Object Manager*:** refere-se à propriedade de um objecto de um determinado utilizador. O dono do objecto pode alterar as propriedades do mesmo, mas não pode alterar outros objectos que não lhe pertençam. Cada utilizador pode alterar propriedades de objectos que estejam dentro dos seus terrenos. Um *Administrator* pode alterar propriedades de objectos de utilizadores que estejam no conjunto de regiões que gere.

O servidor OS tem dois modos de funcionamento: autónomo ou em grelha. O primeiro consiste no servidor executar apenas numa instância (OpenSim.exe) numa só máquina, enquanto que, o segundo pode ter múltiplas instâncias a executarem na mesma máquina ou em máquinas diferentes.

Para aperfeiçoamento, o OS tem uma comunidade onde utilizadores e programadores podem comunicar através de IRC[21] ou *mailing-lists*[22]. É possível a construção de artigos *wiki* de forma a melhorar a documentação disponível. O desempenho do software sai melhorado pelo reporte de *bugs*[23] e envio de *patches*[24]. A comunidade do OS possibilita a utilizadores e programadores a participação no desenvolvimento do servidor, através de *Office Hours*[25] semanais que podem ser para programação, documentação ou testes.

[21] IRC: protocolo de comunicação da Internet usado para conversação instantânea entre utilizadores.
[22] M*ailing-lists*: lista de endereços de correio electrónico, muito útil quando se pretende enviar uma mensagem para todos os endereços ao mesmo tempo.
[23] *Bug*: falha que provoca o funcionamento anormal de uma aplicação informática.
[24] *Patch*: componente de software criado com o objectivo de corrigir defeitos de uma aplicação.
[25] *Office hour*: termo usado para encontros de discussão entre programadores, como vista a descobrir novas soluções.

No caso do OS, existe um projecto de laboratórios remotos, denominado "THE REXLAB V VIRTUAL WORLD", para o desenvolvimento de funcionalidades que possibilitam aos alunos frequentar aulas num mundo virtual. Essas aulas são ministradas por um docente, representado nesse mundo virtual. Durante as aulas, os alunos têm à sua disposição dispositivos electrónicos controlados remotamente. O laboratório é uma reprodução virtual para aulas de física. Este projecto permite que mundos virtuais interajam com a plataforma de *e-learning*[26] Moodle[27]. Resulta de uma parceria entre a UNISUL (Universidade do Sul de Santa Catarina), UFRGS (Universidade Federal do Rio Grande do Sul), Faculdade SATC (Associação Beneficente da Indústria Carbonífera de Santa Catarina), do Brasil e o ISEP [Marcelino, R., et al, 2010].

2.3 Open Wonderland

À semelhança do OS, o Open Wonderland (OW) não é um mundo virtual propriamente dito, mas sim uma tecnologia para construção de mundos virtuais. Esta tecnologia deriva do anterior Project Wonderland, desenvolvido desde Dezembro de 2008 [Project Wonderland Roadmap and Release Estimates, 2007] pela mesma empresa criadora do Java, a Sun Microsystems, mas passou a OW desde que a Oracle adquiriu a Sun em Janeiro de 2010. A partir daí, por decisão da Oracle, a Sun deixou de suportar o Project Wonderland e formou-se a Open Wonderland Foundation com o objectivo de continuar o trabalho feito até então. Nesta altura o OW está na quinta versão, construído sobre as plataformas Project Darkstar e Java 3D. Como o OS, o OW é uma tecnologia gratuita.

Um mundo virtual concebido com o OW é acedido pela aplicação cliente Java Web Start (JWS) [Open Wonderland FAQ, 2009] (Figura 3), que está sempre disponível para *download* quando se navega para a página *Web* de um servidor OW. Essa aplicação pode ser usada num PC com os sistemas operativos Windows, Linux, MacOS ou Solaris desde que tenham a máquina virtual Java instalada. Para funcionar com um desempenho aceitável basta uma máquina (cliente) com um processador, com um mínimo de 1.5 GHz de velocidade de relógio (*clock*), 1 GB de memória RAM e uma placa gráfica que suporte OpenGL, com uma memória igual ou superior a 128 MB.

[26] *E-Learning*: metodologia de ensino não presencial, suportado por tecnologias de informação.
[27] Moodle é uma plataforma de software gratuita de apoio à aprendizagem.

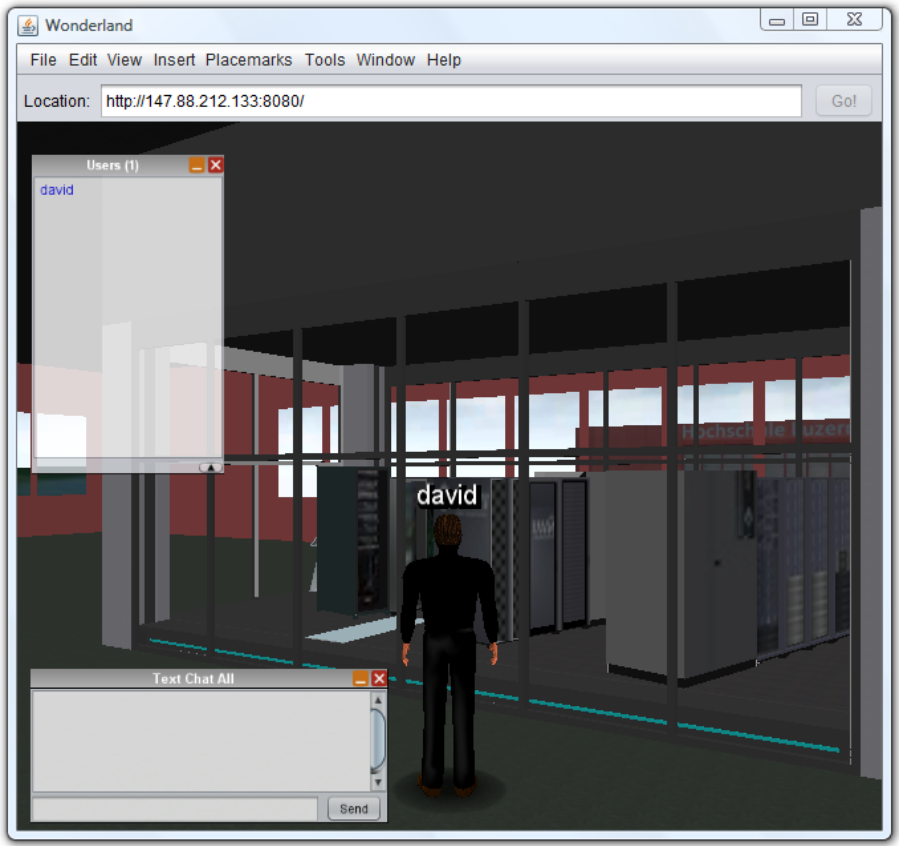


Figura 3 - Open Wonderland e Java Web Start

Para facilitar a adaptação a esta tecnologia, o OW dispõe de documentação [Open Wonderland Documentation Wiki, 2009] para administradores de rede, programadores, utilizadores e estudantes, de acordo com as funcionalidades que se destinam a cada um destes grupos. Como o OW foi construído em Java, apresenta uma API semelhante àquela.

Na página *Web* do OW existe uma secção onde são colocadas propostas de projectos a desenvolver nesta tecnologia [Open Wonderland Project Ideas, 2010]. Destinam-se essencialmente a estudantes, podendo ajudar a divulgação do OW.

Um projecto, que envolve laboratórios virtuais, desenvolvido com o OW, denomina-se Virtual Enterprise Lab (Figura 3) [VEL 2.0 - Virtual Enterprise Lab, 2011]. Reproduz num mundo virtual, um *campus* universitário, que inclui um laboratório, da Universidade de Ciências Aplicadas e Artes de Lucerna[28], na Suíça. Nesse *campus* virtual existem alguns quadros que auxiliam a utilização do mesmo. Não inclui interacção com dispositivos electrónicos, logo não existe qualquer controlo remoto.

Apesar de não ter sido desenvolvido usando o OW, o projecto TEAL Studio [Scheucher, T., et al, 2009] é aqui referido por basear-se no ambiente Project Wonderland (antecessor do OW). Este

[28] Lucerne University of Applied Sciences and Arts.

projecto é um laboratório virtual que permite controlar remotamente uma experiência relacionada com electromagnetismo, designada "Força num Dipolo". Nesta experiência remota, os alunos ou utilizadores podem visualizar graficamente no laboratório virtual o comportamento de um electroíman, quando aqueles lhe aplicam uma carga eléctrica. O TEAL Studio funciona sobre plataformas já existentes tais como: o iLab, uma ferramenta que possibilita o design e construção de mundos virtuais, e o TEAL Sim, que é o motor de simulação que permite o acesso remoto à experiência "Força num Dipolo". Todos estes projectos foram desenvolvidos no MIT.

2.4 Active Worlds

O Active Worlds (AW) (Figura 4) é semelhante ao SL, foi lançado em 1995 pela Activeworlds Inc. [World Review: Active Worlds, 2009]. É acedido através de uma aplicação cliente denominada Active Worlds Browser [Downloads, 2009], que pode-se instalar em qualquer computador, com SO Windows 98 ou mais recente, de acordo com os requisitos apresentados na tabela seguinte:

	Requisitos mínimos	**Requisitos recomendados**
Sistema Operativo	Windows 98, 2000, XP ou Vista	Windows 2000, XP ou Vista
Processador	Pentium 3 a 500 MHz	Pentium 4 a 1.6 GHz
RAM (MB)	≥128	≥512
Memória Gráfica (MB)	≥16	≥64

Tabela 2 - Requisitos mínimos do Active Worlds Browser

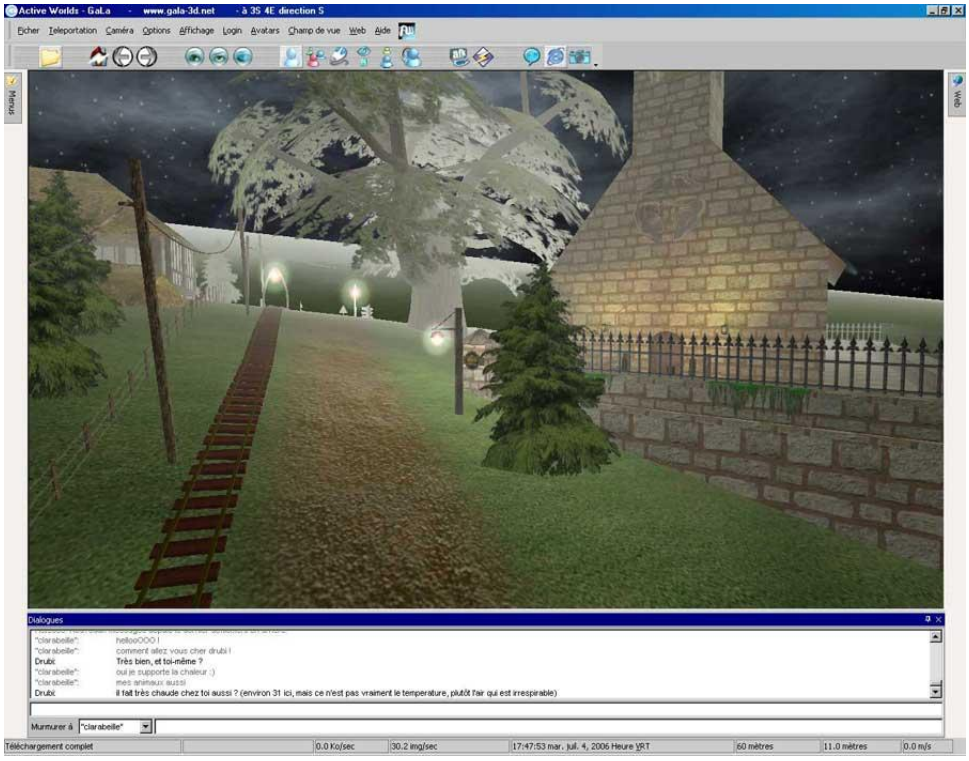


Figura 4 - Paisagem Active Worlds

Neste mundo virtual existem três tipos de utilizadores: *tourist* (turista), *citizen* (cidadão) e *World Owner* (dono de um mundo) [Become a Virtual World Citizen!, 2009]. O primeiro pode aceder somente a áreas públicas sem qualquer custo, o segundo pode aceder a todas as áreas públicas ou de acesso restrito, com um pagamento mensal de 6.95 USD, o terceiro é o dono de um mundo dentro do AW, tendo para isso que o construir e alojá-lo num servidor próprio. A obtenção de um mundo AW está sujeita a custos que incidem sobre a área do mundo virtual, a quantidade de utilizadores *citizen* e a função de incluir utilizadores do tipo *tourist* [World Servers Configuration and Pricing, 2009]. Para os três tipos de utilizador o acesso é feito através das credenciais nome e palavra-chave. O AW permite múltiplas configurações de *avatares* para todos os utilizadores, e é também possível a qualquer utilizador construir edifícios, embora com o risco de serem apagados caso tenham sido construídos na categoria *tourist*.

É possível a comunicação entre utilizadores através de texto ou voz. Existe um leque extenso de comunidades [Activeworlds Community Websites, 2009] desde fóruns, *wikis*, *newsletter*[29], escola, etc. Várias cidades do AW também formam as suas próprias comunidades.

Para construção, desenvolvimento ou programação de mundos virtuais, o AW disponibiliza um pacote de desenvolvimento de software (SDK[30]), que consiste numa biblioteca de funções. A programação para o AW é feita através da linguagem C [Introduction to the Active Worlds SDK, 2009]. A SDK é gratuita e sujeita a licença da Activeworlds Inc. Apesar da aplicação cliente Active Worlds Browser apenas funcionar em sistemas Windows, é possível obter um servidor AW e respectiva SDK para sistemas Windows e Linux. A obtenção de um servidor AW é gratuita sob licença da Activeworlds Inc. É através do alojamento no servidor que se é dono de um mundo AW.

Utilizando mundos virtuais AW, existe um laboratório remoto, designado *Vetunimi* [GANDINI, C., 2001], direccionado para o ensino de medicina veterinária. É italiano, e reproduz virtualmente um laboratório real, que inclui o controlo remoto de dispositivos electrónicos, para análises biomédicas[31].

Actualmente o AW não é tão atractivo como o SL, principalmente pelo seu aspecto gráfico mais antiquado, daí ser menos conhecido para a maioria dos utilizadores.

2.5 There

Outro mundo virtual semelhante ao SL e ao AW é o There, lançado em 2003 pela empresa There Inc. O acesso também é feito através de uma aplicação cliente denominada There software, com

[29] *Newsletter* é um boletim informativo de publicação periódica, sobre um determinado assunto ou temática.
[30] SDK: conjunto de ferramentas, que pode incluir código, documentação ou aplicações para desenvolvimento de software.
[31] Derivam da biomedicina, que é uma ciência cujo campo de pesquisa centra-se entre a medicina e a biologia.

credenciais semelhantes às dos utilizadores do AW. A mesma aplicação apenas executa nos sistemas operativos Windows 2000, XP e Vista com os requisitos apresentados na tabela seguinte:

	Requisitos mínimos	**Requisitos recomendados**
Ligação Internet (kbits/ segundo)	≥56	
Sistema Operativo	Windows 2000, XP ou Vista	
Processador	Pentium 3 800 MHz ou Pentium 4 2.3 GHz	Pentium 4 ≥ 2.4 GHz
RAM (MB)	≥ 256	
Placa Gráfica/Memória Gráfica (MB)	NVIDIA ou ATI 7200 ≥32 MB	NVIDIA ou ATI 7000 ≥32 MB

Tabela 3 - Requisitos mínimos do There software

Existem dois tipos de utilizadores: *Basic* e *Premium*. O primeiro é gratuito, e possibilita a configuração do seu *avatar* associado, desde sexo a roupas, comunicação com outros utilizadores através de texto, ouvir rádio, e ainda participar em compras e leilões. O segundo tipo de conta acrescenta comunicação por voz, teletransporte, organização de eventos, criação de bairros para alugar a outros utilizadores, venda de roupas de *avatar* a outros utilizadores, carros, entre outros objectos. O custo de uma conta *Premium* implica um pagamento único no momento de registo de 9.95 USD.

À semelhança do SL, o There também possui uma economia interior com a moeda Therebuck (Tbux), onde 1 USD = 1,8 Tbux. As actividades que se destacam no There são festas sociais (Figura 5), corridas de *buggy*, *hoverboards* e *paintball*. O comércio consiste basicamente na compra de objectos de *avatar*, veículos e casas. Para suporte a todos os utilizadores, o There disponibiliza ajuda numa página *Web*.

Figura 5 - Festa no mundo virtual There

Em relação a ferramentas para programação, no There resume-se à manipulação de modelos 3D. Para aceder a informação para programação, é necessário ser utilizador do mundo virtual. A programação para o There tem um custo associado, com os montantes a depender do que se pretende conceber.

Na revisão dos ambientes 3D, não se encontrou qualquer referência a laboratórios virtuais ou remotos, construídos num mundo virtual There. Esta tecnologia destaca-se por mundos virtuais, direccionados essencialmente para actividades lúdicas (divertimento).

Este ambiente 3D foi descontinuado em 9 de Março de 2010 por dificuldades financeiras devido à crise económica mundial [There.com closed on March 9th, 2010].

2.6 VastPark

Tal como o OS e o OW, o VastPark (VP) não se trata de um mundo virtual já construído, mas de uma tecnologia que permite a construção e o alojamento de mundos virtuais em servidores. Foi lançado pela empresa australiana VastPark em 2007, apesar do seu desenvolvimento ter começado em 2003. É uma tecnologia gratuita e divide-se em quatro ferramentas essenciais:

- **VastPark Player:** aplicação cliente de acesso a mundos virtuais VP, funciona no SO Windows XP SP2 ou mais recente. Requer a plataforma .NET Framework[32], e uma placa gráfica que suporte DirectX 9c ou mais recente. Está prevista uma interface via *browser* denominada Web Player.
- **VastPark Creator:** ferramenta integrada na SDK do VP, para desenho (Figura 6) de ambientes 3D através da linguagem IMML (Immersive Media Markup Language). É uma linguagem de marcação semelhante ao XML[33], mas com as suas próprias *tags*[34]. Os requisitos de funcionamento são os mesmos do VastPark Player.
- **VastPark Publisher:** facilita a transferência de conteúdo de forma a este estar disponível na Internet. Actualmente o conteúdo é armazenado através de um *Web service*[35] da Amazon denominado Simple Strorage Service, que permite o armazenamento de dados *online*. Para tirar partido do VastPark Publisher é necessário ter uma conta de utilizador dos Amazon Web Services, que é um serviço pago. É também necessário um computador com o SO Windows XP SP2 ou mais recente, .NET Framework 3.5 e uma placa gráfica que suporte DirectX 9c ou mais recente. Esta ferramenta também está integrada na SDK do VP.
- **VastPark Server:** aplicação servidora que aloja mundos virtuais VP, lê ficheiros IMML escritos na linguagem de marcação com a mesma sigla, e torna esse conteúdo visível à aplicação cliente. O VP pode conter um número ilimitado de mundos virtuais, dependendo do hardware disponível e da largura de banda da ligação de rede. Os requisitos de sistema para o VastPark

[32] .NET Framework: plataforma de construção e execução de aplicações informáticas, desenvolvida pela Microsoft.
[33] XML: linguagem de marcação estruturada, usada para facilitar a leitura de uma determinada estrutura de dados por aplicações informáticas.
[34] Uma *tag* é uma instrução de código de um ficheiro XML.
[35] *Web service*: tecnologia que permite a integração de sistemas informáticos entre aplicações diferentes.

Server funcionar correctamente, são os mesmos do VastPark Publisher, com a diferença de requerer no mínimo a .NET Framework 3.0.

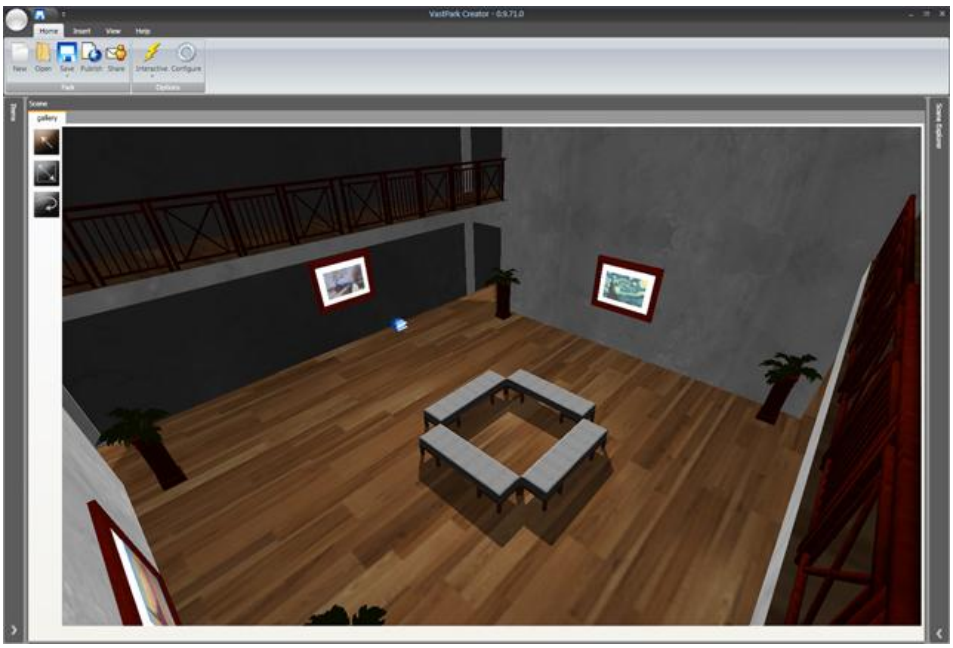


Figura 6 - Desenho de uma sala no VastPark Creator

A programação de um mundo virtual é feita em IMML, para criar interfaces. O seu funcionamento interno é programado na linguagem Vastscript, que baseia-se na linguagem Lua [Lerusalimschy, R.; et al, 2010], com uma sintaxe semelhante à linguagem C.

Em relação a comunidades, o VP disponibiliza fóruns, documentação, tutoriais e uma secção para programadores.

O VP expande-se gradualmente, através de parcerias [The Project Factory Develops in VastPark, 2008] com outras empresas, que pretendam construir o seu mundo virtual usando esta tecnologia.

Não foi encontrada qualquer referência a laboratórios virtuais ou remotos, em mundos virtuais VP. Este centra as suas atenções, em mundos virtuais para auxílio de tarefas empresariais [Enterprise Virtual Worlds, 2011].

2.7 3DXplorer

3DXplorer é simultaneamente uma tecnologia de desenvolvimento de mundos virtuais e um ambiente 3D propriamente dito. Foi introduzido pela empresa Altadyn em 2004. Destaca-se dos mundos virtuais mais conhecidos, por dispensar aplicação cliente de acesso, sendo o 3DXplorer acedido directamente através de um *browser* (Figura 7). Para ser possível navegar no mesmo, [What is 3DXplorer ?, 2011] é

necessário, no mínimo, a máquina virtual Java 1.5 instalada. Os seus requisitos de funcionamento para sistemas Windows, MacOS e Linux são mencionados na tabela seguinte:

	Windows	**MacOS**	**Linux**
Ligação Internet	Cabo ou DSL		
Sistema Operativo	Windows 2000, XP ou Vista	Mac OSX ≥10.3.9	Qualquer distribuição recente
Processador	Pentium 3, Athlon a 800 MHz ou superior		
RAM (MB)	≥ 512		
Placa Gráfica	Qualquer placa gráfica actual		

Tabela 4 - Requisitos mínimos do 3DXplorer

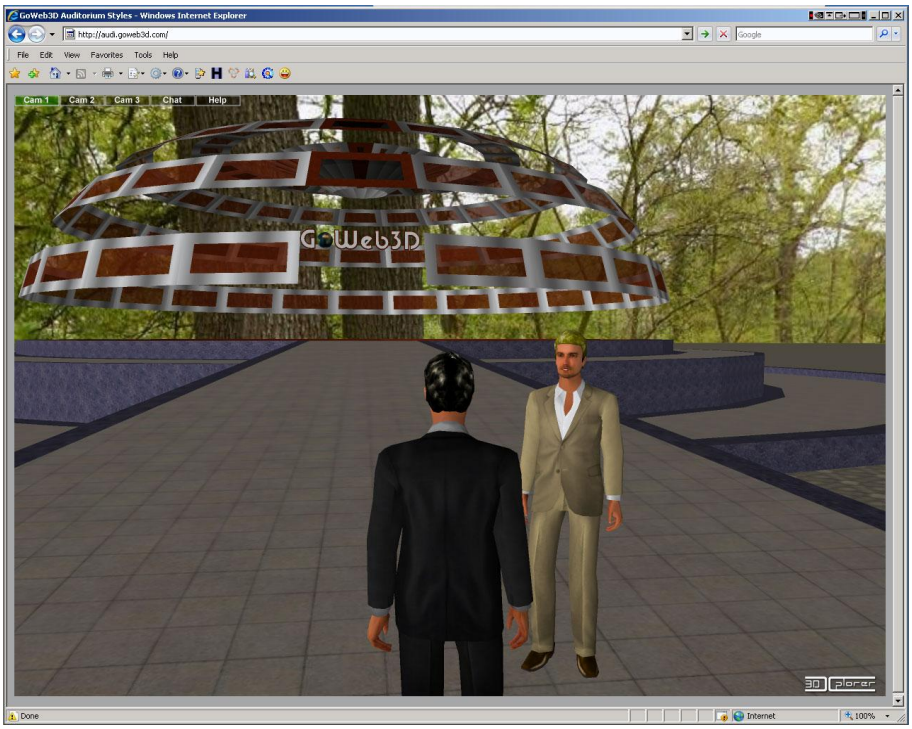


Figura 7 - 3DXplorer num *browser*

Nos mundos virtuais como o SL, AW ou There, tem-se um universo virtual com vários mundos ou regiões diferentes. A abordagem do 3DXplorer é ligeiramente diferente, sendo que são logo disponibilizados diferentes mundos virtuais, já construídos e pagos. Esses mundos virtuais constituem-se como produtos 3DXplorer, em que todos permitem criar diferentes cenários 3D com a ferramenta 3DX Studio, e configurar *avatares*. Os mesmos mundos virtuais são basicamente para serem utilizados em ambientes empresariais, e podem estar alojados tanto nos servidores da Altadyn como em servidores das próprias empresas cliente. Os diferentes mundos virtuais 3DXplorer são:

- **3DXplorer Online Meeting:** ambiente 3D capaz de agrupar até 10 utilizadores em simultâneo para encontros, reuniões ou conferências em espaço próprio. É possível escolher o tipo de espaço em redor, como uma sala ou anfiteatro. Podem ser feitas várias apresentações em simultâneo, em ecrãs virtuais para o efeito. Existe comunicação por texto ou voz. A utilização

deste mundo virtual tem um custo de 49 USD ou 490 USD, respectivamente se for mensal ou anual.

- **3DXplorer Meet-in-3D:** mundo virtual que simula um ambiente 3D de uma loja, onde existe interacção entre empresário e cliente. Permite o convite de outros utilizadores, potenciais clientes para esse espaço. Faz estatísticas de quem visita o espaço, quantas vezes visita e durante quanto tempo visita. Tem a particularidade de possibilitar, a configuração do espaço e do leque de produtos em exposição. Tem um custo mensal de 99 USD ou anual de 990 USD, limitado a 20 utilizadores ligados em simultâneo. Caso pretenda-se um número ilimitado de utilizadores em simultâneo, o custo sobe para 4990 USD anuais.
- **3DXplorer Virtual Conferencing:** mundo virtual com as mesmas características do anterior, mas acrescenta número ilimitado de utilizadores presentes no mesmo espaço, comunicação por voz, divisão dos utilizadores em duas categorias: VIP ou convidado. Estas categorias consistem na resolução gráfica dos *avatares*, maior resolução fica na categoria VIP e menor fica na categoria convidado. Tem um custo mensal de 490 USD, ou um custo anual de 4990 USD. Permite um máximo de 100 utilizadores ligados em simultâneo.
- **3DXplorer Enterprise Edition:** combina todas as características dos três mundos anteriores, sendo possível decidir ter ou não, qualquer uma das funcionalidades acima mencionadas, além de permitir o uso de uma API para programação. O custo depende das funcionalidades que forem utilizadas.

Em todos os mundos virtuais anteriores, à excepção do 3DXplorer Enterprise Edition, existe a limitação do tráfego máximo permitido, ou seja, o mundo virtual só pode ter um máximo de 1000 acessos por dia.

Num nível mais básico o 3DXplorer disponibiliza um produto denominado 3DXplorer Platform, completamente gratuito, que permite a configuração do espaço em redor e a configuração de *avatar*, com o 3DXStudio.

Para terem acesso aos mundos virtuais 3DXplorer, os utilizadores precisam de se autenticar com o seu nome e palavra-chave, sendo que em alguns mundos é possível entrar como convidado, apenas com o nome, mas neste caso não é possível configurar o *avatar*. No 3DXplorer em vez de haver tipos de conta de utilizador, existem planos de tráfego de acordo com o número de acesso que um utilizador de um mundo virtual pretende fazer [Membership Plans, 2009]. O acesso é gratuito até um número máximo de 200 acesso por dia, depois o número de acessos pode variar desde os 500 até aos 2000 com um custo entre 150 USD até 900 USD. É possível configurar um plano de acessos diários, mas com preço sob consulta. Caso se preveja que o número de acessos diários vai ser ultrapassado, é possível adquirir *packs* (pacotes) adicionais de tráfego, a variar desde os 5000 até aos 100000 acessos, com os preços a variarem desde os 225 até aos 2250 USD. O limite de tráfego é adquirido pelas entidades que

detêm um determinado mundo virtual. O número de acessos diários é reinicializado todos os dias às 0:00 GMT[36].

A API do 3DXplorer é proprietária e construída à base de Java, Javascript[37] e PHP[38] [3DXplorer API, 2009].

Os mundos virtuais 3DXplorer são claramente pensados como aplicações para apoio a negócios ou actividades empresariais. Durante a revisão dos ambientes 3D, não foram encontradas referências à sua utilização em laboratórios virtuais ou remotos.

2.8 Análise comparativa

Depois da análise dos diferentes ambientes 3D, é feita uma análise comparativa (Tabela 5), tendo em conta critérios considerados fundamentais para construção e acesso a um laboratório virtual. Ao planear um laboratório virtual, para estar disponível a vários utilizadores, tem de se pensar na forma como este é acedido:

- **Aplicação cliente:** é o software que permite que um utilizador aceda ao mundo virtual ou ambiente 3D, que pode ser através de *download* e instalação, ou apenas *download* de um simples ficheiro executável. Verifica-se que todos os ambientes 3D analisados, à excepção do 3DXplorer, necessitam de uma aplicação cliente para serem acedidos, o que implica tarefas adicionais de como descarregar ou instalar a aplicação, antes de se utilizar o mundo virtual propriamente dito.
- ***Browser*:** este critério serve para averiguar se um ambiente 3D pode ser acedido directamente através de um *browser* ou página *Web*. Apenas o 3DXplorer apresenta essa funcionalidade, o que dispensa o *download* ou instalação de software adicional, estando o ambiente logo acessível no momento em que se acede à sua página *Web*.

Para se ter um laboratório virtual num ambiente 3D é necessário construí-lo, por isso é fundamental que se conheça ou se tenha acesso à sua API ou SDK, para perceber como construir/programar novos conteúdos ou adicionar/modificar funcionalidades. Os próximos dois critérios são a verificação de se uma API é gratuita ou proprietária, pois este factor é decisivo quando se pretende utilizar o ambiente 3D, para depois construir conteúdos adicionais ao menor custo possível.

- **API proprietária:** os ambientes SL, There e 3DXplorer possuem API proprietária, o que implica custos caso se pretenda construir conteúdos para os mesmos.
- **API gratuita:** com uma API gratuita a tarefa de construir conteúdos é facilitada, pois o acesso às suas funcionalidades é gratuito e imediato como é o caso dos ambientes OS, OW, AW e VP.

[36] GMT: define a contabilização do fuso horário a partir do semi-meridiano de Greenwich.
[37] Javascript: linguagem de programação de páginas *Web*, do lado do cliente.
[38] PHP: linguagem de programação para conteúdos dinâmicos para páginas *Web*.

A usabilidade de um ambiente 3D sai melhorada pela possibilidade dos seus utilizadores comunicarem entre si. As formas de comunicação que os ambientes 3D analisados apresentam, são à base de texto, voz e vídeo. O bom trabalho num laboratório real também depende de uma boa comunicação, por isso é importante que o laboratório virtual também disponha de pelo menos uma forma de comunicação entre os utilizadores.

- **Comunicação por texto:** todos os ambientes 3D têm comunicação por texto entre utilizadores.
- **Comunicação por voz:** à excepção do OS, todos os ambientes permitem comunicação por voz.
- **Comunicação por vídeo:** apenas o SL e o OW dispõem de comunicação por vídeo utilizando uma *webcam*[39].

Um laboratório de electrónica é constituído por instrumentos de medição, como multímetro, fonte de alimentação contínua[40], osciloscópio[41] e afins. Para os mesmos serem acedidos a partir de um laboratório virtual, é vital implementar os dispositivos virtuais para controlarem os reais, a partir das funcionalidades da API/SDK disponibilizada, pois nenhum dos ambientes 3D apresenta um laboratório virtual de raiz, nem tem funcionalidades específicas para comunicação com dispositivos electrónicos. O planeamento para estabelecer esta comunicação, deve ser feito tendo em conta, quais os protocolos de rede que um dispositivo electrónico utiliza para ser remotamente acedido, a informação que o dispositivo pode retornar quando acedido, e se a API/linguagem de programação de um ambiente 3D tem funções ou bibliotecas de funções que trabalhem com protocolos de rede, para um dispositivo virtual poder comunicar com um real.

	Second Life	**OpenSim**	**Open Wonderland**	**Active Worlds**	**There**	**Vastpark**	**3DXplorer**
Aplicação cliente	Second Life Viewer	Second Life Viewer/ Hippo Viewer	Java Web Start	Active Worlds Browser	There Software	VastPark Player	
Web browser							✓
API proprietária	✓				✓		✓
API gratuita		✓	✓	✓		✓	
Comunicação por texto	✓	✓	✓	✓	✓	✓	✓
Comunicação por voz	✓		✓	✓	✓	✓	✓
Comunicação por vídeo	✓		✓				

Tabela 5 - Análise comparativa dos mundos virtuais

[39] Uma *webcam* é uma câmara de vídeo, que transfere a imagem captada para um computador.
[40] Uma fonte de alimentação contínua transforma energia eléctrica alternada para contínua, para alimentar circuitos que funcionem com energia eléctrica contínua.
[41] Um osciloscópio fornece uma visualização gráfica do comportamento da energia eléctrica num circuito ao longo do tempo.

Os ambientes 3D revistos podem-se dividir em dois grupos: os que oferecem mundos virtuais já construídos e os que inicialmente são só servidores aplicacionais de mundos virtuais. Do primeiro grupo fazem parte os ambientes: SL, AW, There e 3DXplorer. Este tipo de ambientes 3D foi concebido por empresas com fins lucrativos, sendo que o SL é o mais abrangente, já que oferece soluções empresariais, lúdicas e educativas. O AW é o que mais se assemelha ao SL, mas foi ficando mais discreto com o tempo devido à expansão do SL. Os ambientes There e 3DXplorer têm um público-alvo mais específico, sendo o primeiro para actividades lúdicas e o segundo para suporte de negócio. Em todos os ambientes do primeiro grupo existem custos de utilização, podem existir custos de construção como no SL e AW, que implicam a aquisição de uma área para construir conteúdos. Nos ambientes There e 3DXplorer, os custos de construção de conteúdos não se relacionam com a aquisição de área para construção, mas pela aquisição de conteúdos pré-estabelecidos, que devidamente combinados formam a construção do mundo virtual pretendido. Todos os ambientes 3D deste grupo têm API/SDK com custos associados, com excepção do AW. Do segundo grupo de ambientes 3D fazem parte o OS, OW e VP, todos eles não oferecem mundos virtuais de raiz, ou seja, têm que ser construídos. A sua construção requer que os servidores aplicacionais sejam devidamente instalados, configurados e a funcionar numa máquina disponível na Internet, daí as suas API's serem gratuitas, o que ajuda a sua divulgação. Neste grupo apenas o VP parece caminhar para oferecer mundos virtuais de raiz para fins empresariais.

Um pormenor importante neste capítulo, é que todos os ambientes 3D que apresentaram mundos virtuais com fins educativos, tinham laboratórios virtuais/remotos. Um laboratório é sempre um espaço importante numa escola ou universidade, para auxiliar a aprendizagem dos alunos. Os ambientes 3D que apresentaram laboratórios virtuais ou remotos foram: SL, OS, OW e AW, todos eles tinham dispositivos electrónicos disponíveis para serem controlados remotamente (no caso do OW o controlo remoto desses dispositivos é disponibilizado pelo projecto TEAL Studio, concebido com o Project Wonderland).

O estudo e comparação das características dos ambientes 3D revistos, vai determinar qual o ambiente 3D utilizado, numa solução a desenvolver que cumpra os objectivos referidos no capítulo inicial.

3. Solução a desenvolver

A selecção do ambiente 3D é o ponto de partida para a solução a desenvolver, por isso é necessário ter critérios de selecção bem definidos, pois isso influenciará todo o trabalho desta tese. O desenvolvimento da solução proposta não se cinge apenas à selecção de um ambiente 3D. Depois dessa selecção deve-se planear uma infra-estrutura informática capaz de disponibilizar o laboratório virtual a vários utilizadores que pretendam efectuar medições no multímetro remotamente, a partir da sua representação virtual. Para isso, o multímetro deve possuir conexões que possibilitem a sua ligação a uma rede de computadores. No cenário virtual do ambiente 3D, deverão existir funcionalidades que auxiliem a utilização clara e ordenada da reprodução virtual do multímetro. Esse cenário ou mundo virtual constitui a interface entre o utilizador e o instrumento de medição no laboratório real.

3.1 Ambiente 3D seleccionado

De entre os ambientes 3D revistos no capítulo anterior, a escolha recaiu sobre o OW tendo em conta os seguintes critérios:

- **API gratuita**: uma tecnologia que disponha de uma API gratuita permite que se possam usar certas funções já implementadas de raiz, para produzir conteúdos sem qualquer custo de aquisição ou de utilização. Usando essas funções, poupa-se tempo de desenvolvimento, já que não é necessário implementá-las. Todo o conteúdo de uma API gratuita é normalmente de fácil leitura e acesso, pois estes factores facilitam a sua divulgação.
- **Construção em Java:** como o OW está construído em Java, a sua programação também é feita nesta linguagem. Um ponto forte da linguagem Java é sua facilidade de programação, onde se destaca a sua fácil adaptação inicial, mesmo para um programador menos experiente. Para esta tese em concreto, a linguagem Java apresenta a mais-valia de existirem vários projectos que constroem API's gratuitas, cujas funções não estão incluídas de raiz na API do Java. Por exemplo, e para esta tese, é decisivo que existam projectos que disponibilizem uma API para programação 3D e comunicações de rede, utilizando os protocolos SSH[42] e *telnet*[43]. Isso facilitará a tarefa de desenho de objectos 3D, e poupará muito tempo quando se implementar/programar o mecanismo de comunicação entre o multímetro e a sua reprodução virtual.
- **Acesso via *browser*:** apesar do ambiente 3D seleccionado não ser acessível directamente via *browser*, o *download* da sua aplicação cliente JWS, faz-se acedendo a uma página *Web* de um determinado servidor OW. Essa página *Web* é de fácil leitura e utilização, mesmo um utilizador

[42] SSH: protocolo que estabelece uma ligação segura a um computador de forma a executar comandos remotamente no mesmo.
[43] *Telnet*: protocolo com a mesma funcionalidade do anterior, mas com uma ligação insegura.

menos experiente conseguirá fazer o *download* da aplicação cliente, depois de visualizar a página. Toda a administração/gestão de um servidor OW é feita numa página *Web*, cujas interfaces facilitam as tarefas de um administrador/utilizador, no sentido em que basta um *browser* para as realizar, dispensando assim a instalação de software adicional.

- **Documentação bem organizada e acessível:** a documentação do OW encontra-se na sua página *Web*, destaca-se pelo seu modelo de organização e acessibilidade. O seu modelo de organização consiste numa separação de documentação específica para utilizadores, administradores e programadores, toda esta documentação tem o objectivo de ensinar a tirar a partido das diferentes funcionalidades do OW, não dando muita ênfase a aspectos meramente técnicos. Por exemplo, é relativamente fácil para um programador com alguma experiência em Java construir um conteúdo para um mundo virtual, depois de uma leitura de cerca de trinta minutos da documentação destinada a programadores. A acessibilidade à documentação na página *Web* do OW é facilitada através de menus de navegação, ou de *links* bem visíveis e destacados.
- **Arquitectura modular:** depois de uma leitura da documentação para programadores do OW, verifica-se que grande parte dos conteúdos adicionais para um mundo virtual, baseiam-se em módulos. Assim, para a construção de uma reprodução virtual de um dispositivo electrónico, basta construir o módulo correspondente a essa reprodução.
- **Facilidade de acesso a exemplos existentes:** a documentação do OW apresenta vários exemplos que podem servir de base à configuração, administração e construção de conteúdos adicionais, facilitando assim a adaptação inicial a este ambiente 3D.

A próxima etapa, depois de seleccionar o ambiente 3D, é estudar as funcionalidades do multímetro, para compreender como este pode ser acedido/utilizado remotamente.

3.2 Instrumento de medição seleccionado

Num laboratório de electrónica, o multímetro é dos instrumentos de medição mais básicos e fáceis de utilizar. Um multímetro serve para efectuar medições de várias variáveis eléctricas, tais como tensão, corrente, resistência, frequência, período, etc. As mais conhecidas são a tensão e a corrente, a primeira medida em volts (V) e a segunda medida em amperes (A). O modelo do multímetro seleccionado é um Fluke 8845A, com um visor de seis dígitos e meio, e com vários conectores para ligações remotas, que o tornam acessível a determinado hardware ligado a uma rede de computadores, ou ligado directamente ao dispositivo. Nas próximas figuras (Figura 8, 9 e 10) são mostradas as várias perspectivas ou vistas do multímetro.

Figura 8 - Vistas superior e lateral esquerda/direita do multímetro

Figura 9 - Vista frontal do multímetro

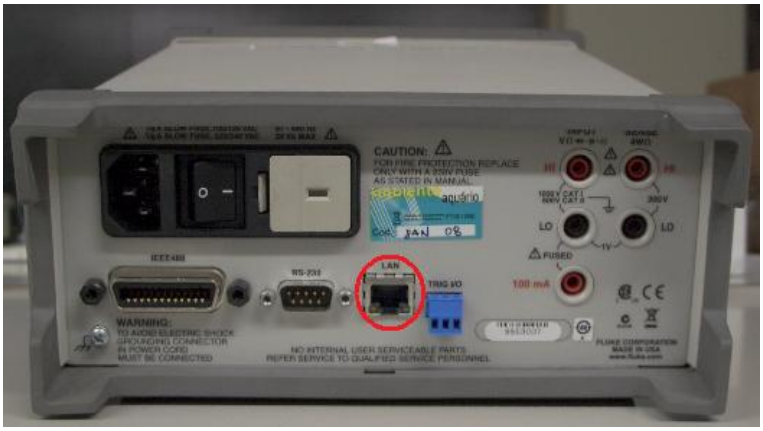


Figura 10 - Vista traseira do multímetro

Todo o multímetro é operado pelos botões da vista frontal na Figura 9. O dispositivo pode ser ligado ou desligado no botão no canto inferior direito da parte frontal. Para além deste, os botões mais usados são: DCV, ACV, DCI e ACI para medições de tensão ou corrente tanto com energia eléctrica do tipo AC[44] ou DC[45]. Na Figura 11 apresenta-se o multímetro ligado, com um valor de tensão contínua no seu visor, que corresponde à medição por omissão, que fica activa após se ligar o dispositivo, ou quando o botão DCV é pressionado.

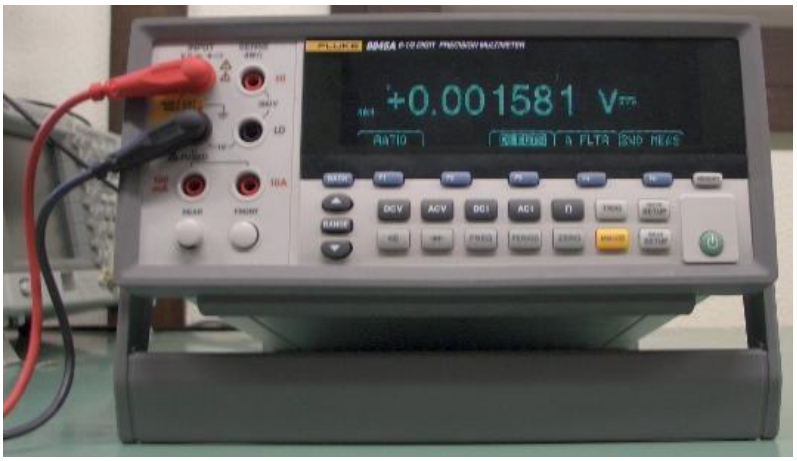


Figura 11 - Multímetro ligado

Para efectuar medições de variáveis eléctricas, o multímetro tem que ter os cabos vermelho e preto (canto superior esquerdo do dispositivo na Figura 11), ligados a pontos de um circuito eléctrico. Caso esses cabos não estejam ligados a um circuito eléctrico, os valores medidos correspondem a ruído eléctrico[46].

Ao observar os conectores da parte traseira do multímetro (Figura 10), verifica-se que o dispositivo pode ser acedido/utilizado remotamente, através de uma rede de computadores, usando o conector *Ethernet* LAN[47] sinalizado na Figura 10. Quando o multímetro é ligado fisicamente a uma rede *Ethernet*, com o seu endereço IP e porto[48] de acesso devidamente configurados (no autocolante na parte lateral esquerda do dispositivo na Figura 8, estão registados os valores do endereço IP e o porto de acesso para o multímetro), fica acessível a outros dispositivos que se liguem à sua rede, para poderem efectuar medições remotamente, desde que o mesmo esteja ligado. O acesso remoto mais comum costuma ser a partir de um PC[49]. O multímetro, ao estar ligado a uma rede *Ethernet*, é acedido remotamente por outros dispositivos, através do protocolo *telnet*. O Comando 1 exemplifica uma ligação *telnet* entre um PC e o multímetro, executado numa linha de comandos de um SO Linux para estabelecer a ligação.

[44] AC: abreviatura usada para indicar energia eléctrica alternada.
[45] DC: abreviatura usada para indicar energia eléctrica contínua.
[46] Considera-se ruído eléctrico qualquer medição indesejada ou inútil no circuito.
[47] LAN: designação para rede local ou interna de comutadores.
[48] Porto: ligação virtual na transmissão de dados em redes de computadores. Também pode ser designado de porta.
[49] PC: computador de uso pessoal, termo aplicado aos computadores mais usuais.

```
telnet 192.168.10.31 1150
```

Comando 1

Se o comando anterior for executado com sucesso, o utilizador acede à interface de controlo remoto do multímetro, conforme mostra o *Output* 1.

```
laboris@laboris-salaestudo:~$ telnet 192.168.10.31 1150
Trying 192.168.10.31...
Connected to 192.168.10.31.
Escape character is '^]'.
```

***Output* 1**

A interface para um utilizador controlar remotamente o multímetro, baseia-se numa linha de comandos semelhante à dos sistemas operativos. A partir do cursor do *Output* 1, o utilizador pode inserir os comandos para o multímetro efectuar as medições pretendidas. O controlo remoto do dispositivo baseia-se em comandos SCPI[50] [Fluke 8845A/8846A Digital Multimeter Programmers Manual, 2010], específico para este tipo de dispositivos. Para indicar ao multímetro que vai ser operado remotamente usa-se o comando *syst:rem*. Depois, já é possível efectuar as medições propriamente ditas, por exemplo, uma medição de tensão contínua irá apresentar os resultados do *Output* 2 e da Figura 12.

```
syst:rem
meas:volt:dc? 1
+1.17330000E-03
```

***Output* 2**

Figura 12 - Multímetro com medição remota de tensão contínua

Uma medição remota de tensão contínua faz-se com o comando *meas:volt:dc? 1*, depois do comando *syst:rem*, que activa o multímetro para receber comandos remotamente. No *Output* 2, o valor no comando de medição da tensão contínua, indica que o valor máximo estipulado para a medição é 1 volt, não podendo ser ultrapassado. Os valores numéricos depois do comando de medição (*Output* 2) e

[50] SCPI: padrão de comandos para programar ou operar dispositivos electrónicos que efectuam medições.

no visor do multímetro (Figura 12), correspondem ao valor de tensão contínua no momento de medição. Esses valores são matematicamente o mesmo, mas apresentados de forma diferente, por exemplo, o valor numérico do *Output* 2 é apresentado em notação científica e o valor da Figura 12 é apresentado em notação decimal. Numa ligação remota ao multímetro, este retorna sempre os valores de medição em notação científica para a linha de comandos. Isso constitui uma das dificuldades a superar, para conseguir ter um valor de medição no visor do multímetro, no mundo virtual, o mais semelhante possível ao do visor do multímetro da Figura 12.

Para desactivar a ligação remota entre um utilizador num PC e o multímetro (também se pode designar de modo de funcionamento remoto), existem duas formas: a primeira de uma forma remota, com a execução do comando *syst:loc*, a segunda, carregando localmente na tecla F1, que corresponde ao separador "Local" (Figura 12) no visor. Assim, o dispositivo fica em modo de funcionamento local, semelhante ao apresentado na Figura 11.

Este multímetro suporta uma ligação *telnet* de cada vez, ou seja, não é possível dois ou mais utilizadores estarem ligados em simultâneo ao dispositivo.

3.3 Arquitectura física

Fisicamente, a solução proposta para esta tese encontra-se ilustrada na Figura 13.

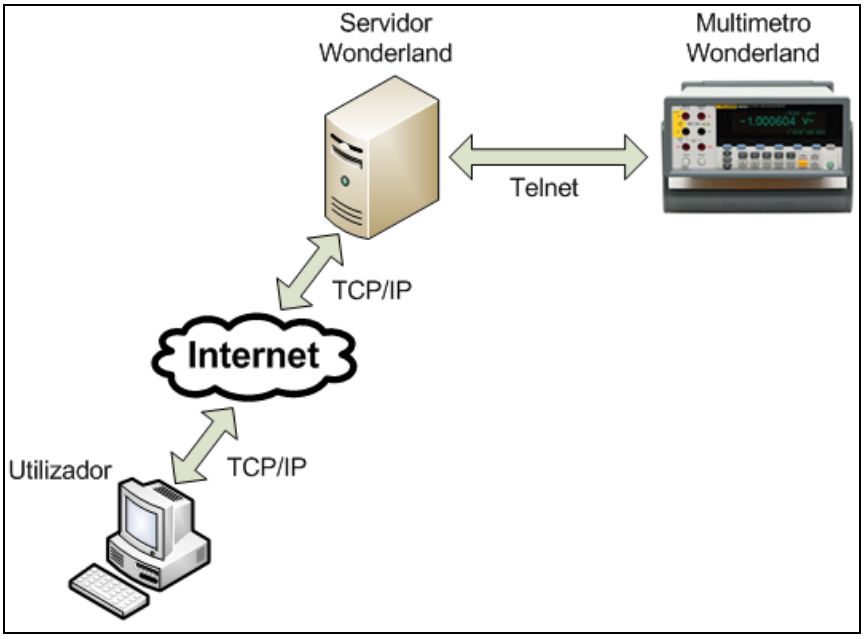


Figura 13 - Arquitectura física da solução

Na figura anterior, verifica-se que a base da solução proposta consiste num ou mais utilizadores ligarem-se à máquina Servidor Wonderland (SW) através da Internet (ligação TCP/IP), para utilizarem o mundo virtual, ou seja, os utilizadores usam o OW através da aplicação cliente JWS. É na máquina

SW que o servidor aplicacional do OW executa. Para o utilizador conseguir operar remotamente o dispositivo Multimetro Wonderland (MW) num mundo virtual, este deve estar ligado a uma rede *Ethernet*. Teoricamente, o MW poderia ser ligado directamente à máquina SW usando outras conexões para além da *Ethernet*, mas isso dificultaria o acesso remoto ao mesmo, levando a estudar formas alternativas de comunicação no SO da máquina SW, o que poderia resultar numa utilização remota do MW restrita à máquina SW. A mesma acede ao MW, via *telnet*, ao estar na mesma rede privada do mesmo, ou seja, tanto o SW como o MW encontram-se na rede interna do CIETI-LABORIS, mas outras máquinas/dispositivos nesta rede também podem aceder remotamente ao MW, desde que usem o Comando 1. Nesse caso, deverá haver a precaução para não aceder ao MW enquanto este está acessível aos utilizadores de um mundo virtual, pois se o MW estiver a ser acedido por outro dispositivo na rede do CIETI-LABORIS, fica inutilizável para os utilizadores do mundo virtual. Estando o MW ligado a uma rede *Ethernet*, tecnicamente seria acessível do exterior, mas assim, o MW poderia ser acedido por qualquer utilizador que executasse o Comando 1, sem necessidade de estar no mundo virtual, que é uma situação que não se pretende para esta solução. Os utilizadores podem-se ligar a um ambiente ou mundo virtual, devido à máquina SW estar acessível ao exterior (Internet) da rede do CIETI-LABORIS, dado que nesta rede existe um mecanismo de redireccionamento de portos. Isto é, quando um utilizador especifica o porto de acesso da máquina SW no endereço principal (URL[51]) do CIETI-LABORIS, ele acede logo a essa máquina. Este mecanismo permite poupar endereços IP públicos.

Basicamente, a arquitectura física da solução final constitui a infra-estrutura informática, que permite ao utilizador fazer o *download* do mundo ou ambiente virtual, ligando-se à máquina SW. Quando aquele pretender efectuar remotamente uma medição no multímetro, executa uma acção, na sua reprodução no mundo virtual, que faz executar na máquina SW uma aplicação que executa medições no dispositivo MW. O valor de medição é retornado do multímetro para a máquina SW, e desta, para a representação virtual do multímetro no PC do utilizador.

A máquina SW é um PC com um processador AMD Sempron com uma velocidade de relógio de 1.6 GHz, 1 GB de memória RAM, e 66 GB de disco, com uma ligação *Ethernet* de 100 Mb/s. A página *Web* do OW indica que os requisitos mínimos para uma máquina suportar um servidor OW, são os mesmos para suportar a aplicação cliente JWS, ou seja, verifica-se que de acordo com as especificações da máquina SW e com os requisitos mínimos referidos na secção 2.3, a mesma pode executar um servidor OW. O funcionamento da aplicação cliente JWS será testado em máquinas com diferentes especificações (secção 6.1), o que confirmará o funcionamento da solução final de acordo com os requisitos mínimos. Esses requisitos, para um servidor OW, variam de acordo com o número

[51] URL: endereço usado para localizar um recurso numa rede de computadores.

de utilizadores em simultâneo que se pretende no mundo virtual. Um maior número de utilizadores será mais exigente em termos de hardware.

O critério de selecção da máquina SW foi o facto de ser uma das que estava disponível no CIETI-LABORIS, e sem uso naquela altura, sem atender a qualquer requisito de hardware. O SO instalado foi a distribuição de Linux Ubuntu 9.10. Chegou-se a equacionar a instalação de uma versão Windows recente, mas depois de analisar as instruções de instalação do servidor OW, optou-se por manter o Ubuntu, pois essas instruções são mais simples e claras para distribuições Linux, dado que a versão do OW escolhida para instalar requer componentes de software adicionais, que são teoricamente mais fáceis de configurar em sistemas Linux do que em Windows, para além de estarem melhor documentados para o primeiro SO.

O dispositivo MW corresponde ao multímetro Fluke 8845A, analisado na secção 3.2 e constitui o multímetro real (MR) da solução final.

Não está incluída na Figura 13 a UPS referida no capítulo 1, devido a este dispositivo não se relacionar directamente com a solução pretendida. Essa UPS está ligada à máquina SW e ao dispositivo MW. A sua finalidade é alimentar electricamente o hardware a ela ligado, prolongando o seu funcionamento em caso de falha de energia eléctrica. Caso se esgote a energia da UPS, o SO da máquina SW deverá estar preparado para ser encerrado automaticamente, para evitar encerramentos abruptos, que causam algumas vezes falhas de inicialização no arranque do SO, devido a ficheiros corrompidos.

Se a máquina SW por qualquer motivo, for desligada, o seu SO deverá implementar um mecanismo de arranque automático do servidor OW, logo após do arranque do SO, para facilitar o restabelecimento do funcionamento da solução proposta, sem necessidade de usar comandos específicos para o arranque do servidor OW. O MW só pode ser ligado/desligado localmente.

3.4 Arquitectura lógica

A arquitectura lógica da solução representa os componentes ou aplicações de software, que executam ou interagem, com os diferentes elementos da arquitectura física.

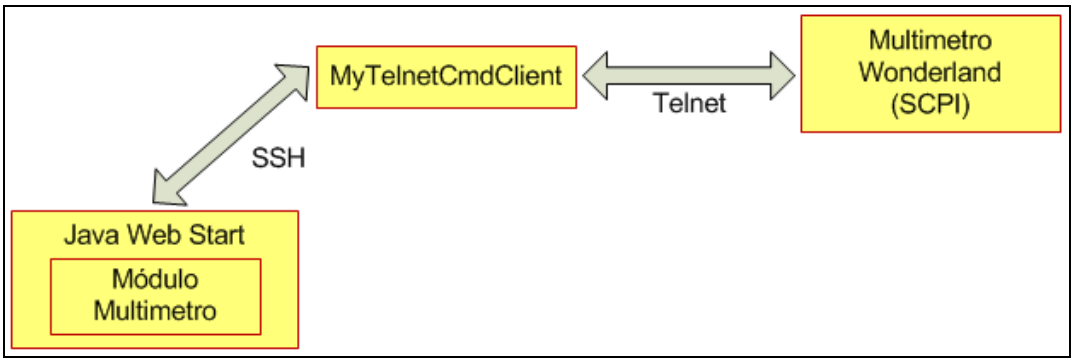


Figura 14 - Arquitectura lógica da solução

Cada componente da Figura 14 é executado num dos elementos da Figura 13, como por exemplo a aplicação JWS, que executa nos computadores dos utilizadores que acedem a um mundo virtual. Essa aplicação constitui a interface entre um utilizador e um mundo virtual OW. O componente Módulo Multimetro é a reprodução virtual do MR, também designada de multímetro virtual (MV). Um módulo do OW corresponde a um componente de software, a ser integrado num mundo virtual OW, que no caso do Módulo Multimetro, encontra-se instalado na máquina SW onde o servidor OW executa. O MV fica disponível aos utilizadores, com a integração do Módulo Multimetro no mundo virtual.

A aplicação JWS permite a usabilidade do mundo virtual, através de uma ligação TCP/IP com a máquina SW, onde o Módulo Multímetro é um componente que permite interacção com o utilizador num mundo virtual. Um módulo OW tem código que executa no PC do utilizador ou na máquina SW.

O Módulo Multimetro interage com a aplicação *MyTelnetCmdClient* através de uma ligação SSH, entre o PC do utilizador e a máquina SW. Essa aplicação executa na máquina SW. A sua base de funcionamento é um comando que executa na linha de comandos do SO, para efectuar medições remotas no dispositivo MW, de uma forma segura, usando o protocolo SSH. Existe código do Módulo Multimetro que executa do lado do utilizador (cliente) o comando da aplicação *MyTelnetCmdClient*. Basicamente, o módulo do MV permite ao utilizador executar um comando remotamente, na máquina SW. A funcionalidade de executar comandos via SSH na máquina SW, é devido a esta ter a executar um servidor SSH paralelamente ao servidor OW.

A aplicação *MyTelnetCmdClient* foi construída com o objectivo de evitar ligações *telnet* directas, entre um utilizador e o dispositivo MW, devido a esse protocolo ser pouco seguro (sem encriptação). Assim, essa ligação *telnet* é estabelecida apenas entre a máquina SW e o MW (ligação entre os componentes *MyTelnetCmdClient* e *Multimetro Wonderland* na Figura 14) pela aplicação referida. Através dessa ligação, a aplicação *MyTelnetCmdClient* obtém os valores de medição, executando comandos SCPI no dispositivo MW. Depois de obter um valor de medição, a aplicação retorna-o para o Módulo Multímetro, que por sua vez apresenta o valor no MV, visualizado por um utilizador ligado ao OW (mundo virtual), através da aplicação cliente JWS.

A utilização do MV deve ser controlada por um mecanismo de acessos concorrentes, que restrinja essa utilização ou controlo do dispositivo, a apenas um utilizador. Pretende-se com esse mecanismo, assegurar uma utilização ordenada do MV, para evitar que o trabalho de um utilizador seja perturbado pelos restantes. Um utilizador poderá solicitar o controlo do MV, através dos mecanismos de comunicação (*chat*) entre utilizadores disponibilizados pelo OW, os mais comuns são através de texto ou voz. Cabe depois ao utilizador que detém o controlo, cedê-lo ou não, ao utilizador que o solicitou.

O MV deve ser complementado por um cenário virtual, com uma aparência minimamente semelhante a um laboratório. Nesse cenário é importante que o OW inclua ferramentas que possibilitem a exposição de documentos, por exemplo no formato PDF[52], para não deixar o utilizador sem nenhuma informação auxiliar. A utilização do MV pode ser auxiliada, por exemplo, por um documento, como o manual do utilizador/programador do MR, assim o utilizador sabe qual o instrumento de medição que está a controlar remotamente. Também seria boa prática existir no mundo virtual um documento que expusesse a especificação de um circuito eléctrico, a sua ligação ao MR, que variáveis eléctricas medir, e conclusão dessas medições. Esse documento faria sentido se o MR fosse ligado a um circuito eléctrico, o que teoricamente, poderia auxiliar o estudo de algumas matérias relacionadas com energia eléctrica.

A solução proposta neste capítulo é posta em prática nos capítulos 4 e 5, que descrevem como tornar utilizável um mundo virtual OW e como desenvolver o componente Módulo Multimetro, respectivamente.

[52] PDF: extensão de ficheiro que representa documentos que contêm texto ou imagens de forma independente de uma aplicação informática.

4. Implementação da solução

Depois do planeamento global da solução final, é altura de segmentar a sua implementação por etapas. Cada etapa corresponde a um conjunto de tarefas semelhantes entre si. Depois de montar a infra-estrutura informática da secção 3.3, o trabalho passa por instalar, configurar e administrar o servidor do ambiente imersivo OW na máquina SW, para ser possível a um utilizador navegar num mundo virtual, ainda antes de existir o MV, para verificar como é a usabilidade do OW do lado do cliente. Com o servidor OW a funcionar e devidamente configurado, a etapa seguinte passa por construir um mundo virtual com uma aparência o mais semelhante possível a um laboratório. Paralelamente a esta tarefa, é construído/desenhado o MV e testada a sua usabilidade, antes do mesmo ser capaz de comunicar com o MR. Toda a construção do MV será abordada em pormenor no capítulo 5. A última etapa desta implementação foi a configuração de dois mecanismos adicionais. O primeiro relacionado com o arranque automático do servidor OW quando arranca o SO, e o segundo com a UPS ligada à máquina SW. Estes mecanismos serão abordados na secção 4.7.

4.1 Instalação/Configuração do Open Wonderland

Existem dois tipos de instalação de um servidor OW: por ficheiro binário [Binary Download, 2010] ou por *download* do código fonte [Open Wonderland v0.5: Download, Configure, Build and Run from the Source Code, 2010], ambos requerem a instalação da máquina virtual Java: o primeiro tipo de instalação consiste num ficheiro binário executável Java, executado por um comando de SO. É uma forma de instalação muito simples, com a desvantagem do servidor OW ficar limitado às suas funcionalidades incluídas de raiz, impossibilitando assim a eventual modificação das mesmas, ou a adição de outras. Neste tipo de instalação, a configuração do servidor OW é mais difícil. O segundo tipo de instalação requer *download* do código fonte, e posterior compilação, através dos componentes adicionais *Subversion* e *Apache Ant*, explicados mais à frente. Nesta instalação, é possível adicionar/modificar funcionalidades no servidor OW, graças ao acesso ao seu código fonte. A configuração pós-instalação do servidor sai facilitada, pelas instruções na página *Web* do OW. Estas vantagens do segundo tipo de instalação foram decisivas, para ser a metodologia seleccionada para instalar o servidor OW na máquina SW.

Essa instalação requer os componentes *Subversion* [Apache Subversion Features, 2010] e *Apache Ant* instalados no SO: o primeiro é um software de controlo de versões de código fonte, cujo objectivo é comparar diferentes versões e actualizar o código da forma conveniente, o segundo é um sistema de automatização de compilação de software Java, que funciona por base de ficheiros de compilação XML, à semelhança das *makefile*[53] em sistemas Linux/Unix.

[53] *Makefile* é uma metodologia de compilação de vários ficheiros de código na construção de uma aplicação.

Para instalar o servidor OW usa-se o Comando 2 (proveniente do componente *Subversion*):

```
svn checkout https://openwonderland.googlecode.com/svn/branches/0.5-preview4 wonderland
```

Comando 2

Este comando faz o *download* do código fonte, da versão do OW mais estável, seleccionada por questões de fiabilidade. Esse comando poderá também ser usado para actualizar o código fonte, que é actualizado regularmente e disponibilizado no URL especificado no Comando 2.

O funcionamento do servidor OW implica que o seu código fonte seja compilado, a partir do ficheiro *build.xml*, com instruções XML de compilação. Esse ficheiro encontra-se no directório raiz do código fonte, para o compilar deve-se executar o comando *ant* (proveniente do componente *Apache Ant*) no mesmo directório do ficheiro *build.xml* (um comando *ant* é sempre executado no mesmo directório desse ficheiro).

Antes de arrancar o funcionamento do servidor OW, é imperativo que seja configurado. A sua configuração é feita no ficheiro *my.run.properties* (incluído no directório do servidor), que contém instruções/propriedades necessárias ao seu arranque e funcionamento. As propriedades vitais ao funcionamento do servidor OW são:

- ***wonderland.webserver.host*:** define o IP da máquina onde o servidor OW vai funcionar.
- ***wonderland.webserver.port*:** é o porto de acesso ao servidor OW.
- ***wonderland.web.server.url*:** estabelece o URL de acesso público ao servidor *Web* OW.

Note-se a palavra *webserver* nas propriedades anteriores, que indica que toda a administração/gestão do servidor OW é feita numa página *Web*, a ser explicada na secção 4.2. É também a partir de uma página *Web* do servidor OW que os utilizadores obtêm a aplicação JWS, para depois acederem a um mundo virtual.

Depois da configuração, o servidor está pronto para arrancar, usando o comando *ant run-server*, que compila código fonte específico para o arranque do servidor.

Quando termina o arranque do servidor OW, surge no SO uma mensagem a indicar a execução do mesmo. Assim, a página *Web* inicial do OW fica disponível para ser acedida, a partir de um endereço URL, onde se especifica o endereço (IP ou nome da máquina SW) e o porto de acesso do servidor OW.

Na página inicial do servidor OW (Figura 15), encontram-se dois grandes botões: *Launch* e *Server Admin*, o primeiro lança aplicação cliente JWS para acesso ao ambiente 3D (será explicado com mais pormenor na secção 4.3), o segundo permite aceder à página de gestão dos componentes do servidor OW exibida na Figura 16.

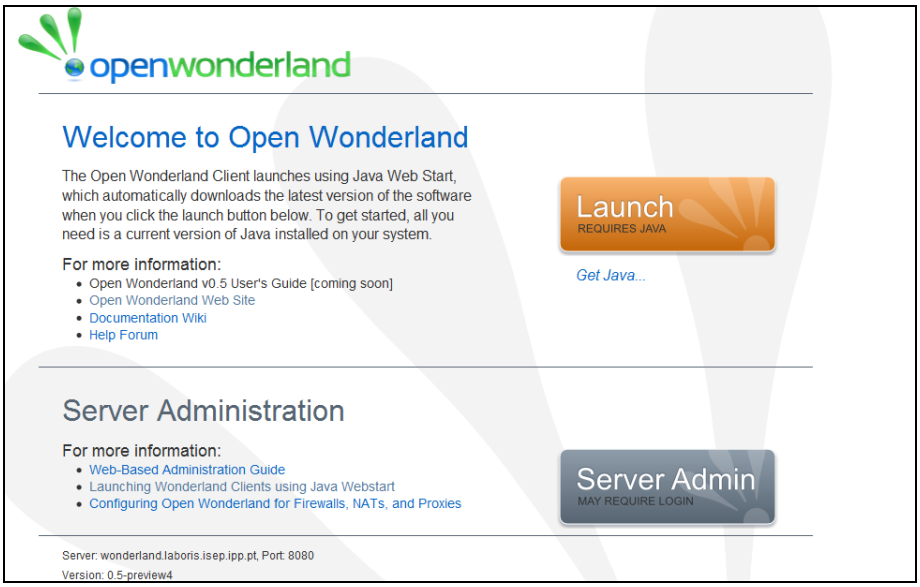


Figura 15 - Página inicial do servidor Open Wonderland

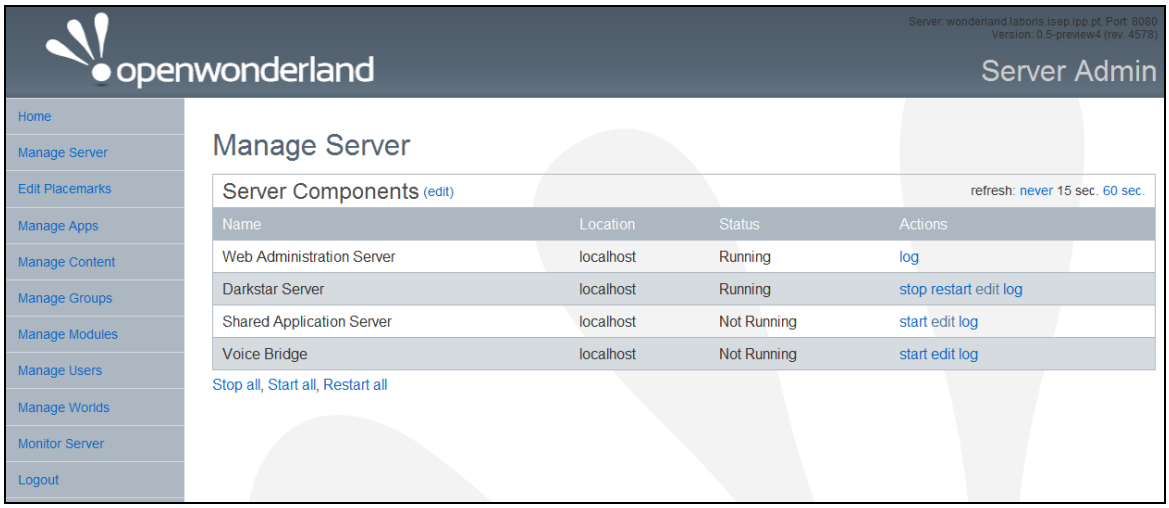


Figura 16 - Página de gestão do servidor Open Wonderland

De acordo com a Figura 16, existem quatro componentes na página de gestão do servidor:

- ***Web Administration Server*:** conforme o nome indica, é responsável por disponibilizar a interface *Web* de administração/gestão do servidor. Arranca sempre com o comando de arranque *ant run-server*, estando sempre a funcionar.
- ***Darkstar Server*:** este componente é o que possibilita o ambiente 3D/mundo virtual do OW, tem que estar activado para os utilizadores lhe acederem.
- ***Shared Application Server*:** permite que programas executáveis na máquina, onde está alojado o servidor OW, sejam executados ou utilizados a partir do mundo virtual.
- ***Voice Bridge*:** fornece funcionalidades de comunicação por voz entre utilizadores do mundo virtual.

Destes componentes, só o *Web Administration Server* e o *Darkstar Server* são vitais ao funcionamento, tanto do servidor OW como de um mundo virtual. Todos os componentes, exceptuando o *Web Administration Server*, podem ser desactivados ou activados. A única forma de desactivar o *Web Administration Server* é terminar o processo arrancado com *ant run-server* no SO.

Nos casos em que o servidor OW se encontra atrás de uma *firewall* numa rede privada, existe a possibilidade de ser configurado correctamente usando a propriedade [Kaplan, J., 2010] *darkstar.host.public* no componente *Darkstar Server*, para um mundo virtual ficar disponível na Internet. O conteúdo da propriedade *darkstar.host.public* será o nome da máquina SW na Internet. A página inicial do servidor OW e o mundo virtual só ficam definitivamente disponíveis na Internet, abrindo na *firewall* da rede do CIETI-LABORIS, os portos de acesso correspondentes ao servidor *Web* e ao *Darkstar Server*.

Todas as funcionalidades do servidor OW são configuradas em páginas *Web* semelhantes à Figura 16. A cada funcionalidade corresponde um *link* do menu esquerdo, sendo a funcionalidade gerida/configurada à direita desse menu.

4.1.1 Autenticação de utilizadores

Uma característica muito importante não incluída por omissão no OW, é a autenticação de utilizadores, ou seja, originalmente a página *Web* de administração servidor está aberta a todos os utilizadores, pormenor indesejável na manutenção de qualquer infra-estrutura informática. A funcionalidade para autenticação de utilizadores num servidor OW, baseia-se num componente/módulo obtido da mesma forma que o *download* do código fonte do OW, por um comando semelhante ao Comando 2, mas com um URL diferente. Para esse componente ser adicionado ao servidor OW, é compilado com o comando *ant*. O componente é depois integrado no servidor OW, alterando propriedades em ficheiros de configuração que fazem parte do mesmo. Assim, é necessário interromper o servidor, para recompilação do código fonte com o arranque do servidor, usando o comando *ant run-server*.

Clicando no botão *Server Admin* na página inicial do servidor OW, surge a página de autenticação da Figura 17, que irá limitar o acesso à página de gestão de componentes do servidor.

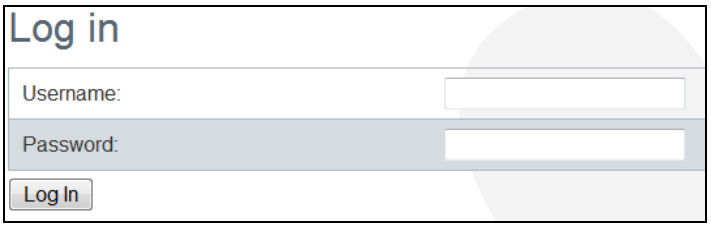


Figura 17 - Página de autenticação da página *Web* de administração do Open Wonderland

A autenticação é baseada em *Username* (nome de utilizador) e *Password* (palavra-chave). Quando a página de gestão do servidor OW inclui autenticação, o funcionamento dos seus componentes requer autenticação, tal como um utilizador. A autenticação dos componentes deve ser automatizada pela criação de um ficheiro de palavra-chave no SO, onde cada componente autentica-se, usando uma propriedade de configuração com o caminho (*path*[54]) do ficheiro no SO (configuração efectuada no ficheiro *my.run.properties* para o componente *Web Administration Server*, e na página de gestão de componentes para os restantes). Esse ficheiro denomina-se *wonderland.password*, deve ser oculto no SO e acedido/gerido apenas pelo administrador do SO, por questões de segurança.

Estando os componentes do servidor correctamente configurados e a sua página de gestão protegida por autenticação, pode-se então fazer a administração/gestão corrente do servidor, estando disponível um mundo virtual, a ser visualizado na aplicação cliente JWS.

4.2 Administração do servidor

Uma boa administração/gestão do servidor OW é fundamental [Project Wonderland v0.5: Web-based Server Administration, 2010], para que um mundo virtual disponibilizado aos utilizadores funcione bem. Essa administração consiste essencialmente na gestão dos componentes do servidor, na gestão de utilizadores e na gestão de mundos virtuais.

4.2.1 Gestão dos componentes do servidor

Depois de uma autenticação bem sucedida no servidor OW, a próxima página a ser exibida, é a de gestão dos componentes do servidor conforme mostra a Figura 18, igual à Figura 16, mas colocada nesta secção para uma compreensão mais clara.

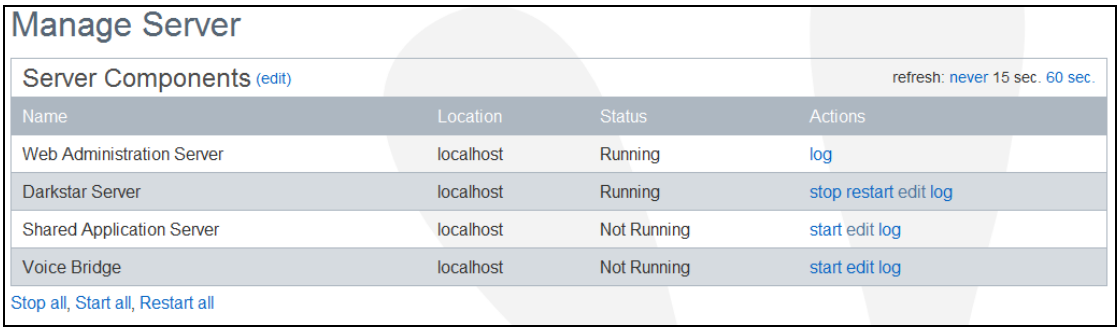


Figura 18 - Página de gestão dos componentes do servidor Open Wonderland

De acordo com a figura anterior tem-se uma tabela com os quatro componentes do servidor explicados na secção anterior, na primeira coluna (*Name*) está o nome do componente, na segunda (*Location*) o nome da máquina onde o componente está armazenado e onde executa, na terceira (*Status*) o seu

[54] *Path*: termo usado para designar a localização de um ficheiro ou directório num SO.

estado de execução e na quarta (*Actions*) estão *links* que são usados para executar acções sobre os componentes. Neste caso, todos os componentes estão armazenados e são executados na máquina *localhost*, ou seja, a máquina local. Em relação ao estado de execução verifica-se (Figura 18) que os componentes *Web Administration Server* e *Darkstar Server* estão a executar, devido a terem o estado em *Running* ou activados, enquanto que os componentes *Shared Application Server* e *Voice Bridge* estão no estado *Not Running* ou desactivados. Os estados de execução para os componentes do servidor OW podem ser:

- ***Running*:** o componente está a executar ou activado.
- ***Not Running*:** componente desactivado.
- ***Starting Up*:** é um estado de curta duração que significa a activação do componente.
- **SHUTTING_DOWN:** significa que o componente está a ser parado ou desactivado. Também é um estado de curta duração.

Antes da acção de arranque do servidor OW, é possível definir no ficheiro de configuração *my.run.properties* a propriedade *wonderland.runner.autostart*, que assume os valores *true* ou *false*: *true* quando pretende-se activar automaticamente todos os componentes no arranque, ou *false* para não activa-los automaticamente. No ficheiro de configuração a mesma propriedade vem com o valor *true* por omissão, isso repercute-se no tempo de arranque do servidor OW, sendo ligeiramente mais demorado devido à activação dos componentes durante o arranque, o que não acontece com o valor a *false*, diminuindo assim o tempo de arranque do servidor.

A parte mais importante da página de gestão dos componentes do servidor são os *links* da coluna *Actions*, pois permitem alterar o estado de execução (activar/desactivar componentes), adicionar/modificar/remover propriedades de configuração e visualizar *logs* (ou eventos registados).

4.2.2 Gestão de utilizadores

O OW foi concebido para utilização de vários utilizadores, mas para isso tem de haver funcionalidades que permitam a sua gestão de acordo com a página *Web* da Figura 19.

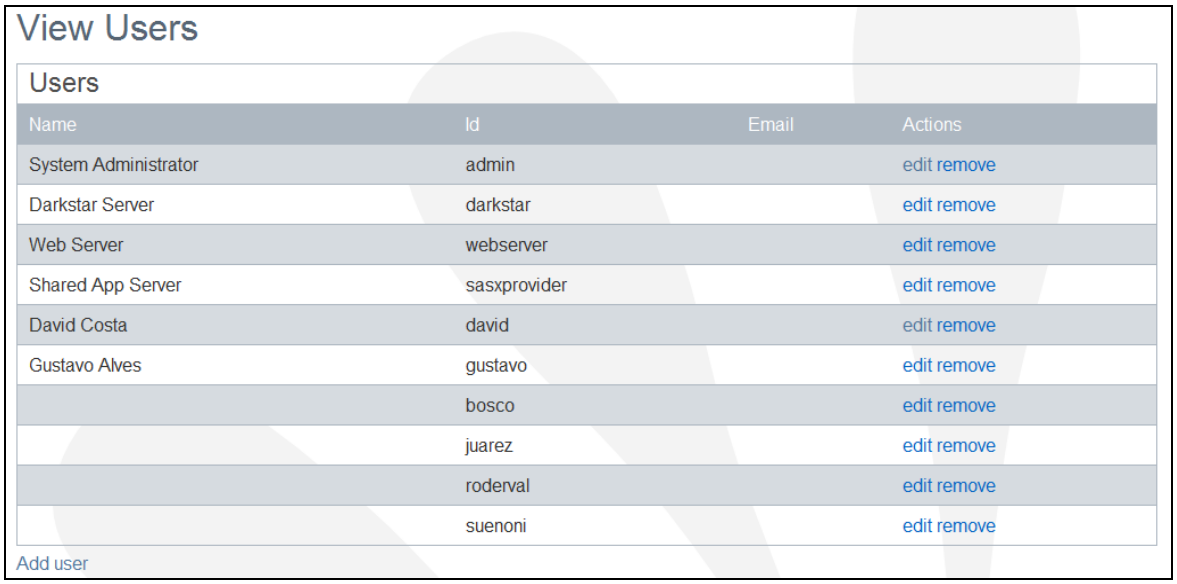


Figura 19 - Página de gestão de utilizadores do Open Wonderland

Nessa página, na tabela denominada *Users*, cada linha corresponde a um utilizador. Ao campo da coluna *Name* corresponde o nome do utilizador, na coluna *id* é guardada a sua identificação (*Username*), que deve ser única, o correio electrónico corresponde à coluna *Email*, e na coluna *Actions* encontram-se os *links edit* e *remove* para respectivamente editar/modificar e apagar um utilizador. A inserção ou modificação dos dados de um utilizador são as operações mais importantes, porque definem as suas credenciais de autenticação, que são *Username* e *Password*.

Existem quatro utilizadores incluídos por omissão no servidor OW, não devendo ser removidos, que são: o *System Administrator*, como o nome indica, é o administrador do servidor, o *Darkstrar Server* é o utilizador representante do componente com o mesmo nome, o utilizador *Web Sever* corresponde ao componente *Web Administration Server* e o *Shared App Server* é o utilizador correspondente ao componente *Shared Application Server*.

O servidor OW fornece a funcionalidade de agrupar utilizadores em diferentes grupos, é um pormenor meramente informativo, porque não existem funcionalidades de atribuição de privilégios de utilização, em função do grupo do utilizador, à excepção do grupo de utilizadores *admin*. Qualquer utilizador que faça parte deste grupo pode administrar o servidor OW, inicialmente fazem parte desse grupo os utilizadores: *System Administrator*, *Darkstar Server*, *Web Server* e *Shared App Server*, sendo possível adicionar outros utilizadores ao grupo desde que efectuado por um utilizador pertencente ao mesmo. O grupo *admin* vem incluído por omissão no servidor OW.

4.2.3 Gestão de mundos virtuais

Um utilizador registado no servidor OW pode aceder a um ambiente 3D ou mundo virtual, para isso o servidor OW deve disponibilizar vários cenários 3D. Na Figura 20 está a página de gestão de mundos virtuais do servidor OW.

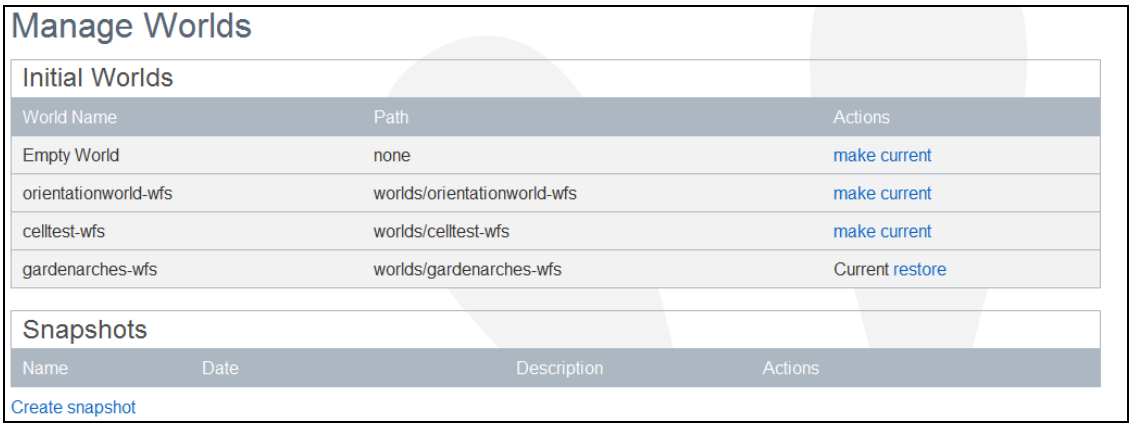


Figura 20 - Página de gestão de mundos virtuais

Cada linha da tabela *Initial Worlds* representa um cenário 3D ou mundo virtual diferente, a coluna *World Name* é o nome do mundo, a coluna *Path* corresponde ao caminho onde o mundo virtual está armazenado no SO em relação ao directório *.wonderland-server/0.5-preview4/wfs*, e finalmente, na coluna *Actions* estão os *links make current* ou *restore*. O primeiro *link* serve para estabelecer qual o mundo virtual que vai ser disponibilizado aos utilizadores na aplicação JWS, sendo que só pode estar um mundo virtual activo de cada vez, sendo impossível a um utilizador escolher um mundo virtual diferente daquele que lhe é apresentado. Uma mudança de mundo virtual implica sempre o acesso à página de gestão de mundos virtuais. A funcionalidade do *link restore* é restaurar um mundo virtual ao seu estado inicial, muitas vezes útil, conforme se perceberá na secção 4.6. As acções de *make current* ou *restore* implicam que o componente *Darkstar Server* seja reiniciado.

Normalmente, os mundos virtuais mais usados são o *Empty World* e *gardenarches-wfs* por serem os menos exigentes em recursos de hardware. O *Empty World* costuma servir de base à construção de outros mundos virtuais. Para exemplificar, mostram-se os mundos *Empty World* e *gardenarches-wfs* nas Figuras 21 e 22, respectivamente.

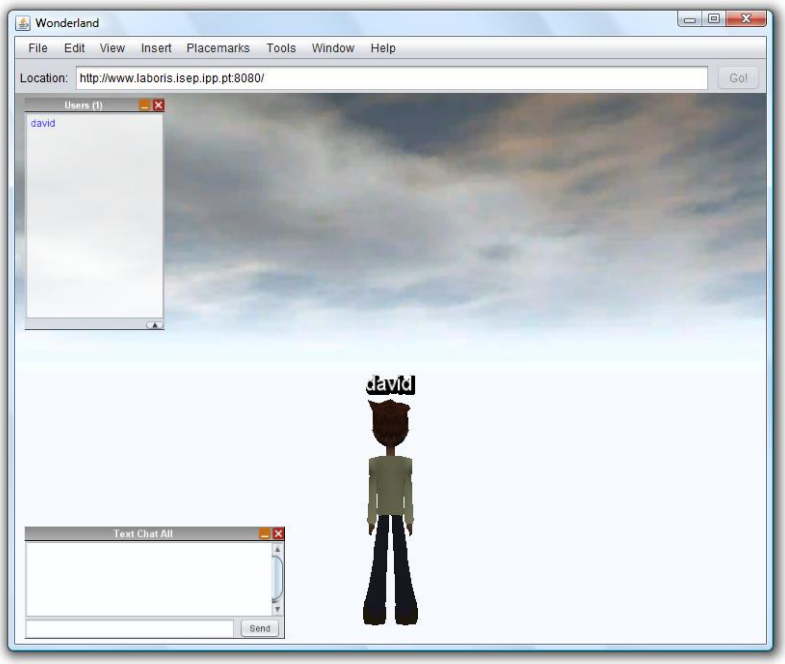


Figura 21 - Mundo virtual *Empty World*

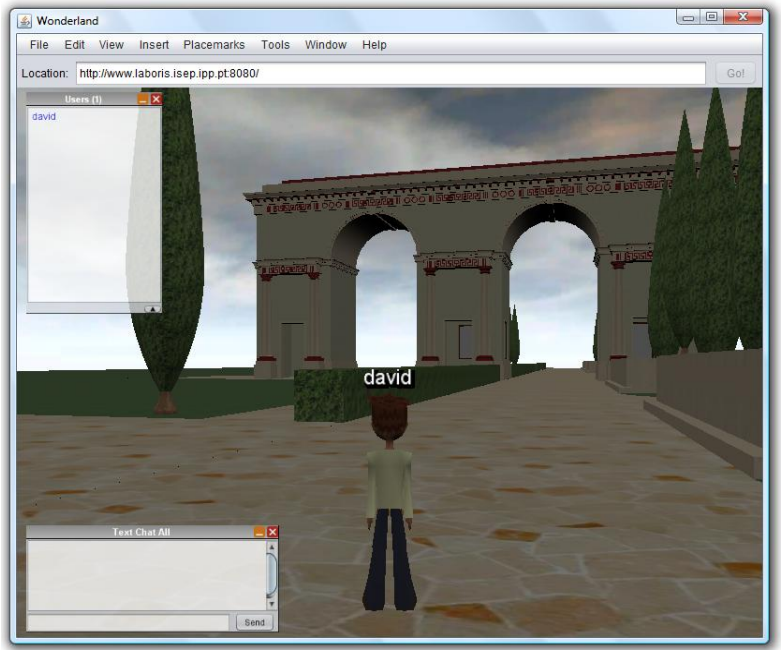


Figura 22 - Mundo virtual *gardenarches-wfs*

Todas as operações descritas nesta secção são exclusivas do utilizador *System Administrator*, também denominado de administrador do servidor OW, ou a qualquer outro utilizador pertencente ao grupo *admin*. Quando um destes utilizadores referidos se autentica com sucesso, é imediatamente direccionado para a página *Web* da Figura 16. A partir daí pode administrar tudo o que o servidor OW disponibiliza para o efeito. Os restantes utilizadores ao tentarem aceder à página *Web* referida, são direccionados para uma página onde podem apenas alterar a sua palavra-chave de acesso.

4.3 Acesso ao Open Wonderland

A finalidade desta secção é explicar como um utilizador pode aceder ao ambiente 3D ou mundo virtual do OW, através da aplicação cliente JWS, e tirar partido de algumas das suas funcionalidades.

Um utilizador obtém a aplicação JWS, ao carregar no botão *Launch* (Figura 15) da página inicial de um servidor OW, que faz o *download* para o utilizador, do ficheiro *Wonderland.jnlp*, que é o executável da aplicação JWS. Esta aplicação requer a máquina virtual Java instalada no SO onde executa.

O acesso a um mundo virtual de um utilizador registado no servidor OW, requer as suas credenciais definidas na página de gestão de utilizadores. Essas credenciais devem ser introduzidas nos campos *Username* e *Password* do ecrã da Figura 23.

Figura 23 - Ecrã de autenticação na aplicação cliente JWS

Tomando como exemplo a Figura 22, o mundo virtual definido no servidor OW é o *gardenarches-wfs*, e o *Username* do utilizador que acedeu é *david*, depois de autenticado.

O boneco das Figuras 21 e 22 denomina-se *avatar*, corresponde à representação de um utilizador no mundo virtual, que neste caso, está presente apenas o utilizador *david*, conforme a indicação por cima do *avatar*. O utilizador controla o movimento do *avatar* através das teclas direccionais do teclado. Num mundo virtual, os utilizadores ou *avatares*, podem comunicar entre si por texto, áudio ou vídeo. Para esta tese foi apenas usada a comunicação por texto.

O OW permite ao utilizador seleccionar diferentes *avatares*, bem como construir/modificar o seu próprio *avatar* [Yankelovich, N., 2010]. As diferentes configurações do *avatar* podem variar desde sexo, cor da pele, tipo de penteado, cor do cabelo e tipo/cor do vestuário. A Figura 24 mostra o resultado da configuração de um *avatar*.

Figura 24 - *Avatar* configurado pelo utilizador

A configuração do *avatar* depende da memória da placa gráfica do PC, onde executa a aplicação JWS, pois uma placa gráfica com uma memória abaixo dos requisitos mínimos impossibilitará essa configuração. Este pormenor será abordado no capítulo 6, nos testes efectuados à solução final.

4.4 Arquitectura modular

Entende-se por arquitectura modular a existência e a forma como os módulos constituintes de uma aplicação ou sistema informático se organizam e interagem entre si. Um módulo é um componente de software que, em conjunto com outros, forma uma só aplicação. Todo o OW é formado por módulos, sendo que uns constituem as suas funcionalidades nucleares, enquanto outros servem para funcionalidades adicionais [Slott, J., 2010 a]. A utilização mais frequente de módulos no OW é para expandir as suas funcionalidades, o próprio trabalho efectuado nesta tese confirma isso, conforme se abordará no capítulo 5. A reprodução virtual do multímetro (instrumento de medição abordado na secção 3.2) foi concebida com base num módulo OW, que corresponde ao componente Módulo Multímetro da arquitectura lógica da solução (Figura 14 na secção 3.4).

No caso do OW, um módulo é construído a partir de código Java que, depois de compilado, cria um ficheiro do formato JAR[55]. São esses ficheiros que constituem os módulos que se adicionam ao OW. Para adicionar módulos, deve-se navegar para a sua página de gestão (Figura 25), na página de administração do servidor OW.

[55] JAR: extensão para ficheiros executáveis num SO, de aplicações construídas na linguagem de programação Java.

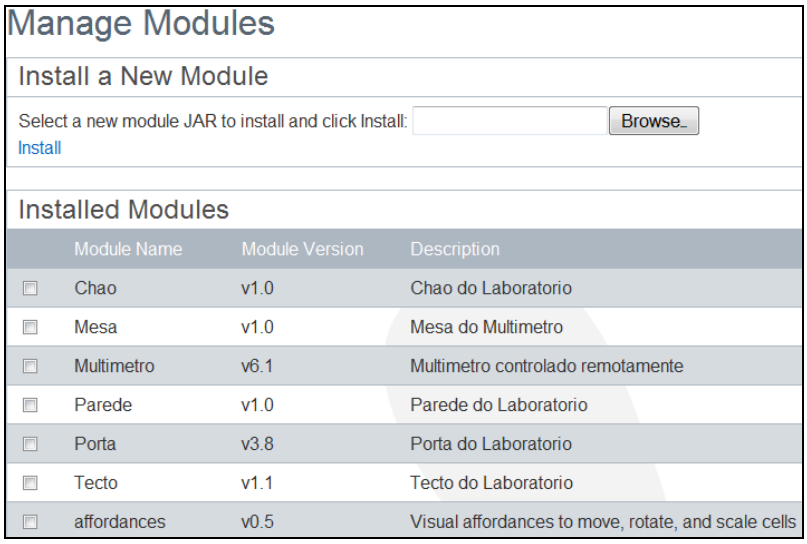


Figura 25 - Página de gestão de módulos

Na página de gestão de módulos é possível inserir, remover e visualizar módulos para o OW. Todos os módulos instalados no servidor OW estão indicados na tabela *Installed Modules*, onde cada linha corresponde a um módulo. Em relação às colunas da tabela, na *Module Name* encontra-se o nome do módulo, a sua versão está na coluna *Module Version*, e na coluna *Description* está a sua descrição. Os atributos destas colunas são definidos num ficheiro de configuração do módulo denominado *my.module.properties*. As instruções de configuração ou propriedades que fazem parte do ficheiro são as seguintes:

- ***module.name*:** é o nome do módulo, indicado na coluna *Module Name* (Figura 25). Instrução obrigatória, recomendando-se que seja única.
- ***module.description*:** o valor desta propriedade vai surgir na coluna *Module Description*. Instrução opcional.
- ***wonderland.dir*:** num módulo OW é sempre obrigatório que o seu ficheiro de configuração inclua o caminho onde se encontra o código fonte do OW, onde alguns dos seus elementos são usados na compilação do módulo.
- ***modulo.plugin.src*:** esta instrução embora sendo opcional, recomenda-se que seja sempre colocada, pois serve para indicar ao OW qual a estrutura de armazenamento do código fonte do módulo. Um módulo sem esta instrução pode não ser correctamente adicionado num mundo virtual.

No ficheiro de configuração não está definida a versão do módulo, essa definição encontra-se no seu ficheiro de compilação *build.xml*. A indicação da versão pode ser importante, para rever código anterior na adição/modificação de funcionalidades.

As propriedades de configuração do módulo são usadas na sua compilação, feita à base de um ficheiro de definições de compilação *build.xml*, à semelhança do que acontece na compilação do código fonte

do OW, abordado na secção 4.1. Um módulo é também compilado, executando o comando *ant* no mesmo directório do ficheiro *build.xml*.

Os módulos são a característica mais importante para expandir as funcionalidades do OW, pois são construídos principalmente para serem adicionados a um mundo virtual, como se verificará na próxima secção.

4.5 Construção de mundos virtuais

Foi abordado nas secções 4.2 e 4.3 como se gere e acede a um ambiente ou mundo virtual. A sua selecção é feita por um utilizador que seja administrador do OW na página de gestão de mundos virtuais (Figura 20 da subsecção 4.2.3). Qualquer utilizador adicionado ao servidor OW pode navegar com o seu *avatar* no mundo virtual, visualizado na aplicação cliente JWS, abordada na secção 4.3. Mas os mundos virtuais no OW não se resumem só aos disponíveis de raiz na página de gestão de mundos virtuais. É possível, no OW, um utilizador construir ou personalizar um mundo virtual com base em mundos existentes. Para isso, basta adicionar conteúdo ou objectos gráficos. Na Figura 26 é exemplificado um mundo virtual construído por um utilizador.

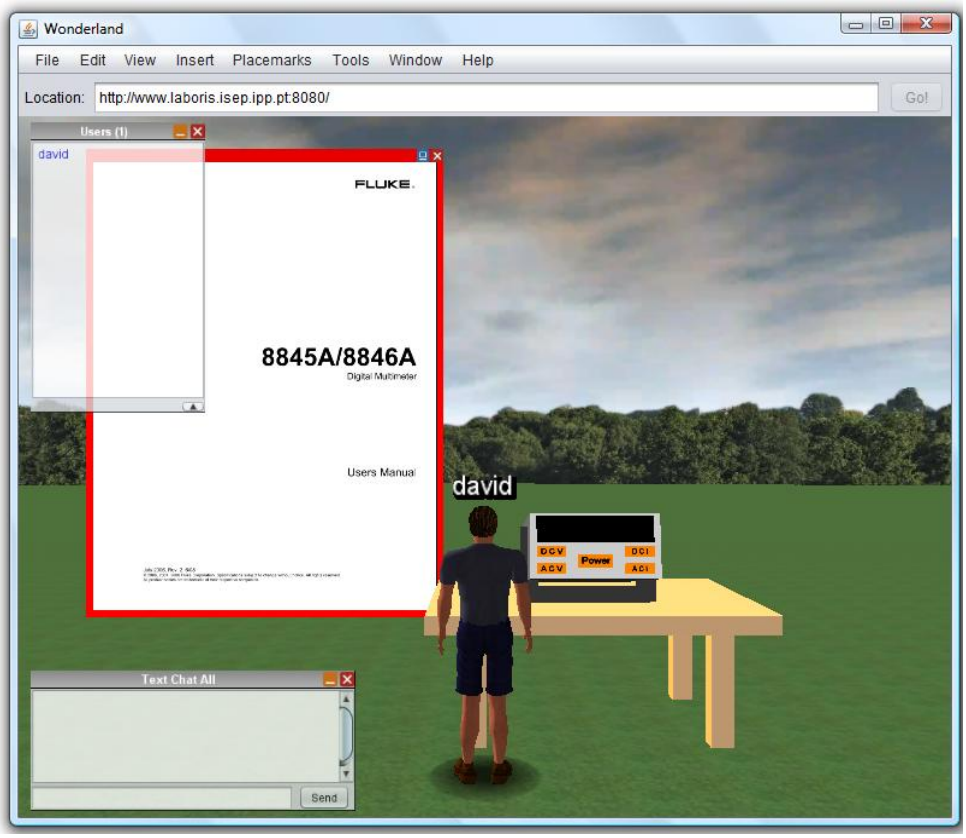


Figura 26 - Mundo virtual personalizado pelo utilizador

Esse mundo virtual representa o disponibilizado aos utilizadores, para efeitos desta tese, pois é o que melhor conjuga funcionalidade com desempenho. Foi construído com base no mundo virtual *Empty*

World da Figura 21 na secção 4.2, que é muito útil como base de construção de mundos virtuais, pois como se apresenta inicialmente vazio, dá mais liberdade para personalização sem comprometer a estética.

Os objectos 3D com os quais o cenário da Figura 26 foi construído, são baseados em ficheiros de modelos 3D no formato KMZ[56] (formato usado no Google Earth) e em módulos OW, devidamente compilados. As árvores no horizonte e o chão relvado são modelos 3D KMZ, adicionados com a simples acção de arrastar e largar (*drag & drop*), com o rato, o ficheiro para dentro da área envolvente da janela da aplicação JWS. Esta operação de adição de modelos 3D é sempre efectuada do lado do cliente ou utilizador. A mesa e o dispositivo por cima desta são módulos criados de raiz, adicionados no servidor com recurso à página de gestão de módulos. Esses módulos têm que estar instalados no servidor OW (Figura 25). O dispositivo por cima da mesa é proveniente do módulo MV. O quadro exposto do lado esquerdo da mesa e do multímetro é um simples ficheiro PDF, também adicionado pela acção de arrastar e largar à semelhança dos modelos 3D, mas neste caso, essa funcionalidade só é possível devido ao OW já incluir de raiz um módulo que o permite. O conteúdo do ficheiro PDF, presente no quadro, é o manual do utilizador do MR explicado na secção 3.2. Serve para dar uma ideia ao utilizador do dispositivo real correspondente ao MV, e explica o funcionamento desse dispositivo.

A simples adição de objectos 3D não é suficiente na construção de um mundo virtual, pois o mesmo deve ter os objectos 3D devidamente posicionados e dimensionados. O posicionamento e dimensionamento aplicado a objectos 3D no OW, faz-se clicando com o botão direito do rato por cima do objecto que se pretende modificar. O exemplo da Figura 27 mostra o menu para edição/modificação das propriedades do MV.

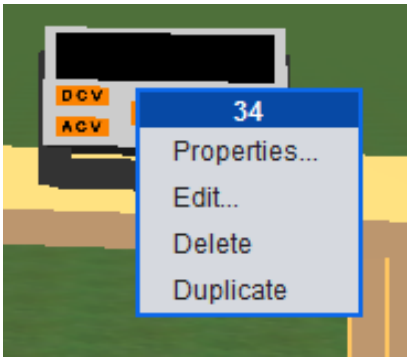


Figura 27 - Menu de edição de propriedades de objectos 3D

Para posicionar e dimensionar objectos 3D, clica-se na opção *Edit*, no menu da figura anterior, que possibilita uma variedade de opções de edição de um objecto 3D, como se verifica na Figura 28.

[56] KMZ**:** extensão de ficheiros que representam mapas, que podem ser de duas ou três dimensões.

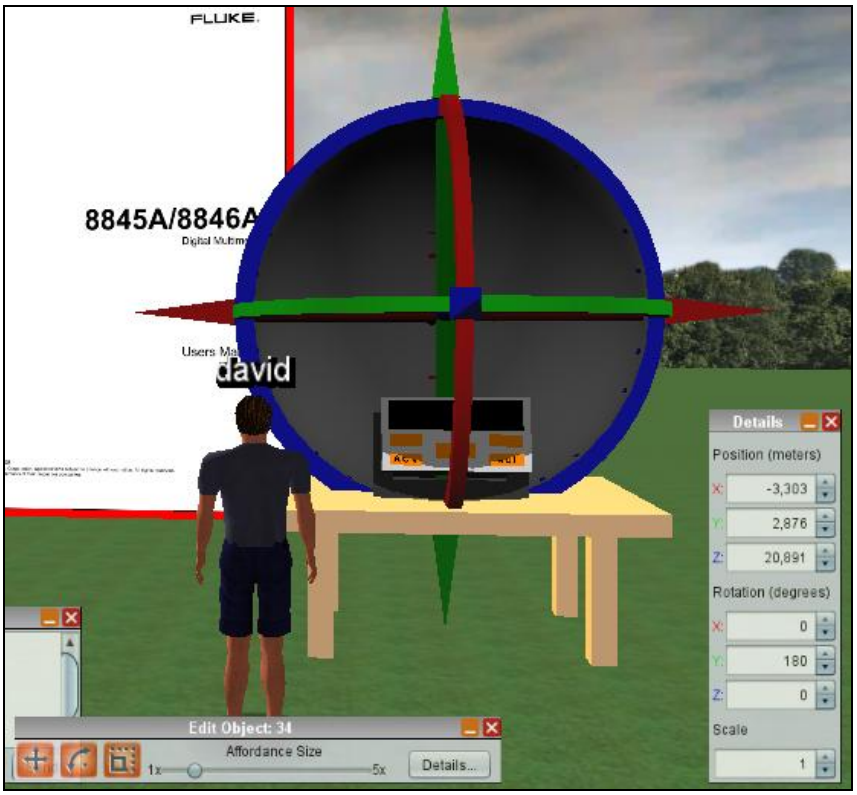
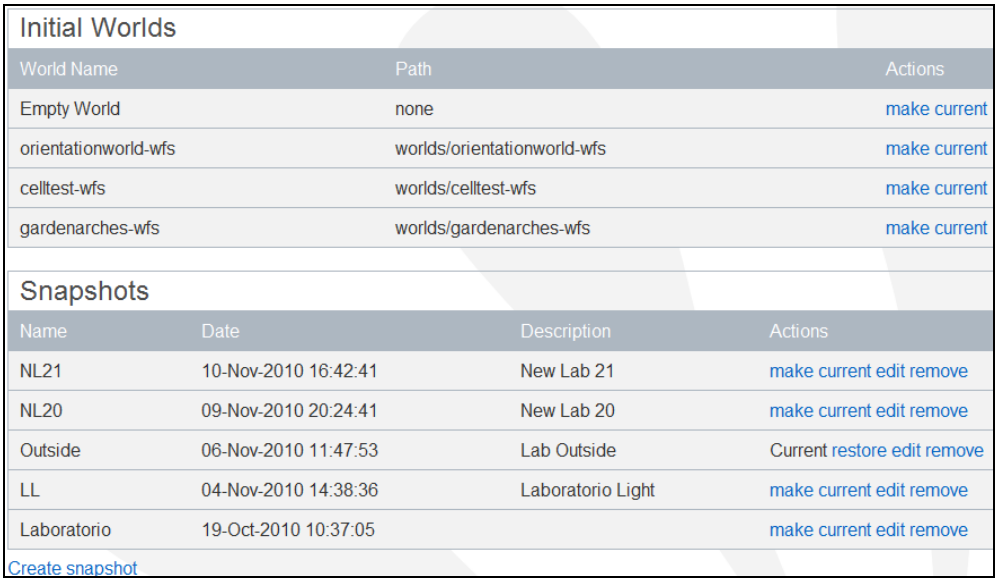


Figura 28 - Edição de um objecto 3D no Open Wonderland

O objecto 3D que está a ser editado na Figura 28 é o MV. O posicionamento de objectos 3D no OW, baseia-se no grupo de eixos e círculos de rotação, para possibilitar que se mova ou rode o objecto directamente a partir do rato nos eixos X, Y ou Z (largura, altura ou profundidade respectivamente). As dimensões de um objecto baseiam-se na sua escala, ou seja, para dimensionar objectos 3D no OW, ajusta-se a sua escala, em função da dimensão pretendida, já que o tamanho do objecto é tratado como um todo, sendo impossível alterar as suas dimensões (largura, altura, profundidade) separadamente. Os menus da Figura 28 permitem posicionar/dimensionar o objecto, usando apenas o rato, ou a partir da definição de valores numéricos para translação/rotação nos eixos XYZ, e para a escala.

4.5.1 Mecanismo de *snapshots*

O OW não memoriza por omissão um mundo virtual criado, assim, as configurações do mesmo podem ser perdidas caso o componente do servidor *Darkstar Server* seja reinicializado. Para evitar essa perda, o utilizador *System adminstrator* ou qualquer outro pertencente ao grupo *admin*, pode guardar um mundo virtual criado, usando o mecanismo de *snapshots*. Este mecanismo consiste em tirar uma "fotografia" ao mundo virtual, para depois este ficar disponível para ser usado, tal como os mundos virtuais existentes de raiz no OW. A criação de *snapshots* só pode ser feita com o componente *Darkstar Server* desactivado. As *snapshots* são criadas ou geridas na página de gestão de mundos virtuais (Figura 20). A Figura 29 ilustra essa página, mas com *snapshots* criadas.



Initial Worlds

World Name	Path	Actions
Empty World	none	make current
orientationworld-wfs	worlds/orientationworld-wfs	make current
celltest-wfs	worlds/celltest-wfs	make current
gardenarches-wfs	worlds/gardenarches-wfs	make current

Snapshots

Name	Date	Description	Actions
NL21	10-Nov-2010 16:42:41	New Lab 21	make current edit remove
NL20	09-Nov-2010 20:24:41	New Lab 20	make current edit remove
Outside	06-Nov-2010 11:47:53	Lab Outside	Current restore edit remove
LL	04-Nov-2010 14:38:36	Laboratorio Light	make current edit remove
Laboratorio	19-Oct-2010 10:37:05		make current edit remove

Create snapshot

Figura 29 - Criação de *snapshots* no Open Wonderland

Para criar uma *snapshot*, clica-se no *link Create snapshot*, por baixo da tabela *Snapshots* da Figura 29, cada *snapshot* representa uma linha na tabela referida. Em relação às suas colunas, a primeira denominada *Name*, representa o nome da *snapshot*, a data e hora da *snapshot* correspondem à coluna *Date*, a coluna *Description* corresponde à descrição da *snapshot*, e na coluna *Actions* encontram-se *links* de manipulação de *snapshots*, que se assemelham aos *links* da coluna *Actions* na tabela *Initial Worlds* das Figuras 29 e 20.

A gestão e funcionamento das *snapshots* provêm da gestão dos mundos virtuais, sendo ambas semelhantes. Para estabelecer a *snapshot* que vai estar disponível aos utilizadores na aplicação JWS, clica-se no *link make current*. O estado inicial da *snapshot* utilizada é reposto ao clicar no *link restore*. No caso da Figura 29, é a *snapshot* com o nome *Outside* que define o ambiente 3D ou mundo virtual disponibilizado aos utilizadores. As acções dos *links make current* e *restore* só funcionam com o componente *Darkstar Server* desactivado. As *snapshots* mais usadas no servidor OW desta tese são a *Outside* (Figura 26) e a *NL21* (Figura 30).

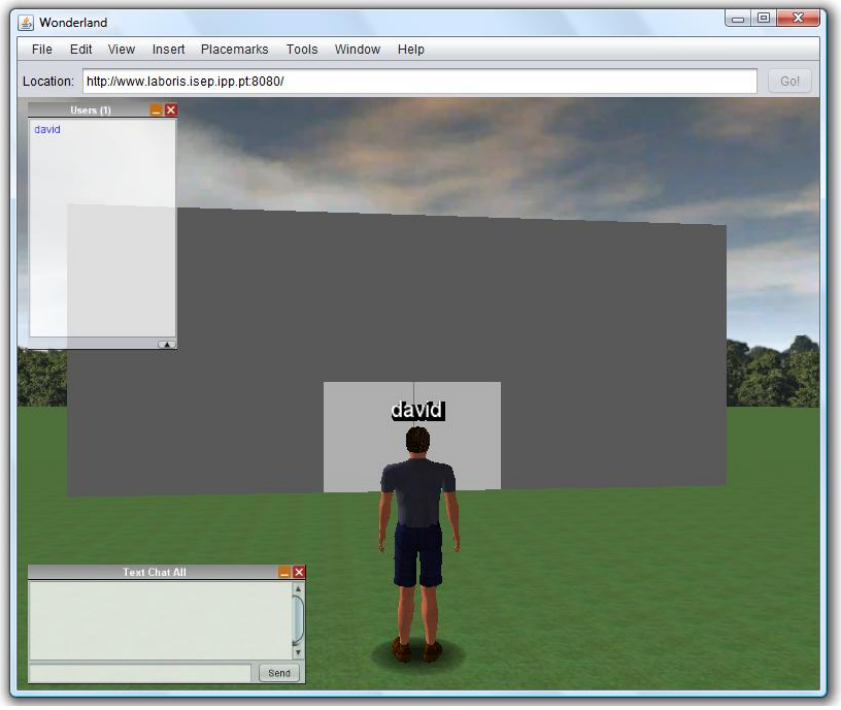


Figura 30 - *Snapshot NL21*

A construção do cenário 3D da Figura 30 foi baseada no mundo virtual da Figura 26. Apenas acrescenta o edifício situado à frente do *avatar*. Na construção de mundos virtuais deve-se ter em atenção o binómio desempenho/estética, pois beneficiar uma característica poderá penalizar a outra, dado que um mundo virtual com muitos objectos 3D ou com um nível de detalhe elevado irá penalizar o desempenho e exigir mais do hardware das máquinas. No entanto, um mundo virtual com essas características será esteticamente mais apelativo. Para esta tese estava previsto usar, por omissão, o mundo virtual da *snapshot NL21*, por ter uma estética minimamente aceitável para um laboratório virtual, mas foi preterido em favor do mundo virtual da Figura 26 (*snapshot Outside*), por este último ser computacionalmente mais leve, principalmente nos tempos de carregamento e menos exigente no hardware requerido, podendo assim abranger um maior número de utilizadores.

4.6 Segurança

Na construção de mundos virtuais é possível a um utilizador mudar a posição ou remover um objecto 3D adicionado por outro utilizador, nestas condições pode ser problemático manter a integridade ou segurança de um mundo virtual. Quando esta integridade é violada, é sempre possível ao utilizador que seja administrador do servidor OW, repor o estado inicial de um mundo virtual. Para evitar ao máximo que esse procedimento seja necessário, é possível aplicar permissões aos objectos 3D, à semelhança do que acontece nos ficheiros e directórios dos sistemas operativos mais recentes.

A aplicação de permissões a um objecto 3D faz-se recorrendo ao menu de edição de propriedades de objectos 3D (Figura 27). O mecanismo de permissões incide sobre utilizadores ou grupos de

utilizadores, ou seja, podem-se aplicar diferentes permissões de acordo com um utilizador, ou grupo de utilizadores. As permissões disponíveis num mundo virtual OW são:

- ***Change Capabilities*:** permissão para adicionar ou remover capacidades adicionais ao objecto 3D. Essas capacidades podem ser, por exemplo, *links* para a Internet ou marcação da localização do objecto 3D no mundo virtual.
- ***Add/Remove Object Children*:** o nome deste tipo de permissão pode induzir em erro por incluir a palavra *Add* (adicionar em português), mas apenas define a permissão para remover objectos 3D.
- ***Move*:** permissão para movimentar o objecto, ou seja, posicionar/dimensionar.
- ***View*:** este tipo de permissão corresponde à visualização de objectos 3D.

Estas permissões estão limitadas às seguintes opções:

- ***Unspecified*:** permissão não especificada, mas quando este valor é seleccionado equivale a permissão autorizada.
- ***Granted*:** permissão autorizada.
- ***Denied*:** permissão negada.

Partindo do princípio que ainda não foram aplicadas permissões a um objecto 3D, qualquer utilizador o pode fazer. O utilizador que aplica as permissões pela primeira vez, deve estabelecer a propriedade/dono (*owner*) do objecto. Pode-se estabelecer como dono a si próprio, outro utilizador, ou grupo de utilizadores. Um objecto 3D pode pertencer a mais do que um utilizador ou grupo. Quem tiver a propriedade do objecto, fica responsável por gerir as permissões do mesmo. Qualquer alteração posterior é negada a utilizadores ou grupos que não sejam donos do objecto 3D.

A funcionalidade de permissões usa-se essencialmente para impedir o posicionamento, dimensionamento e remoções indevidas de objectos 3D num mundo virtual. Na prática, o objectivo das permissões é desabilitar as opções *Edit* ou *Delete* do menu de edição de propriedades de objectos 3D (Figura 27), que permitem a qualquer utilizador posicionar, dimensionar e remover objectos 3D.

A aplicação de permissões a um objecto 3D, torna um mundo virtual computacionalmente mais pesado, aumentando o seu tempo de carregamento na aplicação JWS, daí os mundos virtuais desta tese não implementarem o mecanismo de permissões.

Este mecanismo de permissões, apesar de servir para impedir alguns tipos de manipulação de objectos 3D a alguns utilizadores, não protege na totalidade a integridade do mundo virtual. Não existe nenhum tipo de permissão que impeça um utilizador de adicionar ou duplicar objectos 3D indefinidamente, o que pode causar um sobrecarga do mundo virtual, prejudicando o seu desempenho. Estas situações só

podem ser resolvidas, se um administrador do servidor OW fizer o *restore* do mundo virtual ou da s*napshot*, para repor o seu estado inicial.

4.7 Funcionalidades ou mecanismos adicionais

Nesta secção abordam-se as funcionalidades ou mecanismos adicionais, úteis ao bom funcionamento da solução, mas não estão incluídas de raiz no OW, nem são vitais ao seu funcionamento. Essas funcionalidades são o arranque automático do servidor OW e a protecção contra falhas de energia eléctrica usando uma UPS, ligada à máquina SW e ao dispositivo MW.

4.7.1 Arranque automático do servidor

O arranque automático do servidor OW é uma funcionalidade que consiste em arrancar automaticamente o servidor, no arranque do SO. Pretende-se que esta funcionalidade se efectue com a simples acção de ligar a máquina SW.

Este mecanismo baseia-se na configuração de uma funcionalidade muito comum em sistemas operativos, para executar aplicações automaticamente no arranque. Foi configurado no SO da máquina SW o arranque automático do terminal ou linha de comandos, pois esta aplicação está configurada para quando é executada, executar o *shell script*[57] *loadw*, que arranca automaticamente os servidores SSH e OW. O servidor SSH permite a ligação lógica (comunicação remota) entre os utilizadores de um mundo virtual e a máquina SW. Para arrancar o servidor OW, o *script loadw* executa o comando *ant run-server* no directório apropriado.

O diagrama na Figura 31, mostra a sequência desde que um utilizador carrega no botão para ligar a máquina SW, até arrancar automaticamente o servidor OW.

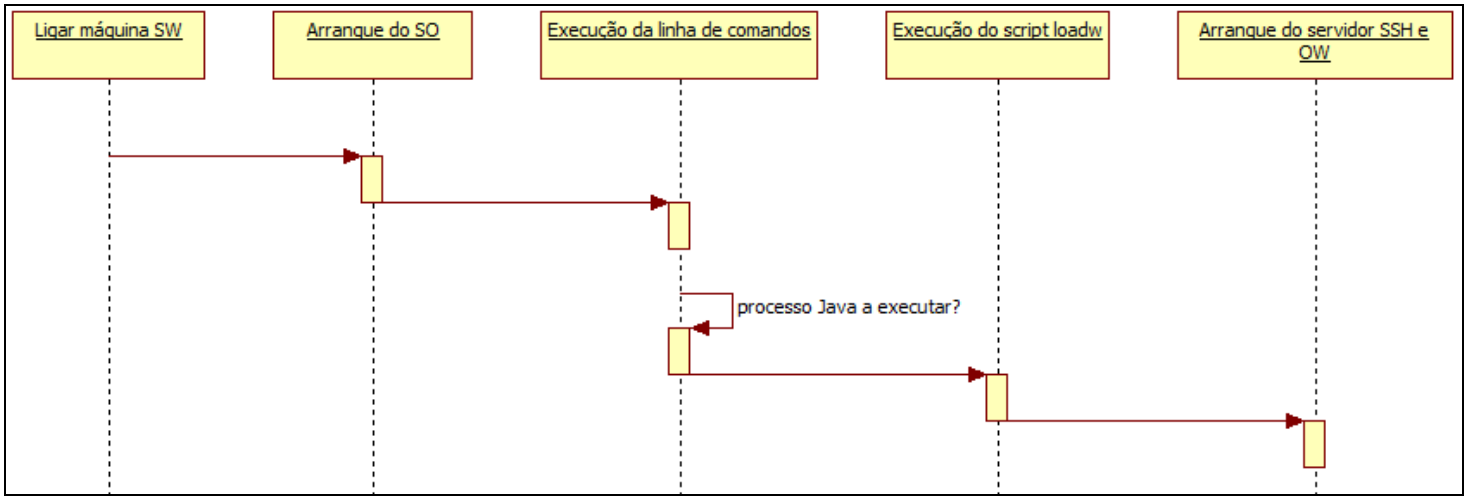


Figura 31 - Sequência do arranque automático do servidor Open Wonderland

[57] *Shell script* é uma linguagem de *script* usada em vários sistemas operativos, que permite executar múltiplos comandos/aplicações num comando só. A sua utilização é mais frequente em sistemas operativos Linux.

Cada acção do diagrama anterior provoca automaticamente a acção seguinte. Houve o cuidado do *script loadw* ser executado na abertura da linha de comandos só quando não existirem processos Java a executar no SO, pois a execução do servidor OW, está representada no SO por esses processos. Assim, se o servidor OW já estiver a executar, não é novamente arrancado pela execução do *script loadw* de cada vez que se abre a linha de comandos.

Optou-se por esta metodologia de arranque automático, por não ser possível ao SO da máquina SW executar automaticamente no arranque o comando *ant run-server* ou o *script loadw*, o que facilitaria bastante este mecanismo.

As funcionalidades de um mundo virtual só são disponibilizadas se os componentes do servidor OW estiverem activados. O seu arranque só será totalmente automatizado se a propriedade *wonderland.runner.autostart* for *true* (secção 4.2), ficando assim o mundo virtual logo disponível. Mas com a propriedade *wonderland.runner.autostart* a *true*, o arranque do servidor será mais lento, devido a activar todos os componentes. Para tornar o arranque automático mais rápido, garantindo o acesso a um mundo virtual, criou-se a propriedade de configuração *wonderland.darkstar.autostart*, também guardada no ficheiro *my.run.properties* e assumindo os valores *true* ou *false*, sendo colocada a *true* para activar automaticamente no arranque, apenas o componente *Darkstar Server*, já que é um componente nuclear para disponibilizar os mundos virtuais. Assim, o arranque do servidor OW torna-se mais eficiente, por não activar componentes não essenciais. A propriedade criada não existe de raiz no servidor OW, logo para este a reconhecer foi necessário alterar o seu código fonte, recompilá-lo e tornar a arrancar o servidor. Esta nova propriedade só funciona correctamente se a propriedade *wonderland.runner.autostart* for *false*. Na alteração do código fonte foram previstas as situações de ambas as propriedades de activação automática dos componentes serem *true*, ou a propriedade *wonderland.darkstar.autostart* ser *false*. Nesses casos prevalece o valor da propriedade *wonderland.runner.autostart*.

O mecanismo de arranque automático do servidor OW pode ser complementado com outro de protecção contra falhas de energia eléctrica, pois será muito útil para uma recuperação mais rápida de uma falha.

4.7.2 Protecção contra falhas de energia eléctrica

Na solução final, a máquina SW e o dispositivo MW encontram-se ligados a uma UPS para protecção contra de falhas de energia eléctrica. Esta ligação não se destina apenas a manter em funcionamento estes recursos, quando ocorrem as falhas referidas, pois o SO da máquina SW deve estar preparado para encerrar automaticamente, desligando posteriormente a máquina SW, caso a carga das baterias da UPS atinja um determinado nível mínimo, para evitar o desligar repentino da máquina SW, podendo acontecer se as baterias da UPS esgotarem a sua carga armazenada. Este desligar repentino, por vezes,

pode causar o funcionamento incorrecto de algumas funções dos sistemas operativos. Para o SO aceder à carga das baterias da UPS, para determinar quando deve provocar o seu encerramento, é necessário ter o *driver*[58] da UPS instalado e configurado. O modelo da UPS é uma MGE Pulsar 1500, igual à apresentada na Figura 32.

Figura 32 - UPS da solução final

Esta UPS inclui um CD de instalação com *drivers* e software de gestão da UPS, a forma mais fácil de instalar/configurar software num SO, mas o Ubuntu 9.10 não suporta os *drivers* do CD de instalação da UPS. Assim optou-se por usar *drivers* da *Network UPS Tools*, vulgarmente designada por *nut*. O pacote de software *nut* inclui uma enorme variedade de *drivers* de UPS, dos mais variados fabricantes para distribuições de Linux, esses *drivers* possuem variantes, de acordo com a ligação física entre a máquina e a UPS, podendo ser por USB[59] ou RS-232[60]. Para saber qual o *driver* adequado à situação em concreto, deve-se consultar a tabela de compatibilidades da *nut* [UPS hardware compatibility list, 2010]. A sua consulta serve para determinar qual o *driver* específico que o *nut* deve utilizar, que no caso do SO Ubuntu para a UPS MGE Pulsar 1500, é o *driver mge-shut* para porta série RS-232 (não existe *driver* para esta UPS e SO, que funcione por USB), o mais apropriado e estável.

A máquina SW e o dispositivo MW quando alimentados pelas baterias da UPS, podem manter-se ligados cerca de quinze minutos, sendo o tempo que leva a carga das baterias a diminuir dos 100% aos 75%. Não se optou por um valor mínimo de carga mais baixo, porque foram efectuados testes com valores menores que 75%, que fizeram a UPS desligar-se repentinamente, causando o desligar abrupto da máquina SW.

As funcionalidades referidas nesta secção, vêm melhorar a disponibilidade do servidor OW e do dispositivo MW, pois ao estarem ligados à UPS, manter-se-ão a funcionar caso a energia eléctrica

[58] Um *driver* é uma aplicação que estabelece a interface entre o SO e seu hardware correspondente.
[59] USB: tipo de conector para ligação a dispositivos electrónicos. O seu uso mais vulgar é a ligação entre um PC e um dispositivo externo.
[60] RS-232: conector padrão para ligações série em dispositivos electrónicos.

falhe. Se a falha se prolongar até se atingir os 75% de carga nas baterias, será despoletado o encerramento automático do SO da máquina SW, um minuto depois. Quando a corrente eléctrica voltar a estar disponível, o servidor OW será automaticamente arrancado com o arranque automático do SO ao ligar a máquina SW. O dispositivo MW terá de ser ligado localmente.

A disponibilidade desta solução é decisiva para melhorar a disponibilidade/fiabilidade do MV e de um mundo virtual, para uma melhor utilização. O desenvolvimento do módulo do MV é abordado no próximo capítulo.

5. Módulo do Multímetro Virtual

Toda a implementação da solução final, descrita no capítulo anterior, só faz sentido se num mundo virtual existir uma reprodução virtual do multímetro, que possibilite aos utilizadores controlar remotamente esse dispositivo localizado no laboratório. Essa reprodução virtual foi designada como MV, e corresponde ao componente Módulo Multímetro, na arquitectura lógica da solução (Figura 14), abordada na secção 3.4, que se baseia num módulo OW.

Este capítulo incide sobre a construção do componente Módulo Multímetro, destacando os seus pormenores técnicos. Para isso, foi necessário estudar a composição dos módulos OW, baseando-se em exemplos existentes. Daí adquiriu-se os conhecimentos necessários para construir um módulo de acordo com as funcionalidades requeridas para o MV. Assim, nas secções seguintes, aborda-se como foram desenvolvidos: o aspecto gráfico do MV, mecanismos de interacção com o utilizador, comunicação remota com o MR, e desenvolvimento de um mecanismo de acessos concorrentes.

5.1 Composição de um módulo Open Wonderland

Um módulo OW baseia-se num ficheiro JAR, logo é código Java compilado. Nesta tese, o módulo do MV foi construído com o auxílio do IDE Netbeans[61] *6.8*. Este dispõe de uma documentação mais vasta, sendo também dos que é mais fácil integrar bibliotecas de código externo quando necessário. A página *Web* do OW recorre ao Netbeans para explicar, nos seus tutoriais, a programação de várias funcionalidades na construção de módulos.

O módulo não foi totalmente desenvolvido de raiz, baseia-se num outro de exemplo, denominado *samplemodule*, disponível para *download* na página *Web* do OW. O mesmo inclui a estrutura de ficheiros necessária para compilar e funcionar correctamente, possibilitando a sua instalação no servidor OW, para poder ser adicionado num mundo virtual.

Depois, na construção do módulo, basta desenvolver o código requerido ao resultado final pretendido, e adicionar as bibliotecas de código necessárias ao seu funcionamento, num mundo virtual, ou para outras funcionalidades impossíveis de programar apenas com as API's do Java e do OW.

Um módulo OW, que se baseie em figuras ou sólidos geométricos, requer bibliotecas externas como a API do *jMonkeyEngine* (*jme*)[62], e outras provenientes do código fonte OW. Todas elas vêm incluídas no *download* do mesmo. O aspecto gráfico do MV é essencialmente constituído por sólidos geométricos, com posições e tamanhos próprios, com o objectivo da sua aparência coincidir o mais possível com o MR.

[61] Netbeans é um software ou ambiente de programação, que suporta várias linguagens, entre as quais Java. Inclui um editor/interface de auxílio à escrita de código (IDE) e compiladores das várias linguagens de programação.
[62] *jMonkeyEngine* é uma tecnologia para construção de objectos 3D na linguagem Java.

Para um módulo funcionar correctamente num mundo virtual, deve respeitar uma estrutura própria de *packages*[63] [Slott, J., 2010 b], já que o código do módulo irá interagir com código do OW. A Figura 33 exemplifica a estrutura da organização do código no módulo do MV.

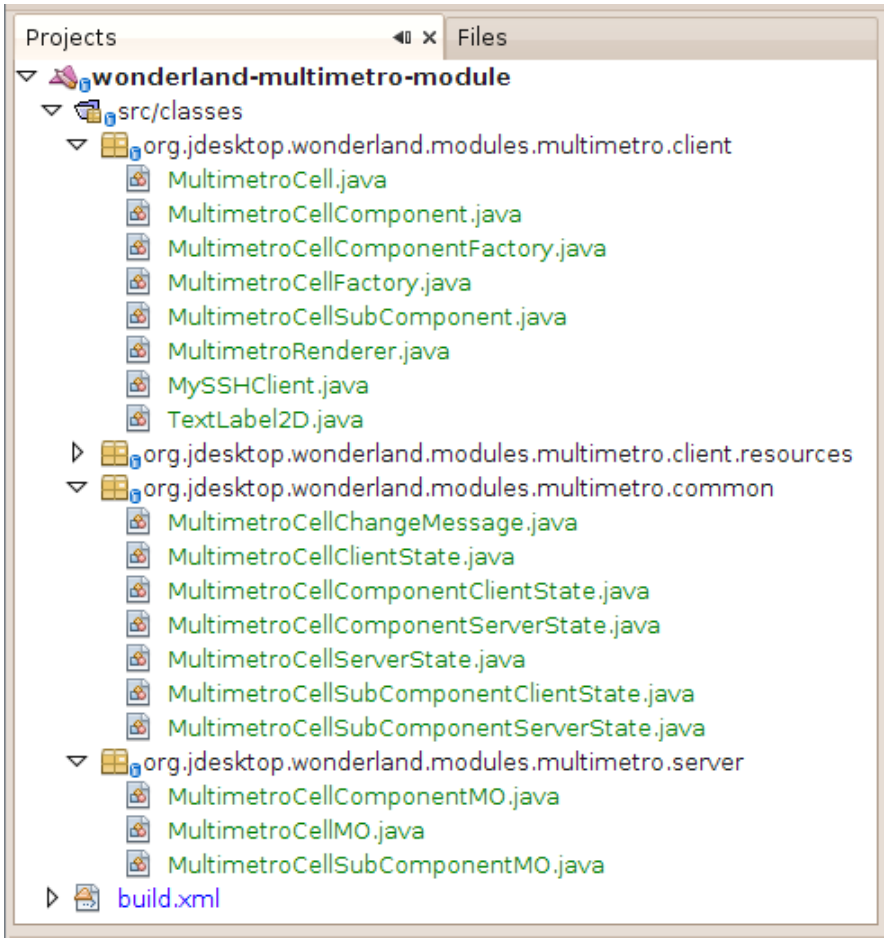


Figura 33 - Estrutura de organização do código do módulo do Multímetro Virtual

O projecto do módulo do MV, denominado *wonderland-multimetro-module*, é constituído pelos seguintes *packages*:

- ***org.jdesktop.wonderland.modules.multimetro.client*:** contém os ficheiros com o código que é executado do lado do cliente OW, ou seja, num mundo virtual na aplicação JWS.
- ***org.jdesktop.wonderland.modules.multimetro.client.resources*:** este *package* é o único que não tem ficheiros com código Java. Os seus ficheiros contêm variáveis de propriedade, depois usadas no código de outros ficheiros nos restantes *packages*.
- ***org.jdesktop.wonderland.modules.multimetro.common*:** os ficheiros com código que é executado tanto do lado do cliente como do lado de servidor do OW encontram-se neste *package*.

[63] Um *package* é um pacote que alberga um ou vários ficheiros de código Java. A sua utilidade é fazer com que funções de código presentes num ficheiro possam ser invocadas no código de outro ficheiro qualquer, desde que esteja no mesmo projecto do IDE.

- ***org.jdesktop.wonderland.modules.multimetro.server*:** *package* cujo código dos seus ficheiros executa apenas do lado do servidor do OW.

É obrigatório num módulo OW, que o nome dos seus *packages* comece por *org.jdesktop.wonderland.modules*, para o OW saber que se trata de um módulo, pois o código do módulo interage com o código do OW.

O projecto Netbeans, *wonderland-multimetro-module* é do tipo Java *Free-Form Project*, que consiste em aproveitar características de aplicações Java já construídas. Neste caso, trata-se de aproveitar a estrutura existente do código fonte do OW. Para este tipo de projectos, o código é compilado usando o *Apache Ant*, tal como na compilação de outros módulos, que segue as instruções XML que estão no ficheiro *build.xml* (Figura 33), onde se deve efectuar qualquer alteração à forma de compilação do módulo.

Recomenda-se que na construção de um módulo OW se teste sempre o seu aspecto e comportamento no mundo virtual, o que implica correcções do código, compilações, e instalações sucessivas do módulo no servidor OW, e assim sucessivamente, até se obter o resultado pretendido.

5.2 Funcionalidades

Como se verificou na secção 3.2, um multímetro é um instrumento de medição de variáveis eléctricas. Para o MV estipulou-se que as variáveis suportadas numa primeira fase seriam: a tensão contínua e alternada, e a corrente contínua e alternada. Na construção do módulo do MV, a metodologia para aceder ao MR é sempre a mesma, independentemente das variáveis sob medição, assim, optou-se pelas mais básicas num sistema/circuito eléctrico.

Para conseguir medir as variáveis seleccionadas para o módulo, recorre-se a funções Java, que executam e retornam o resultado de comandos de sistemas operativos, através de ligações SSH ou *telnet*, daí requerer-se uma ligação SSH entre os utilizadores e a máquina SW (entre aplicação JWS e *MyTelnetCmdClient* na arquitectura lógica na secção 3.4), e outra *telnet* entre a mesma e o dispositivo MW. Os comandos SCPI servem para efectuar medições remotas sobre o MR. Assim, para as medições serem apresentadas no MV, esses comandos são executados por funções Java. Os comandos SCPI para as variáveis suportadas pelo MV são:

- ***meas:volt:dc? 1*:** medição da tensão contínua até um valor máximo de um volt (1V).
- ***meas:volt:ac? 1*:** medição da tensão alternada até um valor máximo de um volt (1V).
- ***meas:curr:dc? 10A*:** medição da corrente contínua até um valor máximo de dez amperes (10A).
- ***meas:curr:ac? 10A*:** medição da corrente alternada até um valor máximo de dez amperes (10A).

A execução destes comandos requer a execução do comando *syst:rem*, que coloca o dispositivo MW em modo de acesso remoto. No acesso remoto ao MR, a partir do MV, é necessário salvaguardar que o dispositivo MW pode ser requerido para utilização local, nesse caso, coloca-se o mesmo em modo de funcionamento local, através do comando *syst:loc*.

Os valores máximos das medições dos quatro comandos anteriores foram estipulados para contornar o valor máximo estipulado, definido localmente no dispositivo MW. Se esses valores fossem omitidos nos comandos, estes executariam com qualquer valor máximo definido localmente.

As próximas secções deste capítulo detalham a construção do módulo do MV, nos seus aspectos mais relevantes.

5.3 Aspecto gráfico

Nesta secção aborda-se a construção/programação do aspecto gráfico do MV e a sua interacção com o utilizador. Com o projecto do módulo no IDE Netbeans devidamente preparado, pode-se proceder à programação gráfica do MV propriamente dita. O seu desenho baseia-se em sólidos geométricos, tais como paralelepípedos ou prismas quadrangulares. O aspecto gráfico de um módulo do OW baseia-se em bibliotecas de código do *jme* (secção 5.1), que disponibilizam objectos e funções para o desenho gráfico 3D do MV.

5.3.1 Desenho do Multímetro Virtual

Num módulo do OW, o desenho de um objecto 3D é executado na função *createSceneGraph* no ficheiro *MultimetroRenderer.java*. Um sólido geométrico de formas rectangulares é desenhado com o objecto *Box*, que permite o posicionamento e dimensionamento do objecto 3D (sólido geométrico) num mundo virtual, com base nas coordenadas X, Y e Z. Nas Figuras 34 e 35 estão representadas a parte frontal e lateral esquerda do MV, respectivamente.

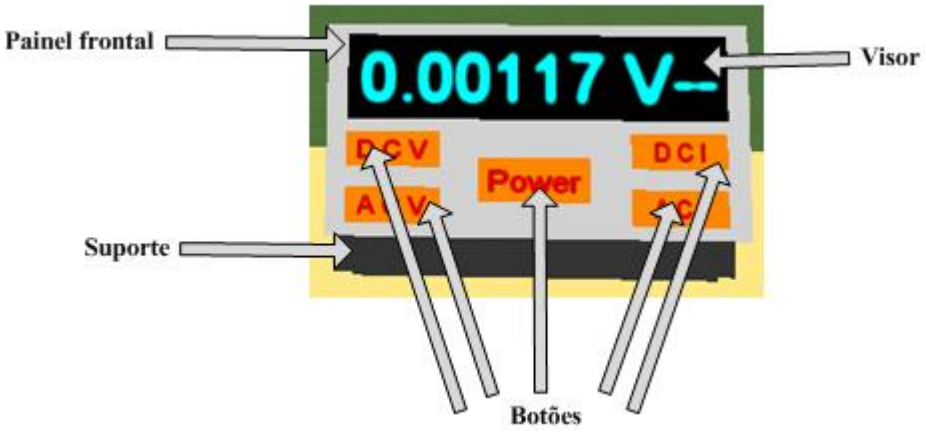


Figura 34 - Parte frontal do Multímetro Virtual

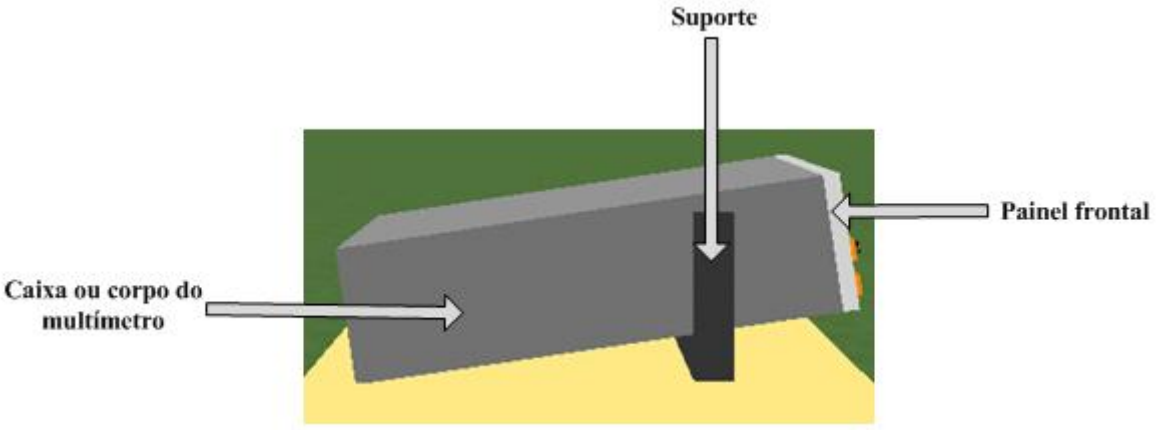


Figura 35 - Parte lateral esquerda do Multímetro Virtual

As formas do dispositivo foram desenhadas, com recurso à função *getShapeMesh*, que retorna um objecto do tipo *Box*. A cor de cada forma também é definida nesta função. Tendo em conta as perspectivas do MV nas duas figuras anteriores, este decompõe-se nas seguintes partes, indicadas nas figuras: caixa ou corpo do multímetro, painel frontal, visor, botões e suporte.

A simples criação dos objectos gráficos, não é suficiente para definir o seu posicionamento, ou para se ter um módulo que represente a aparência de um multímetro. O posicionamento (também pode designar-se por translações, como termo mais técnico) e dimensionamento de todas partes do MV, também foram feitos na função *getShapeMesh*. A inclinação do MV em relação ao seu suporte (Figura 35) foi baseada em rotações. As mesmas foram aplicadas a todas as partes do dispositivo, à excepção do suporte.

O desenho de cada parte do MV num mundo virtual, requer que um objecto 3D (sólido geométrico) do tipo *Box* seja adicionado a um objecto do tipo *Node*. Este tipo é uma classe da API do *jme*, que serve para reservar memória para os objectos 3D. É a partir de objectos *Node*, que se agrupam as diversas partes, até se constituir o MV. Um *node* pode conter mais objectos 3D, ou mais objectos *Node*, sendo o tipo de retorno da função *createSceneGraph*.

O texto exibido no visor e nos botões do MV na Figura 34, foi conseguido com a utilização da classe *TextLabel2D* [Billboard AWT Label, 2008] do ficheiro de código *TextLabel2D.java*. Esta classe recorre a bibliotecas da API do OW. A sua utilização foi necessária devido à classe *Text* do *jme* não suportar texto 100% estático para o OW. A criação do texto a aplicar ao visor e aos botões é feita na função *getLabel*.

5.3.2 Interacção com o utilizador

A interacção com o utilizador no MV é baseada em botões, tal como num dispositivo real, sempre que se carrega num botão espera-se um determinado comportamento. O MV apresenta cinco botões, que de acordo com a Figura 34 são:

- ***Power***: Liga e desliga o MV. Na Figura 34 está ligado, quando desligado fica com o visor totalmente preto, sem valor numérico e com o texto dos botões também a preto.
- **DCV**: efectua a medição de tensão contínua.
- **ACV**: efectua a medição de tensão alternada.
- **DCI**: efectua a medição de corrente contínua.
- **ACI**: efectua a medição de corrente alternada.

A nomenclatura dos botões DCV, ACV, DCI e ACI, é proveniente dos mesmos botões com função igual no MR. Cada botão é baseado num objecto gráfico do tipo *Box*, não é a forma mais usual de se utilizar botões, já que existem bibliotecas de funções Java, que incluem este tipo de objecto de interacção, sendo mais fáceis de programar. A implementação dos botões do MV chegou a ser equacionada de duas formas: a primeira foi usar uma aplicação *JFrame*[64] no lugar do painel frontal, ideia logo abandonada por não permitir um clique directo no botão, pois implicava activar uma janela de execução em primeiro lugar; a segunda forma foi usar a bilioteca *Fenggui*, que disponibiliza este tipo de objectos GUI[65] para o *jme*, mas descobriu-se que não é suportada pelo OW. Assim, para programar o objecto gráfico, que representa um botão a reagir a acções do utilizador por via do rato, tendo depois a acção correspondente, preparou-se o sistema de colisões do módulo do MV na função *setStatus* no ficheiro *MultimetroRenderer.java*. Um sistema de colisões consiste em fazer, com que o MV não possa ser atravessado, por exemplo, no movimento do *avatar*. O movimento do *avatar*, ao chegar ao objecto 3D, tem de ser parado ao colidir com o mesmo. No OW, o sistema de colisões torna os *nodes* sensíveis às acções do rato, para se poder organizar o código de cada botão de acordo com a sua funcionalidade. Se os *nodes* dos botões não fossem sensíveis a acções do rato, qualquer clique num deles seria interpretado de forma igual como clique no MV.

A programação das acções de um botão é efectuada na função *commitEvent*, pertencente à classe *MouseEventListener*, no ficheiro *MultimetroCell.java* [Slott, J., 2010 c]. Existe código que executa conforme uma tecla do rato seja pressionada ou solta.

Para dar a um objecto 3D a aparência do comportamento de um botão, deve-se verificar qual a tecla do rato carregada, e se está pressionada ou solta. A tecla do rato a usar para carregar nos botões do MV é a esquerda, o pressionar da tecla significa carregar sem soltar, a acção de soltar corresponde ao deixar de a pressionar (largar). O código usado na acção de pressionar a tecla do rato serve para dar o efeito de botão carregado ao objecto 3D, visível no botão *Power* na Figura 36.

[64] *JFrame* é uma classe da API do Java, que possibilita a construção de interfaces gráficas, com objectos de interacção com o utilizador.
[65] GUI: corresponde a objectos de interacção com o utilizador como caixas/etiquetas de texto ou botões.

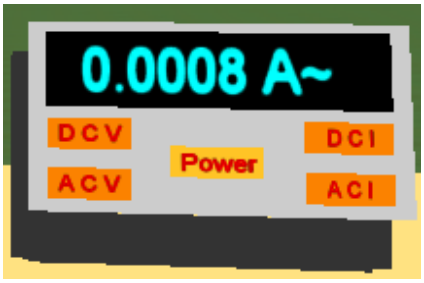


Figura 36 - Multímetro com botão *Power* pressionado

Quando a tecla pressionada na figura anterior for solta, ficará com o aspecto semelhante à mesma da Figura 34.

No bloco de código de cada botão, para as acções de pressionar ou soltar a tecla, é invocada a função *sendDraw*, que invoca a função *updateShape*, sendo esta responsável por desenhar as acções dos botões no MV. A função *sendDraw* encontra-se na classe *MultimetroCell* no ficheiro *MultimetroCell.java*, a função *updateShape* encontra-se no ficheiro *MultimetroRenderer.java*.

O desenho do efeito dos botões do MV, consiste em actualizar os seus *nodes*, essa actualização baseia-se em remover do mundo virtual os *nodes* dos objectos 3D, para depois serem recriados/redesenhados com um objecto 3D, de acordo com a aparência pretendida.

Um multímetro é controlado usando os seus botões, cada um com uma dada funcionalidade, que só pode ser utilizada se o mesmo estiver ligado. No caso do módulo MV, a função de ligar/desligar é feita no botão *Power*, que actua sobre a variável *estado*, que faz parte de classe *MultimetroCell*. Essa variável alterna entre dois valores: 0 (zero) significa que o MV está desligado, e 1 significa que está ligado. A variável é inicializada a 0, por omissão, já que o MV é colocado desligado no mundo virtual, se ainda não tiver sido utilizado. O valor da variável é alterado no código do botão *Power*, quando este é solto pela tecla esquerda do rato, provocando também uma mudança de cor do texto dos botões de preto para vermelho ao ligar o MV, e vice-versa ao desligar.

No MV para os botões DCV, ACV, DCI e ACI só funcionarem se o mesmo estiver ligado, à semelhança do que acontece num MR, o código dos botões referidos só é executado se a variável *estado* for 1.

Todo o processamento de acção de um botão do MV encontra-se resumido no diagrama de sequência da Figura 37, que ilustra as classes que contêm as funções executadas nesse processamento.

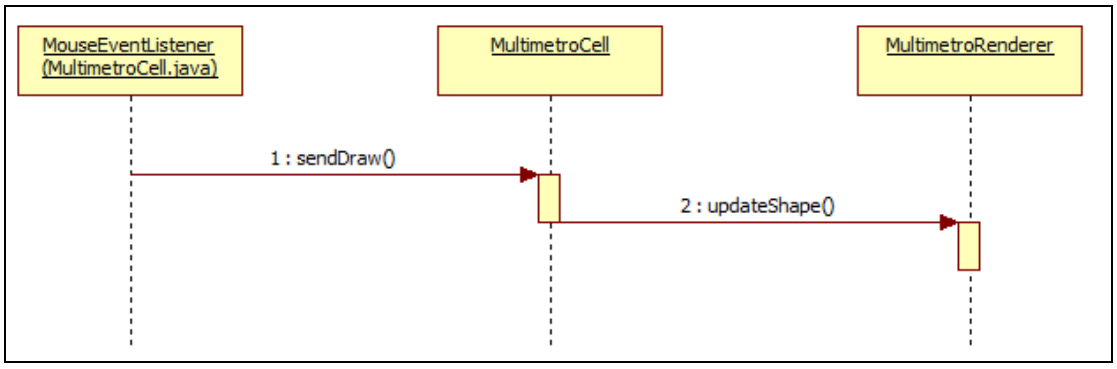


Figura 37 - Acção de redesenho de um botão no módulo do Multímetro Virtual

Um objecto 3D de um módulo do OW, sensível a acções do rato, reage desde que o cursor do mesmo esteja sobre o objecto, independentemente da distância deste para o *avatar*. Isso é inadequado para a utilização do MV. Num laboratório real para se manipular um dispositivo electrónico, seja um multímetro ou outro, o utilizador tem de aproximar-se a uma distância suficiente, para ter os controlos ao seu alcance. No módulo do MV, esta premissa é simulada pelo cálculo de uma distância linear, entre o *avatar* e o MV, ou seja, os seus botões só são activados pelo utilizador, se o seu *avatar* estiver a uma distância igual ou inferior a um valor pré-definido. O cálculo da mesma é feito, pela função *getDistance* no ficheiro *MultimetroRenderer.java*, e consiste em obter as coordenadas X, Y e Z tanto do MV como do *avatar*, recorrendo à seguinte função matemática (Equação 1).

$$\text{Distância} = \sqrt{(x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2}$$

Equação 1

5.3.3 Sincronismo do Multímetro Virtual entre utilizadores

Quando um utilizador está a utilizar o MV no mundo virtual, as suas acções não são visíveis pelos outros utilizadores, que também lá estejam naquele momento, dado que um módulo OW não implementa de raiz qualquer mecanismo de sincronismo. O que acontece, no caso do MV não implementar mecanismos de sincronismo, é um utilizador estar a carregar num botão e essa acção não ser visível pelos outros utilizadores, quando ligados no mesmo mundo virtual, no momento em que o botão é carregado, ou seja, cada utilizador utiliza o MV isoladamente dos restantes. Assim, foi implementado no módulo do MV, um mecanismo de sincronismo baseado em envio de mensagens.

Este mecanismo de sincronismo pode ser implementado em módulos OW, que tenham funcionalidades de interacção com o utilizador. Assim, as acções de um utilizador são visíveis nos restantes, devido a uma mensagem ser enviada do utilizador que actua (por exemplo, ao carregar num objecto 3D) para os restantes, no momento da acção. O conteúdo das mensagens são variáveis do módulo, a acção só é desenhada em todos os utilizadores, porque o conteúdo dessas variáveis vai ser depois armazenado nas variáveis dos utilizadores que recebem a mensagem. Para todos os utilizadores ligados receberem as mensagens, o módulo tem de ser preparado para estar à escuta das mesmas. O

percurso de uma mensagem no módulo consiste na mesma ser construída e enviada, por código executado do lado do cliente, e depois propagada por código do lado do servidor, para ser recebida nos restantes utilizadores, através de código do lado do cliente à escuta.

No caso do módulo do MV, uma mensagem é construída e enviada na função *sendDraw*. O seu percurso é ilustrado na Figura 38, com as classes e funções que o executam.

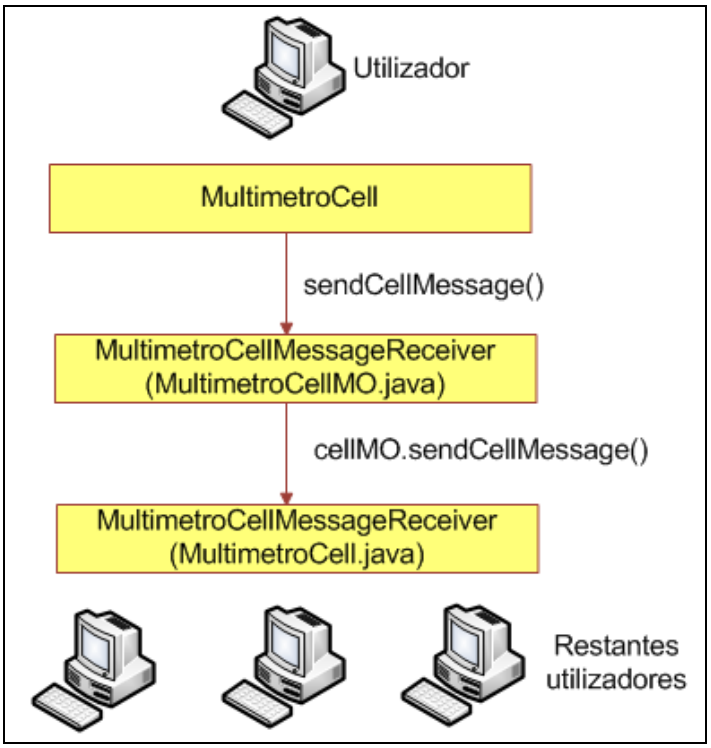


Figura 38 - Percurso de uma mensagem de sincronismo num módulo Open Wonderland

O desenho de uma acção nos restantes utilizadores é feita pela execução da função *updateShape* (na função *messageReceived* da classe *MultimetroCellMessageReceiver* no ficheiro *MultimetroCell.java*), com as variáveis contidas na mensagem, no código que a recebe.

5.4 Comunicação com o Multímetro Real

É através de protocolos de comunicação de redes de computadores que o módulo do MV comunica com o MR. Assim, esta secção constitui-se como o núcleo desta tese, pois descreve todo o processo de comunicação para acesso remoto do MV ao MR. O MR é o dispositivo MW ligado à rede interna do CIETI-LABORIS (arquitectura física da solução na secção 3.3), acedido remotamente pelo protocolo *telnet*, suportando um acesso remoto de cada vez.

A máquina SW é acessível do exterior da rede do CIETI-LABORIS, e dispõe de um servidor SSH, para que um utilizador possa aceder-lhe e executar comandos remotamente, de qualquer ponto da Internet. É através de um comando remoto, via SSH, que se executa a aplicação *MyTelnetCmdClient*.

Esta funciona na linha de comandos do SO da máquina SW, e retorna os valores de medição, efectuados/retornados pelo MR, através de uma ligação *telnet*. A aplicação *MyTelnetCmdClient* é executada pelo módulo do MV, sempre que um botão deste é carregado. Cada botão corresponde a uma medição, determinando o parâmetro da aplicação *MyTelnetCmdClient*, para executar no MR o comando SCPI de acordo com a medição correspondente.

5.4.1 Ligação SSH

Para apresentar um valor de medição no visor do MV, existe a função *getCommand* na classe *MultímetroCell*, que executa remotamente o comando da aplicação *MyTelnetCmdClient* na máquina SW. Essa função executa sempre que se carrega num botão do MV. O retorno da função é o retorno da aplicação *MyTelnetCmdClient*, que é o resultado da medição efectuada no MR. Esse resultado/retorno é depois guardado na variável *display*, pela função *getCommand*. Essa variável é depois usada nos parâmetros das funções *sendDraw* e *updateShape*, para exibir o valor retornado no visor do MV. Na altura da medição, o valor da variável *display* também estará no visor do MR. Na função *getCommand*, existe um parâmetro para indicar à aplicação *MyTelnetCmdClient* qual o comando SCPI (ver secção 5.2) que será executado.

A função *getCommand* executa a aplicação *MyTelnetCmdClient*, através de uma ligação remota SSH à máquina SW. O módulo do MV está preparado para implementar funcionalidades para o protocolo SSH, utilizando a biblioteca *Ganymed SSH-2*, que fornece classes com funções para acesso remoto SSH, programável em Java. Para usar essas classes, a biblioteca deve ser adicionada ao projecto do módulo do MV no Netbeans. A selecção da biblioteca *Ganymed SSH-2* deve-se ao facto de ser a mais simples de programar, gratuita e com uma documentação de fácil leitura e compreensão.

As funções de comunicação SSH no módulo MV foram implementadas na construção da classe *MySSHClient*. Para estabelecer uma ligação SSH, do MV para a máquina SW, existe na classe *MultímetroCell*, a função *connect* que cria um objecto *MySSHClient*. A criação deste objecto, estabelece a ligação SSH, especificando o endereço *Web*, porto da máquina e credenciais (*login* e *password*) para autenticação do utilizador (caso seja requerida), onde se pretende conectar. No caso do módulo do MV, é estabelecida uma ligação SSH para a máquina SW, com o endereço *Web* *www.laboris.isep.ipp.pt* no porto 1140, e com as credenciais de um utilizador, criado no SO. Foi criado um utilizador exclusivamente para acesso remoto do MV para a máquina SW, por questões de segurança e organização dos utilizadores do SO.

Com a ligação SSH estabelecida, a função *getCommand* invoca a função *readCommand*, na Instrução 1.

```
cmdRet=msc.readCommand("java -jar /home/multimetro/MyTelnetCmdClient/MyTelnetCmdClient.jar "+op);
```

Instrução 1

A função *readCommand* da classe *MySSHClient*, é responsável por executar o comando da aplicação *MyTelnetCmdClient* (com o ficheiro *MyTelnetCmdClient.jar*) na máquina SW, para obter o valor da medição do MR e colocá-lo no visor do MV. O parâmetro *op* especifica o argumento da aplicação *MyTelnetCmdClient*, de acordo com a medição pretendida.

Durante uma ligação SSH entre o MV e a máquina SW, a execução/escrita de comandos e posterior leitura do seu retorno, baseiam-se em *streams*[66] de dados, à semelhança do que é usado na programação de leitura/escrita de ficheiros. A execução do comando da Instrução 1 é auxiliada por um *stream* de saída/escrita, e a leitura do retorno (para a variável *cmdRet*) é feita por um *stream* de entrada/leitura.

Basicamente, de cada vez que um botão do MV é carregado, a função *getCommand* cria a ligação SSH (função *connect*), que obtém o resultado de um comando (função *readCommand*), terminando depois a ligação (função *disconnect*). Esta pode não ser a metodologia mais eficiente para ligações remotas, por obrigar sempre a estabelecer/terminar uma ligação, quando o estabelecimento de uma ligação é das operações mais demoradas. Neste caso, o tempo perdido não é suficientemente longo, para pôr em causa tempos de resposta minimamente aceitáveis. A metodologia da ligação SSH na função *getCommand*, foi aproveitada de versões anteriores do módulo do MV, quando se tentou que este acedesse directamente ao MR via *telnet*, pois seria a melhor forma de o utilizar a partir do mundo virtual num cenário multi-utilizador, devido a não suportar várias ligações remotas em simultâneo. Assim, o MV pode ser utilizado em simultâneo por um número teoricamente infinito de utilizadores, dependendo da capacidade do servidor OW, dispensando também a configuração do número máximo de utilizadores no servidor SSH da máquina SW. Teoricamente, seria mais eficiente a operação de estabelecer/ terminar a ligação SSH apenas quando o botão *Power* do MV fosse carregado, ou seja, a ligação só seria estabelecida/terminada ao ligar/desligar o MV, assim, os restantes botões só executariam o comando relativo à sua funcionalidade, tornando as medições mais rápidas por não terem de estabelecer/terminar a ligação SSH.

5.4.2 Leitura e tratamento do valor de medição

É a aplicação *MyTelnetCmdClient*, abordada na secção 3.4, que permite obter valores das medições efectuadas pelo MR. Foi construída na linguagem Java, para funcionar na linha de comandos do SO, neste caso, na máquina SW. O comando que executa esta aplicação está no parâmetro da função *readCommand* (Instrução 1), o mesmo pode ser executado na linha de comandos do SO da máquina SW com o respectivo argumento. O resultado do comando é exibido (*Output* 2) na linha de comandos, em vez de ser tratado para exibição no visor do MV.

[66] Um *stream* é um fluxo de dados em memória num SO. Por exemplo, ao abrir um ficheiro para edição, ele é colocado num *stream* para efectuar essa edição.

As funções que possibilitam a programação de uma ligação *telnet*, entre a máquina SW e o dispositivo MW, são provenientes da biblioteca *Apache Commons Net*, uma biblioteca Java gratuita que inclui funções para programação com vários protocolos de rede, incluindo *telnet*. À semelhança da biblioteca *Ganymed SSH-2*, também é de fácil programação, com documentação de fácil leitura, compreensão e acessibilidade. Na aplicação *MyTelnetCmdClient*, a leitura de resultados de um comando e sua execução também se baseia em *streams* de entrada/saída, durante a ligação. Esta aplicação foi construída no Netbeans, integrando a biblioteca *Apache Commons Net*.

Em toda a aplicação *MyTelnetCmdClient*, existe apenas uma classe com o mesmo nome. A ligação *telnet* entre a máquina SW e o dispositivo MW é estabelecida na criação do objecto dessa classe. Nessa ligação, especificam-se o nome/endereço do MW e seu porto de acesso na rede do CIETI-LABORIS. Esse dispositivo não requer autenticação. Depois de estabelecida a ligação, é executada a função *write*, que aproveita o *stream* de saída para executar um comando SCPI no MR. A leitura do valor retornado é feita pela função *read* ao ler o *stream* de entrada.

Ambas as funções *write* e *read*, são executadas na função *getCommand* que executa o comando SCPI no MR, de acordo com a medição pretendida, especificada no argumento do comando da aplicação *MyTelnetCmdClient*. Antes da execução da função *getCommand*, é executado com a função *write*, o comando SCPI *syst:rem*, para colocar o dispositivo em modo de funcionamento remoto.

Foi abordado na secção 3.2, que o valor das medições retornadas pelo MR é colocado em notação científica (*Output* 2) sempre que este é operado remotamente, apesar de ser o mesmo valor que surge no visor do MR, a forma numérica apresentada no mesmo é em notação decimal (Figura 12, secção 3.2). Apresentar o valor da medição em notação científica no visor do MV, é desadequado para o utilizador do mundo virtual, pois não coincide com a forma numérica apresentada no visor do MR. Assim, a aplicação *MyTelnetCmdClient*, na função *getCommand*, faz a conversão do valor lido de notação científica para decimal, para o valor exibido ser o mais semelhante possível tanto no visor do MV como do MR. A conversão de notação científica para decimal baseia-se, na conversão/tratamento de variáveis do tipo *string* para *double*, com o auxílio das classes *NumberFormat* e *DecimalFormat* da API do Java. Essas classes têm funções para conversão para notação decimal, com uma determinada precisão. Tendo em conta a largura do visor do MV, o valor de medição é truncado[67] para ter menos um dígito que no MR, e assim evita-se um aumento do erro do valor numérico, normalmente causado por arredondamentos.

O valor de medição depois de convertido para notação decimal, é convertido em *string*, sendo esta concatenada com a indicação da variável eléctrica do valor medido. No *Output* 3 está exemplificada a

[67] Truncar um número consiste em omitir, pelo menos, um dos seus algarismos menos significativos.

execução da aplicação *MyTelnetCmdClient*, na linha de comandos do SO, onde se efectuou uma medição de tensão contínua.

```
laboris@laboris-salaestudo:~$ java -jar /home/multimetro/MyTelnetCmdClient/MyTelnetCmdClient.jar 1
0.00125 V--
```

***Output* 3**

Cada execução da aplicação *MyTelnetCmdClient* efectua uma ligação *telnet* ao dispositivo MW, que termina imediatamente após a exibição do resultado. Um resultado semelhante será colocado no visor do MV, através da execução dessa aplicação, pela função *readCommand* (Instrução 1), numa ligação SSH entre um utilizador e a máquina SW.

Todo o processo de comunicação, desde o utilizador que carrega num botão do MV, até ao retorno de um valor pelo MR para o visor do MV, é resumido pelo diagrama da Figura 39 com as classes e funções intervenientes no processo.

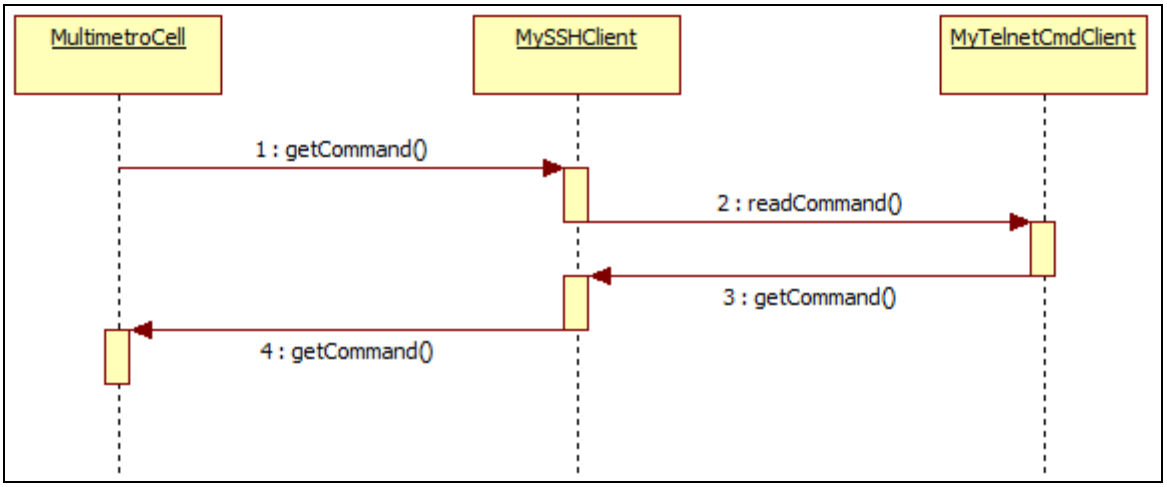


Figura 39 - Sequência de comunicação entre o Multímetro Virtual e o Multímetro Real

Na prática, uma acção representada pelo diagrama anterior traduz-se no resultado da Figura 40, onde foi efectuada, a partir do MV, uma medição de tensão contínua provocada pelo carregar no seu botão DCV. O valor apresentado no visor do MV (retornado pelo MR através da aplicação *MyTelnetCmdClient*) assemelha-se ao do visor do MR.

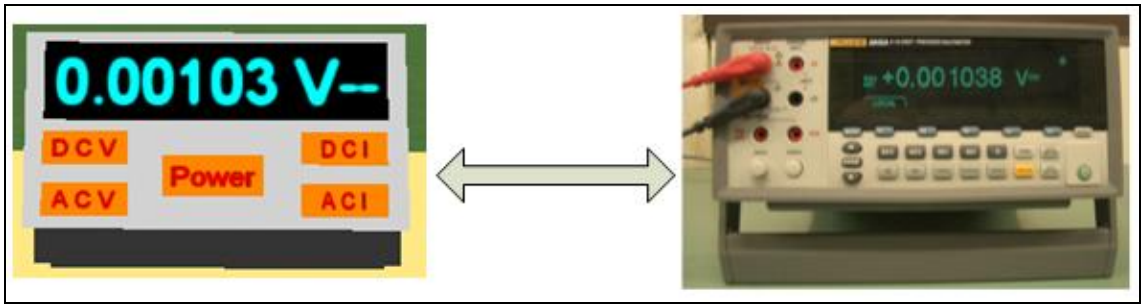


Figura 40 - Medição remota de tensão contínua

Os valores apresentados nos visores do MV e do MR são exactamente o mesmo, mas truncado no MV. O MR deve estar sempre ligado para receber ligações remotas, daí é colocado em modo de funcionamento local quando o MV é desligado, para ser libertado para eventuais tarefas que não se enquadram nesta tese. A Figura 41 exemplifica o MV desligado e seu efeito no MR.

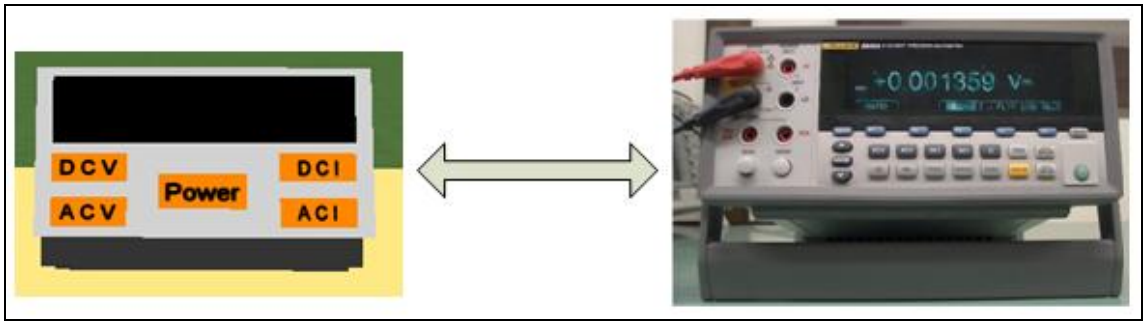


Figura 41 - Multímetro Virtual desligado e seu efeito no Multímetro Real

5.4.3 Manutenção do estado

No módulo do MV é possível manter o estado deste no mundo virtual, ou seja, se um utilizador sair do mundo virtual (aplicação cliente JWS) deixando o MV ligado, quando o mesmo ou outro utilizador entrar no mundo virtual, o dispositivo estará ligado e apresentará no seu visor o valor da última medição efectuada. Se o MV for deixado desligado ao sair do mundo virtual, este manter-se-á desligado até um utilizador entrar no mundo virtual e ligar o dispositivo. Para suportar esta funcionalidade existem espalhadas por todas as classes do módulo do MV, as variáveis *estado* do tipo *int*, e *display* do tipo *string*. A primeira é a mesma que indica ao botão *Power* (ver subsecção 5.3.2) para ligar ou desligar o MV (assume os valores 0 para desligado ou 1 para ligado), a segunda armazena o valor de medição a ser colocado no visor do MV.

A metodologia da manutenção do estado do MV, consiste no conteúdo das variáveis *estado* e *display* ser guardado no servidor OW, através de código no ficheiro *MultimetroCellMO.java*. Mesmo se o componente *DarkStar Server* for desactivado, o conteúdo das variáveis *estado* e *display* só se perde, fazendo um *restore* ao mundo virtual. O valor da variável *estado* é apenas modificado no código do botão *Power* ao ligar/desligar o MV, enquanto que, o valor da variável *display* é modificado no código de qualquer botão, ao efectuar uma medição.

Sempre que um utilizador se liga ao mundo virtual, o conteúdo guardado nas variáveis *estado* e *display* na classe *MultimetroCellMO*, é captado por intermédio da classe *MultimetroCellClientState*, com a função *setClientState* na classe *MultimetroCell*, que é a primeira função a executar no carregamento de um módulo OW, quando um utilizador se liga a um mundo virtual, com a aplicação JWS.

O estado em que o MV foi deixado anteriormente no mundo virtual é redesenhado na função *createSceneGraph*, pelo valor armazenado nas variáveis *estado* e *display*.

5.4.4 Tratamento de falhas

Na comunicação entre o MV e o MR podem acontecer os seguintes tipos de falhas:

- **Falha na ligação SSH:** relacionada com anomalias na ligação SSH, entre o mundo virtual de um utilizador e o servidor SSH a funcionar na máquina SW. Pode ser provocada por falhas de comunicação na rede de computadores, falha no servidor SSH ou falha na máquina SW.
- **Mudanças na aplicação *MyTelnetCmdClient*:** poderão estar relacionadas com mudanças no nome da aplicação *MyTelnetCmdClient*, ou no seu directório (mudança de nome, mudança de directório ou remoção do mesmo), ou remoção da aplicação do SO da máquina SW.
- **Falha na ligação *telnet*:** ocorre apenas na ligação *telnet* entre a máquina SW e o dispositivo MW, durante a execução da aplicação *MyTelnetCmdClient*. Pode estar relacionada com causas semelhantes às que provocam falha na ligação SSH, ou seja, falha de comunicação na rede de computadores, ou se o dispositivo MW estiver a ser acedido remotamente por outro utilizador, no momento que a aplicação *MyTelnetCmdClient* vai executar.
- **Sobrecarga do MR:** esta falha não se relaciona com a comunicação entre o MV e o MR, mas é considerada falha no MV, porque o MR, nesta situação, retorna para a aplicação *MyTelnetCmdClient* valores de medição extremamente elevados, para caberem no visor do MV.

Quando uma destas falhas ocorre, é colocada a indicação *Push Power* no visor do MV, de acordo com a Figura 42. Esta indicação serve para informar o utilizador que deve carregar no botão *Power*, para o MV recuperar da falha. Em caso de falha, o botão *Power* é o único disponível para ser carregado, devido a que numa falha o MV é forçado a desligar-se. Se a falha persistir, a indicação *Push Power* permanece no visor do MV independentemente do número de vezes que se carregue no botão *Power*.

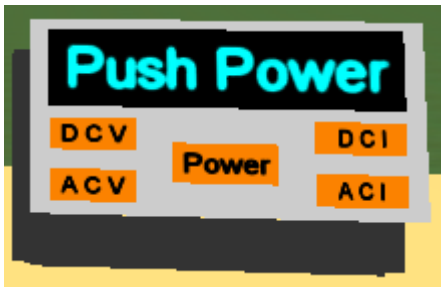


Figura 42 - Multímetro Virtual com mensagem de falha

A indicação do MV na figura anterior é colocada por código dentro de um bloco *try catch*[68], que envolve a invocação da função *getCommand* (da classe *MultimetroCell*) no código de todos os botões, para indicar/recuperar da falha. Pode ter origem nas funções da classe *MySSHClient*, ou nas da aplicação *MyTelnetCmdClient*, a serem tratadas na função *getCommand*.

O código de tratamento de falha consiste em colocar a variável *estado* a 0 (zero), forçando o MV a desligar-se. A indicação *Push Power* é colocada na variável *display*, que ao ser usada como parâmetro nas funções *sendDraw* e *updateShape*, vai colocar a indicação no visor do MV.

A variável booleana *operate* na classe *MultimetroCell*, serve para indicar o estado da falha. Caso seja *true* o MV está a funcionar normalmente, caso esteja a *false* o dispositivo referido está em estado de falha. A variável *operate* também é usada como parâmetro nas funções *sendDraw* e *updateShape*, nesta última função essa variável serve para definir a cor do texto dos botões para o MV, de acordo com o seu estado (ligado/desligado/falha).

Para o MV recuperar de uma falha, existe código no botão *Power* que executa quando a variável *operate* é igual a *false*, colocando-a a *true*. Assim, o MV recupera da falha e continua a funcionar normalmente. Se o MV falhar novamente, o utilizador deve tentar recuperar, carregando no botão *Power*.

5.5 Acessos concorrentes

No módulo do MV, o mecanismo da troca de mensagens de sincronismo, explicado na subsecção 5.3.3, permite a utilização partilhada do dispositivo, por todos os utilizadores ligados ao mundo virtual. Isto significa que é possível um utilizador ligar o MV e logo a seguir outro o desligar, ou um utilizador efectuar uma medição de tensão contínua e depois outro efectuar uma medição de tensão alternada. Resumindo, as acções de um utilizador sobre o MV são visíveis pelos restantes ligados ao mundo virtual.

Este tipo de utilização partilhada pode criar desordem entre os utilizadores do mundo virtual, pois ao pretenderem usar o dispositivo, pode não haver tempo suficiente para os resultados das medições serem observados. Por exemplo, se um utilizador necessita de efectuar uma série de medições de tensão, para estudar o comportamento de um circuito eléctrico, e se durante esse trabalho outro utilizador carrega no botão *Power*, desligando o MV, estraga o trabalho do utilizador que estava a efectuar as medições de tensão, o que cria situações desagradáveis.

[68] Mecanismo de tratamento de excepções/falhas da linguagem Java. Para tratar uma excepção, uma função tem de activá-la quando ocorre uma falha, onde depois será tratada num bloco de código *try catch*, que envolverá a invocação da função, para indicar/recuperar da falha.

Para evitar este tipo de situações criou-se um mecanismo de acessos concorrentes (este mecanismo atribui a utilização do MV, exclusivamente ao utilizador que o ligou em primeiro lugar, bloqueando essa utilização aos restantes utilizadores), resumido pelo fluxograma da Figura 43.

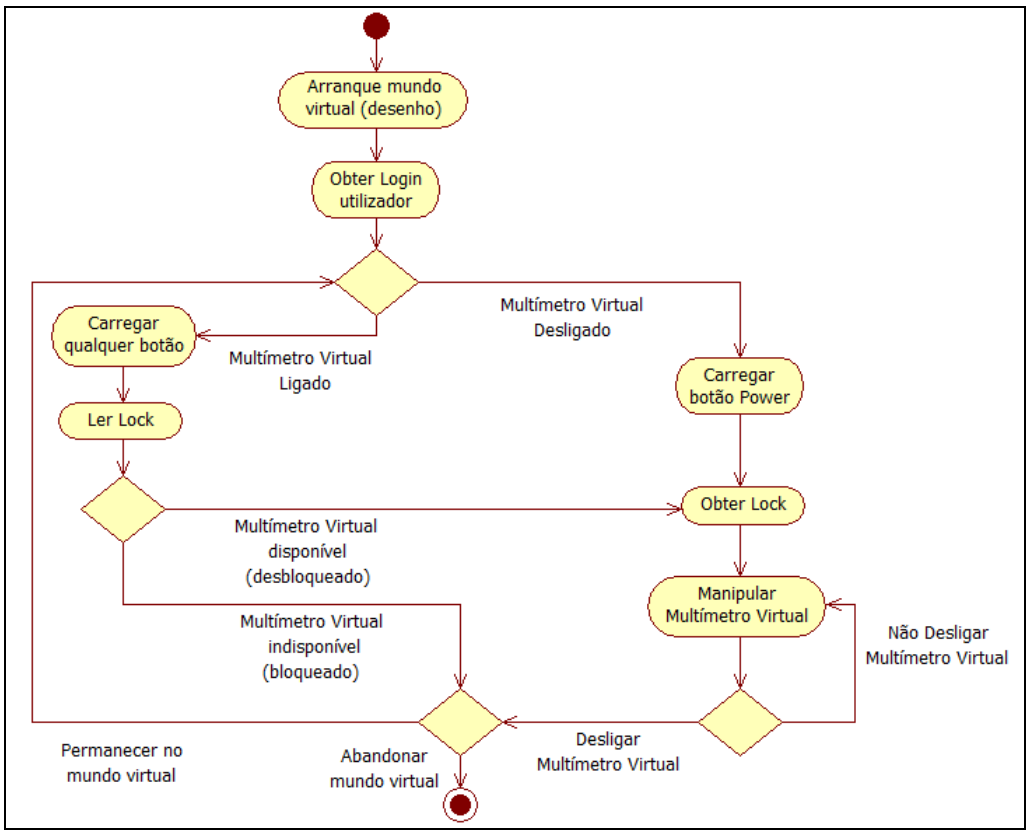


Figura 43 - Funcionamento do mecanismo de acessos concorrentes

O funcionamento deste mecanismo consiste numa variável denominada *lock* do tipo *string*, comum a todo o módulo do MV, que armazena o *login* ou identificação do utilizador que adquiriu o controlo do MV. Em programação de acessos concorrentes, principalmente em sistemas operativos, é comum usar o termo *lock*, ou seja, quando um utilizador ou processo obtém o controlo de um recurso partilhado em situações, que esse recurso só pode/deve ser usado por um utilizador ou processo de cada vez. Assim, pode-se afirmar que quando um utilizador obtém o controlo do MV, obtém o seu *lock*. Quando a variável *lock* é preenchida, deve existir o cuidado do seu acesso ser atómico, ou seja, a operação de preenchimento da variável com o login do utilizador, ocorre na totalidade ou não ocorre. O acesso atómico à variável *lock* é conseguido através do uso de semáforos[69], evitando assim, que o preenchimento dessa variável falhe a meio da operação, o que poderia originar o bloqueio do MV a todos os utilizadores, devido à variável *lock* não ficar devidamente preenchida.

[69] É um tipo de estrutura de dados que protege um bloco de código, denominado secção crítica, de ser acedido por múltiplos processos ao mesmo tempo. Com semáforos, a secção crítica é acedida apenas por um processo de cada vez.

Para os outros utilizadores saberem que o MV está bloqueado, o mecanismo de acessos concorrentes aproveita o mecanismo das mensagens de sincronismo, pois a variável *lock* faz parte de um parâmetro do conteúdo da mensagem enviada, para depois colocar o conteúdo dessa variável, na homóloga de cada utilizador, ao receberem a mensagem.

De acordo com o fluxograma anterior, sempre que um utilizador se liga ao mundo virtual em uso no servidor OW, o módulo do MV obtém o *login* do utilizador, imediatamente após o arranque do mundo virtual (desenho do MV). O *login* é obtido dentro da função *createCellRenderer* da classe *MultimetroCell*, através da função *getUid*, que tem código da classe *ClientContext*, proveniente da API do OW.

Na verificação do estado do MV (ligado/desligado), ao carregar num botão, o primeiro utilizador que ligar o MV (carregar no botão *Power*) obtém o controlo do MV, nesse caso a variável *lock* será igual à do *login* desse utilizador. Para os restantes que estejam no mundo virtual, ou que se liguem posteriormente, obtêm a variável *lock*, a partir da sua homóloga guardada no servidor OW (variável com um funcionamento semelhante às *estado* e *display* abordadas na subsecção 5.4.3), ou de uma mensagem enviada pelo utilizador que detém o controlo, quando carrega num botão. Quando estes utilizadores carregam num botão, não acontece qualquer acção, devido ao seu *login* ser diferente da variável *lock*, sendo a condição de bloqueio do MV quando está ligado.

A transferência do controlo do MV, pode acontecer se um utilizador solicitar o controlo do MV ao utilizador que o detém, usando as formas de comunicação entre utilizadores disponibilizadas pelo OW, seja por texto, voz ou vídeo. Para isso, o utilizador que controla o MV deve-o desligar, para libertar o controlo do MV (colocando a variável *lock* vazia), ficando o dispositivo disponível para todos os utilizadores. Assim, o utilizador que solicita o controlo (a variável *lock* passa a ser igual ao *login* desse utilizador) deve ligar o MV para o utilizar, ficando o dispositivo novamente bloqueado para outros utilizadores.

Neste mecanismo, colocou-se o problema do utilizador, que detém o controlo do MV, deixar o mundo virtual sem desligar o dispositivo, nesse caso, a variável *lock* ficaria indefinidamente com o *login* desse utilizador, impossibilitando sua alteração em tempo de execução, causando o bloqueio permanente do MV. Nessa situação, o MV só seria desbloqueado com um *restore* do mundo virtual, ou se o utilizador que detinha o controlo voltasse ao mundo virtual para desligar o dispositivo.

Para evitar essa situação indesejável, sempre que um utilizador carrega num botão com o MV ligado, é enviada uma mensagem, que ao ser tratada no código do servidor do módulo do OW para ser propagada, verifica o *login* do utilizador que a enviou, ou seja, se o *lock* (ou *login*) do utilizador que detém o controlo se encontra no mundo virtual. Se o *login* for igual ao *lock*, significa que o utilizador que detém o controlo do MV carregou num botão, assim, a mensagem é propagada, sendo a acção do

botão e apresentação do valor de medição no MV, visíveis para os restantes utilizadores. Caso o *login* não coincida com o *lock*, a mensagem foi enviada por um utilizador que carregou num botão, estando o MV bloqueado. Nesse caso, verifica-se se o *lock* do utilizador ainda permanece no mundo virtual, com a execução da função *isLoggedIn* (proveniente da API do OW), que retorna *true* ou *false*, consoante um utilizador esteja ou não ligado. Se a função retornar *true*, significa que o utilizador que detém o controlo permanece no mundo virtual, logo a mensagem não é propagada e o MV permanece bloqueado; caso a função retorne *false*, o MV é desbloqueado pelo utilizador que carregou num botão com o dispositivo bloqueado, assim, a mensagem enviada é propagada para os restantes utilizadores, visualizando a acção, e o controlo do MV é atribuído ao último utilizador que carregou num botão.

Para avaliar a eficácia do mecanismo de acessos concorrentes e de outras funcionalidades do MV, há que o submeter a cenários de teste, que prevejam a utilização de diferentes funcionalidades, e eventuais problemas previstos no seu desenvolvimento.

6. Metodologia de teste e resultados obtidos

Os testes efectuados à solução proposta no capítulo 3 verificam a sua utilidade prática. A metodologia de teste consistiu em submeter a solução a diferentes cenários de utilização, analisando posteriormente os resultados obtidos. Numa fase de testes é importante abranger a maior diversidade de cenários possível, para averiguar/prever o funcionamento da solução em situações de utilização real. A sua viabilidade é definida em função dos resultados obtidos em cada cenário de teste. Num grande número de soluções/aplicações informáticas, mais de metade do tempo de desenvolvimento é dedicado a testes. Quando algum dos testes falha, deve-se localizar, diagnosticar, e solucionar o problema que origina a falha. Depois do problema ser corrigido, o teste deve ser novamente efectuado.

Cada cenário de teste descrito nas secções seguintes corresponde a uma situação específica. No primeiro testa-se a utilização de um mundo virtual OW, na aplicação JWS, com apenas um utilizador, variando o computador e a rede onde se encontra. O funcionamento do MV, com destaque para o seu mecanismo de acessos concorrentes, é testado com dois utilizadores, no segundo cenário. No terceiro e último, pretende-se testar o acesso ao mundo virtual, a partir de pontos geograficamente distantes, já que a distância geográfica influencia o tempo de transmissão de dados numa rede de comunicações. Depois da descrição dos cenários de teste, a última secção deste capítulo trata da comparação entre os mesmos e sua possível expansibilidade.

Na tabela seguinte, estão as características e seus critérios de sucesso, que se podem testar em função dos elementos do cenário de teste. A cobertura/abrangência dos testes efectuados será tanto maior, quanto maior for a combinação/quantidade dos elementos do cenário de teste.

Elementos do Cenário de teste	Característica a testar	Critério de sucesso
Computador onde executa a aplicação JWS (*browser* e SO)	Nível de detalhe dos objectos 3D num mundo virtual	Baixo
	Tempo de carregamento de um mundo virtual	$\leq$ 30 segundos
	Tempo de resposta das acções efectuadas num mundo virtual (inclui utilização do MV)	$\leq$ 3 segundos
	Funcionamento do MV	Obtenção do valor de medição, carregando em todos os botões do MV
Número de utilizadores ($\geq$ 2)	Mecanismo de acessos concorrentes do MV	Transferência do controlo do MV, entre os utilizadores
		Bloqueio do MV aos utilizadores que não detenham o controlo
Ponto de rede de onde se acede ao servidor OW (distância geográfica)	Tempo de carregamento de um mundo virtual	$\leq$ 35 segundos
	Tempo de resposta das acções a efectuar sobre o MV	$\leq$ 5 segundos

Tabela 6 - Cobertura/Abrangência dos cenários de teste

6.1 Utilização do mundo virtual

O objectivo deste cenário é validar o funcionamento/utilização da aplicação cliente JWS do OW em diferentes computadores, com diferentes especificações de hardware, diferentes sistemas operativos e em redes de computadores internas e externas ao CIETI-LABORIS. Em aplicações gráficas, o hardware de um computador faz toda a diferença na visualização gráfica e no tempo de resposta das acções do utilizador. O SO pode ser decisivo em função de compatibilidades de software, por exemplo, para qualquer computador executar a aplicação JWS, deve ter a máquina virtual Java instalada no SO. O funcionamento dos ambientes 3D revistos depende de uma rede de comunicação entre computadores, que no caso do OW o desempenho da rede também influencia a inicialização do mundo virtual, apesar de não ser significativo.

Neste cenário, apenas um utilizador acede ao mundo virtual, e o resultado do teste será positivo se a inicialização (carregamento) do mundo virtual durar no máximo 30 segundos, e se o utilizador obtiver a resposta a uma acção sobre de um objecto 3D (que permita interacção) até um máximo de 3 segundos. Neste último critério está incluída a utilização de todas as funcionalidades do MV, para obter o valor de medição. Estipulou-se um tempo de 30 segundos para a inicialização, por se considerar uma duração máxima aceitável, que o utilizador pode aguardar até utilizar o mundo virtual. Em relação ao tempo máximo de resposta de um objecto 3D ter sido estipulado em 3 segundos, é entendido como sendo a duração máxima aceitável, de espera do utilizador, pela resposta à acção. Qualquer tempo de resposta superior compromete a utilização do objecto 3D.

Para este teste foram usados quatro computadores diferentes, cada um com um SO diferente, ou seja, não houve nenhum SO comum aos computadores. Não existiu critério de localização em relação à rede, sendo essa localização aleatória, embora neste grupo de computadores alguns foram testados a partir da rede interna do CIETI-LABORIS, e outros testados a partir de uma rede externa.

Cada computador usado neste cenário caracteriza-se pela marca, modelo, processador e sua velocidade de relógio, memória RAM, modelo e memória da placa gráfica e versão do SO. Esta caracterização tem em conta os requisitos mínimos/recomendados para execução da aplicação cliente do OW. Foi referido no capítulo 2 que essa aplicação requer no mínimo um processador com 1.5 GHz de velocidade de relógio, memória RAM de 1 GB e uma placa gráfica com uma memória mínima de 128 MB. Assim, neste cenário os computadores usados para teste foram os indicados na tabela seguinte:

Marca/Modelo	Processador/Velocidade de relógio (GHz)	RAM (GB)	Placa Gráfica/Memória Gráfica	SO/ Versão
Toshiba Satellite A300-1ID (Computador 1)	Intel Core 2 Duo 2.5	4	ATI Mobility Radeon HD 3650 com 512 MB	Windows Vista Business SP2
Fujitsu Siemens Amilo D 7850 (Computador 2)	Intel Pentium 4 3.0	1	ATI Mobility Radeon 9200 com 64MB	Windows XP SP2
Apple MacBook 1,1 (Computador 3)	Intel Core Duo 2.0	2	Intel GMA 950 com 64MB	Mac OSX 10.6 (Snow Leopard)
Acer Aspire 5740G (Computador 4)	Intel Core i5520M 2.4	4	ATI Radeon HD 5650 com 1GB	Ubuntu 10.04 (Lucid Lynx)

Tabela 7 - Especificações dos computadores usados para teste

Em todos os computadores usou-se o mundo virtual da Figura 26 (secção 4.5). O teste foi positivo, dado que esse mundo virtual não é computacionalmente exigente em termos gráficos, pois o nível de detalhe dos objectos 3D é baixo, mas sem comprometer a sua visualização.

É possível activar na aplicação JWS, uma janela de estado que exibe mensagens informativas ou de erro. A execução do mundo virtual OW nos computadores 1, 2 e 3 apresentou os resultados das janelas da Figura 44.

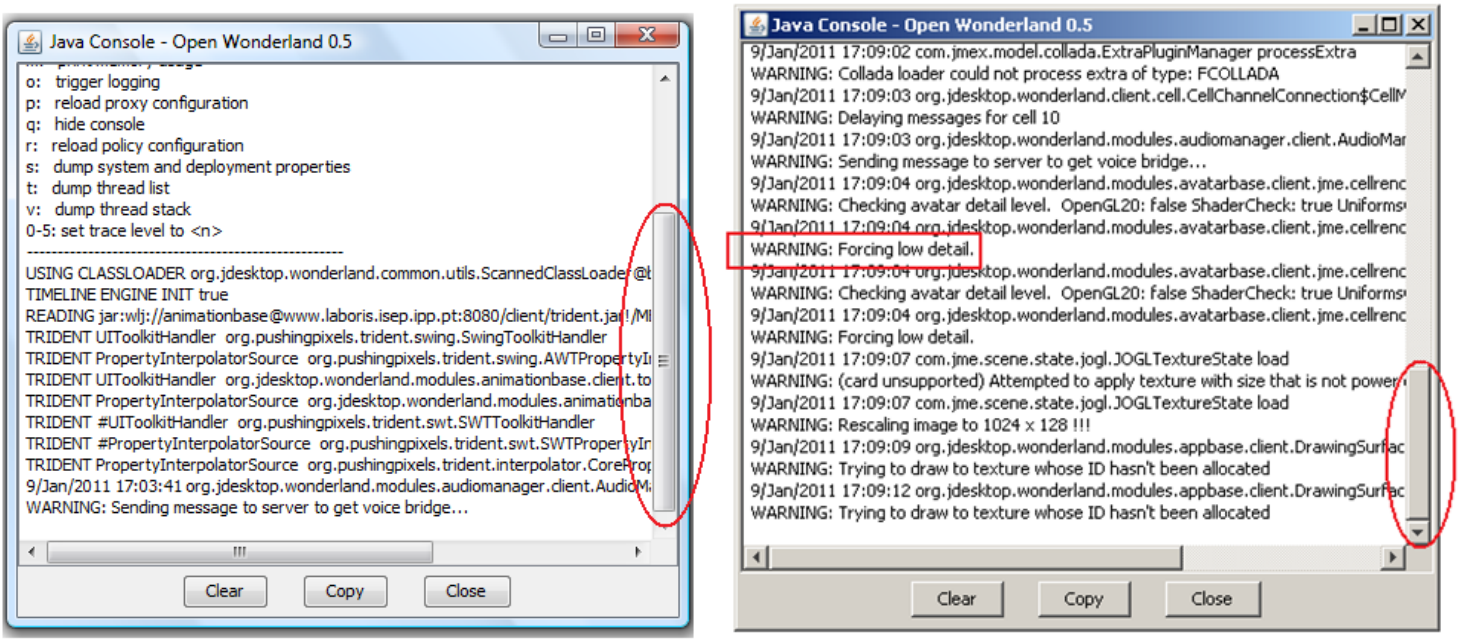


Figura 44 - Janelas de estado da aplicação JWS

Pela observação das janelas da figura anterior verifica-se que a janela esquerda contém menos mensagens que a direita (ver altura da barra de navegação vertical assinalada em ambas as janelas). A janela da esquerda foi o resultado da execução do mundo virtual OW no computador 1 e a da direita correspondeu aos computadores 2 e 3. As mensagens da janela da direita, informam que a aplicação cliente está a forçar os objectos gráficos do mundo virtual, a terem um nível de detalhe o mais baixo

possível (mensagem "*Warning: Forcing low detail*" assinalada com o rectângulo), devido à memória das placas gráficas dos computadores 2 e 3 ser de 64 MB, ou seja, abaixo dos 128 MB, a memória gráfica mínima exigida para executar a aplicação JWS.

Foi também testado nos computadores 1, 2 e 3 o mundo virtual da Figura 30 (subsecção 4.5.1), que é mais completo que o da Figura 26, e com o mesmo nível de detalhe dos objectos 3D. O resultado do teste foi positivo no computador 1, mas negativo nos computadores 2 e 3, por problemas diferentes. Um mundo virtual mais completo é mais exigente em recursos de hardware, daí a razão principal da falha do teste, devido à memória das placas gráficas dos computadores 2 e 3 está abaixo da memória mínima exigida. O problema que levou à falha do teste no computador 2 foi a inicialização do mundo virtual ter sido superior a 30 segundos, mas neste computador ainda foi possível navegar com o *avatar* no mundo virtual, e utilizar o MV com tempos de resposta inferiores a 3 segundos. No computador 3 durante a inicialização do mundo virtual, a aplicação cliente do OW bloqueou em todas as tentativas de execução. A diferença dos resultados negativos dos computadores 2 e 3 é devido aos 64 MB de memória das suas placas gráficas serem acedidos de maneira diferente; no computador 2 a memória da placa gráfica é dedicada[70] e no computador 3 é partilhada[71]. Regra geral em memória de placas gráficas, o desempenho de uma memória partilhada é inferior ao de uma dedicada.

A diferença de memória das placas gráficas também é notória para a configuração do *avatar*, já que nos computadores 2 e 3 a mesma não foi possível.

O uso do computador 4 neste teste teve por objectivo de verificar a execução do OW num SO derivado de Linux. Aqui o teste foi positivo, notou-se apenas que a movimentação do *avatar* é ligeiramente mais lenta em relação ao computador 1, mas sem comprometer a utilização.

Em relação à localização de rede, os computadores 1, 2 e 3 foram testados na rede interna do CIETI-LABORIS e numa externa, sendo obtidos os mesmos resultados independentemente da localização de rede, ou seja, este factor não se repercute na inicialização e utilização da aplicação JWS. A única diferença perceptível é na utilização do MV, que tem tempos de resposta superiores (sem ultrapassar os 3 segundos), quando usado numa rede externa.

Isso justifica-se devido à máquina SW e ao dispositivo MW (Figura 13 da secção 3.3), estarem na rede interna do CIETI-LABORIS. Quando um utilizador acede ao OW a partir desta rede, não existem nós de ligação intermédios na comunicação entre o utilizador e a máquina SW, contrariamente ao acesso feito a partir redes exteriores. Em transmissões de dados, os nós intermédios causam um ligeiro atraso, sendo praticamente imperceptível em condições normais, mas que pode tornar-se significativo quando

[70] Memória dedicada: a memória da placa gráfica está toda na própria placa.
[71] Memória partilhada: a placa gráfica usa a memória proveniente da RAM.

os atrasos de cada nó aumentam e acumulam-se sucessivamente. Depois do mundo virtual ser carregado, o MV funciona em perfeitas condições.

Neste cenário de teste surgiu a questão se o *browser*, de onde a aplicação cliente OW é obtida, interfere com o desempenho desta. Entre todos os computadores foram usados *browsers* diferentes, como o Mozilla Firefox 3.6, Internet Explorer (IE) 7 ou Safari[72] 5, e não se notaram diferenças na execução do mundo virtual de acordo com o *browser*.

A pouca importância dada ao MV, deve-se à utilização intensiva do mesmo não fazer parte dos objectivos deste cenário de teste. O cenário de teste da próxima secção dedica-se exclusivamente ao MV.

6.2 Funcionamento do Multímetro Virtual

Neste cenário de teste valida-se o mecanismo de acessos concorrentes do MV, e a resposta do comando dos botões. Os testes foram efectuados com dois utilizadores, distribuídos pelos computadores 1 e 2 (Tabela 7) por questões de operacionalidade, e efectuados tanto na rede interna do CIETI-LABORIS como numa externa. Os resultados dos testes dependem do sucesso da utilização do MV nas seguintes situações:

a) MV utilizado/bloqueado por um utilizador
b) Abandono[73] do mundo virtual, do utilizador que detinha o controlo do MV
c) Inicialização do mundo virtual com o MV ligado

O teste da situação a), exposto na Figura 45, consiste em dois utilizadores estarem no mundo virtual e alternarem entre si o controlo do MV.

Figura 45 - Teste do mundo virtual com dois utilizadores

[72] Safari: este browser é menos popular que o Mozilla Firefox e Internet Explorer, é desenvolvido pela Apple e está incluído no SO Mac OSX 10.6.
[73] Considera-se abandono do mundo virtual quando um utilizador fecha a aplicação cliente do OW.

No mundo virtual da Figura 45, estão os utilizadores *david* e *oliveira*. Partindo do princípio que é o utilizador *david* que liga primeiro o MV, então fica com o controlo do dispositivo, assim, este utilizador deve carregar em todos os botões do MV sem o desligar e verificar se consegue utilizá-lo, tendo em conta várias distâncias entre o *avatar* e o MV quando carrega num dos seus botões. O utilizador *oliveira* deve tentar, próximo do MV, carregar nos seus botões para verificar se este se encontra bloqueado. Depois, este utilizador deve solicitar ao *david* o controlo do MV através da janela de *chat* (canto inferior direito da Figura 45). Para o mudar o controlo, o utilizador *david* deve desligar o MV e o *oliveira* deve ligá-lo de seguida, invertendo-se as tarefas de teste anteriormente descritas para estes utilizadores. Neste caso, o teste da situação a) será positivo quando:

- O utilizador que detém o controlo conseguir carregar em todos os botões do MV.
- Ambos os utilizadores conseguirem visualizar as acções de utilização sobre o MV.
- For impossível carregar nos botões do MV, para além da distância estipulada (abordada na subsecção 5.3.2) entre o dispositivo e o *avatar*.
- O utilizador que não detém o controlo, não conseguir carregar em nenhum dos botões do MV, quando este está a ser utilizado.
- O MV for desligado e o próximo utilizador que o ligar ficar com o controlo.

Depois de vários ciclos de depuração na programação do mecanismo de acessos concorrentes, foi possível testar com sucesso a situação a) de acordo com a verificação das premissas anteriores.

O teste da situação b) faz-se a seguir ao da situação a), com o objectivo de verificar se o MV fica permanentemente bloqueado. No seguimento do exemplo da Figura 45, partindo do princípio que o utilizador *oliveira* detém o controlo, este deve abandonar o mundo virtual sem desligar o MV. Este teste será positivo se, depois, o *david* conseguir obter o controlo do MV carregando em qualquer um dos seus botões, o que acabou por verificar-se. A situação b) foi uma das principais preocupações na programação do mecanismo de acessos concorrentes, para evitar que o MV ficasse permanentemente bloqueado, impossibilitando a sua utilização a outros utilizadores.

A manutenção do estado do MV é testado na situação c), que implica que um dos utilizadores torne a aceder novamente ao mundo virtual. No seguimento do teste anterior, o utilizador *david* deve abandonar o mundo virtual deixando o MV ligado. Estando o mundo virtual sem utilizadores, o *oliveira* acederá novamente. Este teste será positivo se este utilizador visualizar o MV ligado, com o valor da última medição efectuada no visor, conseguindo depois utilizar o dispositivo. O resultado deste teste foi positivo, mas teria sido negativo se não tivesse sido prevista a situação de teste b), pois na situação c) o mundo virtual foi abandonado, na altura que o utilizador *david* detinha o controlo, e o o*liveira* poderia ter encontrado o MV bloqueado devido ao controlo não ser transferível, quando carregou no botão, tendo em conta o teste da situação c). Basicamente o MV está sempre utilizável

pelo primeiro utilizador que acede ao mundo virtual, independentemente do estado em que foi deixado.

O mecanismo de acessos concorrentes do MV foi concebido para o caso de ser utilizado por dois ou mais utilizadores, por isso, teoricamente, num cenário de teste com mais de dois utilizadores em simultâneo, os resultados seriam iguais aos obtidos neste cenário.

Note-se a diferença dos *avatares* dos utilizadores *david* e *oliveira* na Figura 45, verificando-se que o *avatar* do *david* está mais detalhado. Este nível de detalhe superior está relacionado com o *david* ter usado o computador 1, assim, foi possível configurar o seu *avatar* devido à memória da placa gráfica. Como o utilizador *oliveira* usou o computador 2, mais limitado na placa gráfica, foi-lhe impossível configurar o *avatar*. A aplicação cliente JWS memoriza os *avatares* configurados, assim, experimentou-se os utilizadores trocarem de computador. Nessa situação, o *avatar* do utilizador *david* no computador 2 ficou com o mesmo aspecto do *oliveira*, que se manteve no computador 1, mas com a possibilidade de ser configurado.

Todos os testes deste cenário foram feitos dentro e fora da rede do CIETI-LABORIS, tendo-se verificado um tempo de resposta do MV ligeiramente mais longo (abaixo dos 3 segundos), nos testes efectuados fora daquela rede, pelas razões referidas na secção anterior. Esse tempo de resposta pode ser agravado com o aumento da distância geográfica, que é o cenário testado na próxima secção.

6.3 Influência da distância geográfica no acesso ao mundo virtual

Os testes efectuados neste cenário são semelhantes aos da secção 6.1, embora neste cenário interesse verificar se a aplicação JWS é acessível e utilizável a partir de um ponto geograficamente distante, independentemente da especificação do computador, do *browser* usado para obter a aplicação, ou do número de utilizadores que acedem ao mundo virtual. Para isso, foi solicitada a colaboração de investigadores brasileiros, que acederam ao mundo virtual a partir do Brasil. Os critérios de sucesso neste cenário de teste são: a inicialização do mundo virtual num tempo máximo de 35 segundos, e a utilização do MV com tempos de resposta sem ultrapassar os 5 segundos (ver Tabela 6). Estes tempos têm em consideração uma duração máxima aceitável, que um utilizador pode aguardar ao inicializar/carregar um mundo virtual, e ao obter resposta de uma acção efectuada sobre os objectos 3D (com interacção) na sua utilização, de acordo com as condições deste cenário. No OW, os investigadores representam-se pelos utilizadores *juarez* e *roderval*.

O utilizador *juarez* fez dois acessos ao OW, sendo que no primeiro acesso a inicialização do mundo virtual demorou mais de 35 segundos a carregar, porque numa primeira execução da aplicação cliente, todos os objectos 3D têm de ser descarregados do servidor, sendo uma operação computacionalmente pesada e altamente dependente do desempenho da ligação Internet. Já no segundo acesso, o tempo de

carregamento foi mais rápido (inferior a 35 segundos) devido a que alguns dos objectos 3D são guardados localmente no computador do utilizador para uma próxima inicialização mais rápida, porque dispensa o *download* dos objectos. Este utilizador não detectou nenhuma diferença entre obter a aplicação cliente a partir do browser IE ou Firefox, e conseguiu depois efectuar medições remotas através do MV, com tempos de resposta considerados aceitáveis.

O utilizador *roderval* tentou descarregar a aplicação cliente usando os *browsers* IE e Firefox, mas em ambos não conseguiu executá-la devido a erros de assinatura digital[74] no SO. Este é um exemplo de como o estado do SO, influencia a utilização da aplicação JWS.

6.4 Breves considerações

Os tempos máximos estabelecidos, de inicialização/carregamento do mundo virtual e de resposta ao utilizar os seus objectos 3D, foram estabelecidos em 35 e 5 segundos, respectivamente, como critérios de sucesso do cenário de teste da secção 6.3. São superiores em relação aos tempos máximos estipulados para as mesmas condições, testadas na secção 6.1 (30 e 3 segundos), devido aos utilizadores no cenário 6.3 acederem ao mundo virtual a uma grande distância geográfica da máquina SW, pois há que incluir o atraso causado pelos sucessivos nós de ligação nas comunicações em longas distâncias.

No cenário da secção 6.1, não se notaram diferenças significativas na inicialização do mundo virtual independentemente de ser ou não a primeira vez que era executado, mas essa inicialização foi ligeiramente mais lenta no caso de ser uma primeira execução. Esse pormenor, para o utilizador *juarez*, pode ter sido agravado pela distância geográfica ou ligação Internet.

O grau de cobertura ou abrangência dos testes poderia ser superior, caso se dedicasse mais tempo a esta fase, testando um maior número de combinações de cenários de teste. Outro factor que poderia ter aumentado o grau de cobertura, seria a utilização de software para automatização de testes, o que não se verificou.

Depois da solução final ser submetida a testes, verificando a eficácia do trabalho desenvolvido, é feita no próximo capítulo uma apreciação global da solução final, e sua possível expansibilidade.

[74] A assinatura digital de software é uma técnica que permite identificar a aplicação, usando criptografia. Um erro de assinatura digital pode acontecer quando o SO não é capaz de descodificar a identificação de uma aplicação descarregada da Internet.

7. Conclusão e perspectivas futuras

Pode parecer pouco haver todo este trabalho para uma solução cujo objectivo é o controlo remoto de apenas um multímetro, através de um ambiente imersivo/3D. Todo o trabalho desta tese exigiu uma pesquisa muito abrangente, sendo que antes do trabalho começar não havia qualquer tipo de conhecimento sobre ambientes 3D, por parte do autor. O máximo que sabia sobre a matéria, era da existência do SL. Assim o trabalho arrancou com a revisão dos ambientes 3D, onde se seleccionou o OW. Depois, a pesquisa recaiu sobre como construir conteúdos adicionais e durante a mesma descobriu-se na página *Web* do OW, que era através da programação/construção de módulos. A partir daí ficou claro como se desenhava um objecto 3D, e o trabalho continuou com a pesquisa de como instalar, configurar e administrar o servidor OW, cuja implementação não causou grandes dificuldades. A primeira grande dificuldade surgiu quando se descobriu que não era possível utilizar objectos GUI num módulo OW, apesar de existirem bibliotecas Java gratuitas, construídas para suportar este tipo de objectos em programação gráfica 3D. Esta dificuldade foi contornada usando objectos, texto e funções próprias do *jme*, sendo uma solução que acabou por ser mais trabalhosa. Quando o MV já estava com o aspecto apresentado nesta tese, ou seja com o desenho concluído, com os botões a funcionar e com a possibilidade de surgir texto no visor, construíram-se diferentes mundos ou cenários virtuais que simulassem minimamente o aspecto de um laboratório, uma tarefa relativamente simples, mas algo morosa.

Com o trabalho gráfico dado como concluído, passou-se à fase de programar a comunicação com o MR. Sabia-se ainda pela revisão dos ambientes 3D, que era possível construir aplicações Java que usassem os protocolos de rede *telnet* ou SSH. Assim, a prioridade foi dada à pesquisa de programar para o protocolo *telnet*, já que era o único protocolo de comunicação suportado pelo MR. Aqui surgiu a maior dificuldade de todo o trabalho, pois o número de bibliotecas para programação *telnet* era escasso e a linha de comandos do MR era pouco amigável, pois não tinha *prompt*[75], que era a base de funcionamento da leitura dos valores de medição, retornados do MR para o MV. Para agravar as dificuldades, o MR tinha de estar também acessível de uma rede exterior ao CIETI-LABORIS, pois todo o código proveniente da biblioteca *Apache Commons Net* executava do lado do cliente. Foi tentado que fosse executado do lado do servidor, mas sem sucesso. A ideia para executar do lado do servidor era para somente a máquina SW ligar ao dispositivo MW. O mecanismo de comunicação que usava exclusivamente *telnet*, foi testado durante cerca de três dias, mas revelou-se lento, pouco fiável, e criou uma situação indesejável em termos de segurança, ao ter o MW acessível do exterior através de um protocolo inseguro, pelo que o mecanismo referido foi abandonado. Daí equacionou-se o uso do protocolo SSH para comunicar apenas com a máquina SW. Um novo mecanismo de comunicação foi rapidamente implementado, por existirem inúmeras bibliotecas Java para programar com o protocolo

[75] *Prompt*: é o texto indicativo à esquerda do cursor de uma linha comandos de um SO antes de escrever um comando.

SSH, ao contrário do protocolo *telnet*, e também pela facilidade de programação oferecida pelas funções da biblioteca *Ganymed SSH-2*. Este novo mecanismo obrigava a usar um comando para comunicar com o MR e obter o valor de medição, daí construiu-se a aplicação *MyTelnetCmdClient* para funcionar na linha de comandos, aproveitando o código do anterior mecanismo de comunicação abandonado. Houve grandes melhoramentos neste novo mecanismo, principalmente de fiabilidade.

Já com o MV a efectuar medições remotas no MR, implementou-se o mecanismo de acessos concorrentes, cuja ideia em teoria, parecia difícil, mas com o aproveitamento do código das mensagens de sincronismo, e os exemplos existentes na página *Web* do OW, acabaram por auxiliar a programação desse mecanismo.

Toda a pesquisa deste trabalho acabou por ser bastante abrangente, pois levou ao estudo de domínios completamente diferentes, desde a configuração do ambiente 3D, até programação com protocolos de rede, passando pela programação de objectos 3D e construção de mundos virtuais. Isto tudo para ter um MV com apenas cinco botões, capaz de medir apenas quatro variáveis eléctricas diferentes. Teoricamente, o MV poderia ser enriquecido com mais botões, para mais medições, mas nesse caso apenas mudaria o comando a executar no MR. Poderia ainda, haver a possibilidade de seleccionar o alcance máximo do valor das medições, o que permitiria uma medição de valores muito mais abrangente. Porém, o tratamento dos valores obtidos, a serem apresentados no MV, teria de ser numericamente mais rigoroso. Em relação a aspectos gráficos, seria bem conseguido que existisse a acção do *avatar* carregar num botão do MV. Na aparência do cenário virtual poderia existir um laboratório com mais divisões e bancadas, e principalmente, estar desenhado o circuito eléctrico ligado ao MR para medições, tudo isto para uma aparência mais próxima de um laboratório real. Este trabalho seria ainda mais próximo de um laboratório virtual, caso existissem mais dispositivos virtuais com acesso remoto aos correspondentes reais. Embora não façam parte deste trabalho, a metodologia base para a sua construção já está lançada, dado que para a construção de outros dispositivos, basta desenhá-los no mundo virtual, apenas com formas ou dimensões diferentes do MV, e estudar como esses dispositivos são acedidos remotamente, tentando implementar o seu mecanismo de acesso remoto através da linguagem Java, ou seja, basicamente a metodologia seguida para construção do MV.

Note-se que a utilização de um dispositivo num laboratório remoto, está sempre dependente do estado em que os dispositivos e os circuitos estão montados. Não será possível efectuar uma medição correcta com o MV, se o MR não estiver fisicamente ligado ao circuito, e o mesmo se aplica a outros potenciais dispositivos virtuais.

Muitas outras funcionalidades a incrementar ou a melhorar poderiam ser abordadas, mas foi imperativo construir uma solução minimamente utilizável, e com resultados verificáveis em tempo útil, daí o título da dissertação ser "Acesso a Laboratórios Remotos via Ambientes Imersivos".

Os aspectos adicionais propostos neste capítulo serviriam para retirar a palavra "acesso" do título da tese, pois existe um grande potencial de expansão desta solução, devido à facilidade de construção de mundos virtuais no OW, e pela metodologia de construção de dispositivos virtuais adoptada. O grande desafio que se coloca ao OW é melhorar a sua fiabilidade, por vezes sofrível, por ser um ambiente imersivo relativamente recente, mas isso será certamente melhorado, já que esta tecnologia está em constante desenvolvimento.

Referências

ACTIVE WORLDS. Adabeltriellis's Blog [*online*]. Outubro, 2009. Disponível em: http://adabeltriellis.wordpress.com/2009/10/01/active-worlds/. Consultado em: 2009-11-12.

ACTIVEWORLDS COMMUNITY WEBSITES. Activeworlds, Inc. [*online*]. Disponível em: http://www.awcommunity.org/. Consultado em: 2009-10-24.

ACTUALITY SYSTEMS, INC. [*online*]. Disponível em: http://actuality-medical.com/Home.html. Consultado em: 2009-10-15.

AMAZON SIMPLE STORAGE SERVICE (AMAZON S3). Amazon Web Services [*online*]. Disponível em: http://aws.amazon.com/s3/. Consultado em: 2009-11-07.

APACHE ANT. The Apache Ant Project [*online*]. Disponível em: http://ant.apache.org/. Consultado em: 2010-03-04.

APACHE COMMONS NET. Apache Commons [*online*]. Disponível em: http://commons.apache.org/net/. Consultado em: 2010-09-09.

APACHE SUBVERSION FEATURES. Subversion [*online*]. Disponível em: http://subversion.apache.org/features.html. Consultado em: 2010-03-04.

ASSINATURAS DIGITAIS [*online*]. Junho, 2007. Disponível em: http://assinaturas-digitais.web.simplesnet.pt/. Consultado em: 2011-01-10.

BECOME A VIRTUAL WORLD CITIZEN! Activeworlds, Inc. [*online*]. Disponível em: http://www.activeworlds.com/products/citizenships.asp. Consultado em: 2009-10-24.

BILLBOARD AWT LABEL. jMonkeyengine.org [*online*]. Setembro, 2008. Disponível em: http://jmonkeyengine.org/wiki/doku.php?id=billboard_awt_label&s. Consultado em: 2010-05-25.

BINARY DOWNLOAD. Open Wonderland [*online*]. Disponível em: http://openwonderland.org/download/binary. Consultado em: 2010-03-04.

BUSINESS CENTER. IBM. [*offline*]. Disponível em: http://www.ibm.com/3dworlds/businesscenter/us/en/. Consultado em: 2009-10-17.

CLASS CLIENTCONTEXT. Project Wonderland API [*online*]. Disponível em: http://download.java.net/lg3d/wonderland/api-update/api/org/jdesktop/wonderland/client/ClientContext.html. Consultado em: 2010-10-11.

CLASS CLIENTCONTEXTJME. Project Wonderland API [*online*]. Disponível em: http://download.java.net/lg3d/wonderland/api-update/all/org/jdesktop/wonderland/client/jme/ClientContextJME.html. Consultado em: 2010-06-14.

CLASS COLOR. Java 2 Platform, Standard Edition, v 1.4.2 API Specification [*online*]. Disponível em: http://download.oracle.com/javase/1.4.2/docs/api/java/awt/Color.html. Consultado em: 2010-05-25.

CLASS COLORRGBA. jME API [*online*]. Disponível em: http://www.jmonkeyengine.com/doc/com/jme/renderer/ColorRGBA.html. Consultado em: 2010-04-03.

CLASS DECIMALFORMAT. Java 2 Platform, Standard Edition, v 1.4.2 API Specification [*online*]. Disponível em: http://download.oracle.com/javase/1.4.2/docs/api/java/text/DecimalFormat.html. Consultado em: 2010-09-13.

CLASS FONT. Java 2 Platform, Standard Edition, v 1.4.2 API Specification [*online*]. Disponível em: http://download.oracle.com/javase/1.4.2/docs/api/java/awt/Font.html. Consultado em: 2010-05-25.

CLASS NODE. jME API [*online*]. Disponível em: http://www.jmonkeyengine.com/doc/com/jme/scene/Node.html. Consultado em: 2010-04-03.

CLASS NUMBERFORMAT. Java 2 Platform, Standard Edition, v 1.4.2 API Specification [*online*]. Disponível em: http://download.oracle.com/javase/1.4.2/docs/api/java/text/NumberFormat.html. Consultado em: 2010-09-13.

CLASS TELNETCLIENT. Commons Net 2.1-SNAPSHOT API [*online*]. Disponível em: http://commons.apache.org/net/apidocs/index.html. Consultado em: 2010-09-09.

CLASS USERMANAGER. Project Wonderland API [*online*]. Disponível em: http://download.java.net/lg3d/wonderland/api-update/api/org/jdesktop/wonderland/server/UserManager.html. Consultado em: 2010-10-11.

CLASS VECTOR3F. jME API [*online*]. Disponível em: http://www.jmonkeyengine.com/doc/com/jme/math/Vector3f.html. Consultado em: 2010-04-03.

CLASS WONDERLANDCONTEXT. Project Wonderland API [*online*]. Disponível em: http://download.java.net/lg3d/wonderland/api-update/api/index.html?org/jdesktop/wonderland/server/WonderlandContext.html. Consultado em: 2010-10-11.

COMMUNITY DOWNLOADS. VastPark [*online*]. Disponível em: http://www.vastpark.com/resources/downloads.html. Consultado em: 2011-01-28.

CONFIGURATION. OpenSim [*online*]. Disponível em: http://opensimulator.org/wiki/Configuration. Consultado em: 2009-10-21.

CURRENCY EXCHANGE: BUY OR SELL LINDEN DOLLARS (L$), THE CURRENCY OF SECOND LIFE. Second Life [*online*]. Disponível em: http://secondlife.com/whatis/currency.php. Consultado em: 2009-10-18.

DOWNLOADS. Activeworlds, Inc. [*online*]. Disponível em: http://www.activeworlds.com/products/download.asp. Consultado em: 2009-10-24.

ENTERPRISE EDITION. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/3DX_enterprise_edition.html. Consultado em: 2009-11-08.

ENTERPRISE VIRTUAL WORLDS. VastPark [*online*]. Disponível em: http://www.vastpark.com/solutions/enterprise-virtual-worlds.html. Consultado em: 2011-01-25.

FENGGUI. Java GUIs with OpenGL [*online*]. Disponível em: https://fenggui.dev.java.net/. Consultado em: 2010-05-07.

FLUKE 8845A/8846A DIGITAL MULTIMETER PROGRAMMERS MANUAL. Fluke [*online*]. Disponível em: http://assets.fluke.com/manuals/8845a___pmeng0200.pdf. Consultado em: 2010-09-03.

FLUKE 8845A/8846A DIGITAL MULTIMETER USERS MANUAL. Fluke [*online*]. Disponível em: http://assets.fluke.com/manuals/884xa___umeng0200.pdf. Consultado em: 2010-09-03.

FORCELLA, T. BES NO SECOND LIFE! UMA APOSTA GANHA? Blog do TP [*online*]. Setembro, 2007. Disponível em: http://tpg.blogs.sapo.pt/1341.html. Consultado em: 2009-10-17.

FORMAT A SCIENTIFIC NOTATION DOUBLE TO A NON SCIENTIFIC NOTATION DOUBLE. Java Ranch [*online*]. Janeiro, 2007. Disponível em: http://www.coderanch.com/t/417237/java/java/format-scientific-notation-double-non. Consultado em: 2010-09-13.

FRANLKIN, C. How 3-D Graphics Work. HowStuffWork [*online*]. Disponível em: http://computer.howstuffworks.com/3dgraphics1.htm. Consultado em: 2009-10-15.

FRIAS, P. Quantos Linden Dólares se compram com 1 Euro?. Discursos do Outro Mundo [*online*]. Julho, 2007. Disponível em: http://discursosdooutromundo.blogspot.com/. Consultado em: 2009-10-18.

GANDINI, C. A LABORATORY OF VIRTUAL VETERINARY. Interview held on the 3D world of Vetunimi [*online*]. Maio, 2001. Disponível em: http://www.learnholistically.it/projects/3dfantasia/1issue/vetunimi_en.htm. Consultado em: 2011-01-25.

GANYMED SSH-2 FOR JAVA. Cleondris [*online*]. Disponível em: http://www.cleondris.ch/opensource/ssh2/. Consultado em: 2010-09-25.

GET AVATAR CURRENT POSITION. Java.net [*online*]. Janeiro, 2010. Disponível em: http://www.java.net/forum/topic/archived-forums/project-looking-glass-3d/wonderland-mpk20-interest/get-avatar-current-p-0. Consultado em: 2010-06-14.

GOWEB3D PROPOSE DES MONDES VIRTUELS ET DES SALONS 3D À LA PORTÉE DE TOUS. Le blog officiel de 3dxplorer [*online*]. Setembro, 2008. Disponível em: http://3dxplorer2.wordpress.com/2008/09/15/goweb3d-propose-des-mondes-virtuels-et-des-salons-3d-a-la-portee-de-tous/. Consultado em: 2009-11-14.

IMML [INTERACTIVE MEDIA MARKUP LANGUAGE]. VastPark developer zone. VastPark [*online*]. Disponível em: http://www.vastpark.org/wiki/vp/IMML. Consultado em: 2011-01-07.

INTRODUCTION TO THE ACTIVE WORLDS SDK. Activeworlds, Inc. [*online*]. Disponível em: http://www.activeworlds.com/sdk/. Consultado em: 2009-10-24.

JAVA – WRITING AN AUTOMATED TELNET CLIENT. twit88.com [*online*]. Dezembro, 2007. Disponível em: http://twit88.com/blog/2007/12/22/java-writing-an-automated-telnet-client/. Consultado em: 2010-09-09.

JOHNSON, D. Wonderland Release 0.5 Input System [*online*]. Outubro, 2009.Disponível em: https://docs.google.com/View?docid=dck87q9p_24d5bnvbft. Consultado em: 2010-04-20.

KAPLAN, J. Private / Public IP address issue - running OWL on Amazon EC2. Open Wonderland Forum [*online*]. Disponível em: http://groups.google.com/group/openwonderland/msg/40aa39dfcefc9b59. Consultado em: 2010-04-01.

LERUSALIMSCHY, R.; FIGUEIREDO, L.; CELES, W. Lua 5.1 Reference Manual [*online*]. Julho, 2010. Disponível em: http://www.lua.org/manual/5.1/manual.html#2.1. Consultado em: 2011-01-28.

LSL PORTAL. Second Life Wiki [*online*]. Disponível em: http://wiki.secondlife.com/wiki/LSL_Portal. Consultado em: 2009-10-17.

MCKAY, P. Network UPS Tools (NUT) on Ubuntu. Keystone IT Tech [*online*]. Setembro, 2006. Disponível em: http://keystoneit.wordpress.com/2006/09/25/network-ups-tools-nut-on-ubuntu/. Consultado em: 2010-11-08.

MEMBERSHIP PLANS. 3DXplorer [*offline*]. Disponível em: http://www.3dxplorer.com/i.php?p=static/2. Consultado em: 2009-11-08.

MIASNIKOVAS, A.SSH with Java. Andrius Miasnikovas's Blog [*online*]. Julho, 2009. Disponível em: http://andrius.miasnikovas.lt/2009/07/ssh-with-java/. Consultado em: 2010-09-25.

NODE. jMonkeyengine.org [*online*]. Disponível em: http://jmonkeyengine.org/wiki/doku.php/nodes. Consultado em: 2010-04-03.

NUT - CHANGING THE LOW BATTERY THRESHOLD WITHOUT FRONTVIEW. NETGEAR ReadyNAS [*online*]. Abril, 2009.Disponível em: http://www.readynas.com/forum/viewtopic.php?f=11&t=28429. Consultado em: 2010-11-10.

OPEN VIRTUAL WORLDS PLATFORM. VastPark [*offline*]. Disponível em: http://www.vastpark.com/documentation/publisher. Consultado em: 2009-11-07.

OPEN WONDERLAND DOCUMENTATION WIKI. Google Code [*online*]. Disponível em: http://code.google.com/p/openwonderland/wiki/OpenWonderland. Consultado em: 2009-10-21.

OPEN WONDERLAND FAQ. Open Wonderland [*online*]. Disponível em: http://openwonderland.org/about/faq. Consultado em: 2009-10-21.

OPEN WONDERLAND PROJECT IDEAS. Google Code [*online*]. Disponível em: http://code.google.com/p/openwonderland/wiki/ProjectIdeas. Consultado em: 2010-02-10.

OPEN WONDERLAND V0.5: DOWNLOAD, BUILD AND DEPLOY MODULES. Google Code [*online*]. Disponível em: http://code.google.com/p/openwonderland/wiki/DownloadBuildModules05. Consultado em: 2010-03-06.

OPEN WONDERLAND V0.5: DOWNLOAD, CONFIGURE, BUILD AND RUN FROM THE SOURCE CODE. Google Code [*online*]. Disponível em: http://code.google.com/p/openwonderland/wiki/DownloadBuildSource05. Consultado em: 2010-03-04.

PERMISSIONS (SERVER). OpenSim [*online*]. Disponível em: http://opensimulator.org/wiki/29. Consultado em: 2009-10-21.

POWERFUL SDK. VastPark [*online*]. Disponível em: http://www.vastpark.com/platform/powerful-sdk.html. Consultado em: 2011-01-28.

PROJECT HISTORY. OpenSim [*online*]. Disponível em: http://opensimulator.org/wiki/History. Consultado em: 2011-02-07.

PROJECT SNOWSTORM. Second Life Wiki [*online*]. Disponível em: http://wiki.secondlife.com/wiki/Project_Snowstorm. Consultado em: 2009-10-18.

PROJECT WONDERLAND AT THE CHEMISTRY DEPARTMENT. UWI-Mona [*offline*]. Disponível em: http://wwwchem.uwimona.edu.jm/WL/. Consultado em: 2009-11-12.

PROJECT WONDERLAND ROADMAP AND RELEASE ESTIMATES. Java.net [*online*]. Fevereiro, 2007. Disponível em: http://wiki.java.net/bin/view/Javadesktop/WonderlandRoadmap. Consultado em: 2009-10-21.

PROJECT WONDERLAND V0.5: CONFIGURING AUTHENTICATION. Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandAuthentication05. Consultado em: 2010-03-06.

PROJECT WONDERLAND V0.5: WEB-BASED SERVER ADMINISTRATION. Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandServerAdministration05. Consultado em: 2010-03-06.

SANTOS, C; PEDRO, L; ALMEIDA, P. Second Life e Educação. Dep. Comunicação e arte – Universidade de Aveiro [*online*]. Disponível em: http://www.slideshare.net/csantos/second-life-e-educao. Consultado em: 2009-10-18.

SECOND LIFE (ANÁLISE). Mega Forum Evolution [*online*]. Maio, 2008. Disponível em: http://mfevolution.informe.com/second-life-annolise-dt430.html. Consultado em: 2009-11-10.

SECOND LIFE COMPETITOR, VASTPARK, RELEASES BETA CREATION TOOLS TO THE WORLD. My Disguises [*online*]. Outubro, 2007. Disponível em: http://mydisguises.com/2007/10/19/second-life-competitor-vastpark-releases-beta-creation-tools-to-the-world/. Consultado em: 2009-11-14.

SECOND LIFE DEVELOP. Second Life Wiki [*online*]. Disponível em: http://secondlifegrid.net/technology-programs/virtual-world-api. Consultado em: 2009-10-18.

SIMMONS, C. Sweden opens virtual embassy 3D-style. SWEDEN.SE [*online*]. Maio, 2007. Disponível em: http://www.sweden.se/eng/Home/Lifestyle/Reading/Second-Life/. Consultado em: 2009-10-17.

SLOTT, J. Project Wonderland v0.5: Capturing mouse input (Part 2). Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandDevelopingNewCell05Part2. Consultado em: 2010-04-20.

SLOTT, J. Project Wonderland v0.5: Creating a simple shape (Part 1). Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandDevelopingNewCell05Part1. Consultado em: 2010-04-02.

SLOTT, J. Project Wonderland v0.5: Synchronizing state across clients (Part 4). Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandDevelopingNewCell05Part4. Consultado em: 2010-05-26.

SLOTT, J. Project Wonderland v0.5: Working with Modules. Java.net [*online*]. Disponível em: http://wiki.java.net/bin/view/Javadesktop/ProjectWonderlandWorkingWithModules05. Consultado em: 2010-04-01.

SLOTT, J.; KAPLAN, J. Open Wonderland v0.5: Launching Clients using Java Web Start. Google Code [*online*]. Disponível em: http://code.google.com/p/openwonderland/wiki/WebStart05. Consultado em: 2010-03-07.

SOLUTIONS DOCUMENTATION. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/documentation.html. Consultado em: 2009-11-08.

SYSTEM REQUIREMENTS. Second Life [*online*]. Disponível em: http://secondlife.com/support/system-requirements/. Consultado em: 2009-10-17.

THE PROJECT FACTORY DEVELOPS IN VASTPARK. The Project Factory [*online*]. Novembro, 2008. Disponível em: http://theprojectfactory.com/the-project-factory-develops-in-vastpark/. Consultado em: 2009-11-07.

THERE - DEVELOPER FAQ. Threre Inc. [*offline*]. Disponível em: http://www.there.com/developerFAQ.html. Consultado em: 2009-10-31.

THERE – DEVELOPERS – SPECIFICATIONS AND PRICING. Threre Inc. [*offline*]. Disponível em: http://webapps.prod.there.com/developer/price.cgi. Consultado em: 2009-10-31.

THERE - GET STARTED. Threre Inc. [*offline*]. Disponível em: http://www.there.com/help.html. Consultado em: 2009-10-31.

THERE HELP. Threre Inc. [*offline*]. Disponível em: http://webapps.prod.there.com/scripts/help.py/helpPortal. Consultado em: 2009-10-31.

THERE. Virtual Worlds Review [*online*]. Disponível em: http://www.virtualworldsreview.com/there/. Consultado em: 2009-10-24.

THERE.COM CLOSED ON MARCH 9TH, 2010. Threre Inc. [*online*]. Disponível em: http://www.prod.there.com/info/announcement. Consultado em: 2010-01-31.

THERE.COM: 3D ONLINE VIRTUAL WORLD. Interactive Rich Media [*online*]. Disponível em: http://www.saraproft.org/?p=72. Consultado em: 2009-11-14.

UNIVERSIDADE DE AVEIRO INAUGURA ILHA NO SECOND LIFE. JornalismoPortoNet [*online*]. Maio, 2007. Disponível em: http://jpn.icicom.up.pt/2007/05/25/universidade_de_aveiro_inaugura_ilha_no_second_life.html. Consultado em: 2009-10-17.

UPS HARDWARE COMPATIBILITY LIST. Network UPS Tools [*online*]. Disponível em: http://www.networkupstools.org/compat/stable.html. Consultado em: 2010-11-08.

VASTPARK PLAYER. VastPark [*online*]. Disponível em: http://www.vastpark.com/platform/player.html. Consultado em: 2011-01-28.

VASTPARK SERVER. VastPark [*online*]. Disponível em: http://www.vastpark.com/platform/server.html. Consultado em: 2011-01-28.

VASTPARK. Crunchbase [*online*]. Disponível em: http://www.crunchbase.com/company/vastpark. Consultado em: 2009-11-07.

VEL 2.0 - VIRTUAL ENTERPRISE LAB. Lucerne University of Applied Sciences and Arts [*online*]. Disponível em: http://virtual.enterpriselab.ch/en/. Consultado em: 2011-01-25.

VISIR@ISEP [*online*]. Setembro, 2010.Disponível em: http://www.isep.ipp.pt/mostra_prox_evento.php?id=804. Consultado em: 2011-01-11.

WELCOME TO THERE!. Threre Inc. [*offline*]. Disponível em: http://www.there.com/pr_acquisition.html. Consultado em: 2009-10-31.

WHAT IS A VIRTUAL WORLD?. Virtual Worlds Review [*online*]. Disponível em: http://www.virtualworldsreview.com/info/whatis.shtml. Consultado em: 2009-10-15.

WHAT IS OPENSIMULATOR?. OpenSim [*online*]. Disponível em: http://opensimulator.org/wiki/Main_Page. Consultado em: 2009-10-21.

WHAT IS 3DXPLORER ?. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/introduction.html. Consultado em: 2011-01-30.

WORLD REVIEW: ACTIVE WORLDS. Virtual World Lets [*offline*]. Disponível em: http://www.virtualworldlets.net/Worlds/Reviews/Reviews.php?ID=5. Consultado em: 2009-10-24.

WORLD SERVERS CONFIGURATION AND PRICING. Activeworlds, Inc. [*online*]. Disponível em: http://www.activeworlds.com/products/worlds_pricing.asp. Consultado em: 2009-10-31.

XENKI. OpenSim [*online*]. Disponível em: http://opensimulator.org/wiki/Xenki. Consultado em: 2011-01-20.

YANKELOVICH, N. Learning the Basics Tutorial. Open Wonderland [*online*]. Disponível em: http://sites.google.com/site/openwonderland/tutorials/learning-the-basics-tutorial. Consultado em: 2010-03-07.

3DXPLORER API. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/3DX_api.html. Consultado em: 2009-11-08.

3DXPLORER MEET-IN-3D. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/3DX_meet_in_3D.html. Consultado em: 2009-11-08.

3DXPLORER ONLINE MEETING. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/3DX_online_meeting.html. Consultado em: 2009-11-08.

3DXPLORER PLATFORM (FREE). 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/platform.html. Consultado em: 2009-11-08.

3DXPLORER VIRTUAL CONFERENCING. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/3DX_conferencing.html. Consultado em: 2009-11-08.

3DXPLORER: PRODUCT COMPARISON & PRICING. 3DXplorer [*online*]. Disponível em: http://www.3dxplorer.com/static-v3/product_comparison_pricing.html. Consultado em: 2009-11-08.

GARCÍA-ZUBIA, J.; M. CASTRO, M.; ORDUÑA, P. et al. SecondLab: A Remote Laboratory under Second Life. IEEE EDUCON 2010.Abril, 2010.

MARCELINO, R.; SILVA, J.B.; ALVES, G.R. et al. An extended immersive learning environment for solid mechanics theory and demonstration(s). REV 2010. Junho, 2010.

NEVES, N. Second Life: De um motor 3D para um mundo social. Tecnologias e Sistemas Multimédia. Mestrado em Engenharia Informática, Sistemas Gráficos e Multimédia. Dezembro, 2008.

SCHEUCHER, T.; BAILEY, P.; GÜLT, C. et al. Collaborative Virtual 3D Environment for Internet-accessible Physics Experiments. REV 2009. Junho, 2009.

Anexos

Anexo 1 - Configuração do servidor Open Wonderland

Depois de compilar o código fonte do servidor OW é necessário configurá-lo, para iniciar o seu funcionamento. A configuração do OW requer instruções ou propriedades, necessárias ao seu arranque/funcionamento. É feita num ficheiro incluído no *download* do código fonte do servidor OW, denominado *my.run.properties.example*, contendo instruções exemplo de configuração. Com base nesse exemplo é boa prática copiar o ficheiro *my.run.properties.example* para *my.run.properties* e alterar as respectivas instruções, exemplificadas no Bloco de configuração 1 de forma ao servidor OW funcionar correctamente.

```
wonderland.webserver.host=192.168.10.11
wonderland.webserver.port=8080
wonderland.web.server.url=http://www.laboris.isep.ipp.pt:8080/
```

Bloco de configuração 1

Com o servidor OW devidamente configurado, procede-se ao seu arranque, que demora cerca de 10 - 15 minutos, dependendo do desempenho do PC onde está a executar. A indicação de arranque concluído corresponde ao surgimento do seguinte *output* de mensagens, na linha de comandos.

```
[java] ------------------------------------------------------------
[java] Wonderland web server started successfully.
[java] Log files are in /home/laboris/.wonderland-server/0.5-preview4/log
[java] Web server running on http://www.laboris.isep.ipp.pt:8080/
[java] ------------------------------------------------------------
```

***Output* 4**

Este *output* indica que o processo do servidor OW está activo e a executar, a partir daqui já é possível utilizar funcionalidades do OW. Para isso basta abrir um *browser* e introduzir o endereço respectivo, com base no seguinte URL:

```
http://<host name>:<port>/wonderland-web-front
```

URL 1

Onde *host name* é o IP ou nome da máquina onde o servidor OW executa, e *port* o seu porto de acesso. Quando o endereço URL 1 é introduzido convenientemente no *browser*, abre a página *Web* da Figura 15 (secção 4.1).

Passa possibilitar o acesso ao servidor OW ou a um mundo virtual, a partir da Internet, é imperativo verificar se a propriedade *wonderland.web.server.url* no ficheiro *my.run.properties* tem o URL correcto, e adicionar a propriedade *darkstar.host.public* ao componente *Darkstar Server*, com o nome da máquina à qual os utilizadores acedem da Internet, de acordo com a Propriedade de configuração 1.

darkstar.host.public	www.laboris.isep.ipp.pt

Propriedade de configuração 1

Anexo 2 - Implementação de autenticação

A autenticação deve ser configurada com o servidor OW completamente desactivado. A sua configuração inicia-se previamente com o *download* de módulos ou componentes de software adicionais, com o comando:

```
svn checkout http://openwonderland-modules.googlecode.com/svn/branches/0.5-preview4 wonderland-modules
```

Comando 3

O seu funcionamento é análogo ao explicado na secção 4.1. O comando deve ser executado no mesmo directório onde se encontra o directório do código fonte do OW. Todo o conteúdo descarregado deve ser guardado dentro do directório *wonderland-modules*, criado pelo comando *Subversion svn*, caso o directório seja inexistente. Esta operação de *download* demora cerca de 10-15 minutos, dependendo das condições de rede, e termina com uma indicação informativa na linha de comandos. Terminado o *download* dos módulos, procede-se à configuração da autenticação propriamente dita. Em primeiro lugar, é necessário aceder ao ficheiro *build.xml* do directório *~/wonderland/modules/tools* e comentar a Instrução 2, para impossibilitar a funcionalidade de acesso aberto, no arranque do servidor.

```
<!--file name="security-session-noauth/build.xml"/-->
```

Instrução 2

Depois, é imperativo indicar ao servidor que quando arrancar, tem de incluir o módulo de autenticação. Para isso, acede-se ao ficheiro *build.xml* localizado no directório *~/wonderland-modules/stable*, e adiciona-se a Instrução 3 na *tag* XML respectiva. Recomenda-se depois, que o ficheiro *security-session-noauth.jar* seja apagado (no directório *~/wonderland/modules/dist*, e sua cópia guardada no directório *~/wonderland/modules/tools/security-session-noauth*), pois está relacionado com acesso aberto a todos os utilizadores.

```
<file name="security-session-auth/dist/security-session-auth.jar"/>
```

Instrução 3

Antes de qualquer configuração, acede-se com as credenciais *admin admin* para *Username* e *Password*, respectivamente, para se aceder à página de gestão de componentes do servidor.

O conjunto de procedimentos finais da configuração da autenticação, baseiam-se em criar um ficheiro de palavra-chave, onde o seu conteúdo é a própria palavra-chave sem qualquer encriptação. O ficheiro é para ser usado como autenticação dos componentes do servidor OW, já que cada componente requer um utilizador associado, que precisa de se autenticar para o componente funcionar. Esse ficheiro deverá ter o nome de *wonderland.password*, e estar guardado no directório *~/.wonderland-server/0.5-dev*, apenas com permissões de leitura para o utilizador dono do SO, como política de segurança, sendo esta reforçada pelo facto do directório *.wonderland-server* ser oculto. Com o ficheiro criado, finaliza-se a configuração da autenticação, adicionando as propriedades requeridas para cada

componente do servidor se autenticar. Assim, adiciona-se a Propriedade de configuração 2 ao ficheiro *my.run.properties*, para o servidor *Web* do OW saber onde está o ficheiro de palavra-chave, para se autenticar.

```
wonderland.webserver.password.file=/home/laboris/.wonderland-server/0.5-preview4/wonderland.password
```

Propriedade de configuração 2

Á semelhança desta propriedade, deve-se fazer o mesmo procedimento, para os componentes *Darkstar Server*, *Shared Application Server* e *Voice Bridge*, nas Propriedades de configuração 3, 4 e 5, respectivamente, que à semelhança do que está exposto na Propriedade de configuração 1, o nome da propriedade fica no campo da esquerda e a sua configuração no campo da direita.

sgs.password.file	/home/laboris/.wonderland-server/0.5-pre

Propriedade de configuração 3

sas.password.file	/home/laboris/.wonderland-server/0.5-pre

Propriedade de configuração 4

voicebridge.password.file	/home/laboris/.wonderland-server/0.5-pre

Propriedade de configuração 5

O valor usado na configuração é igual em todas estas propriedades, e corresponde ao caminho da instrução seguinte.

```
/home/laboris/.wonderland-server/0.5-preview4/wonderland.password
```

Instrução 4

O caminho indica a cada componente do servidor onde se encontra o ficheiro *wonderland.password*, para se poderem autenticar.

Anexo 3 - Gestão de componentes do servidor

Na página de gestão dos componentes do servidor OW (Figura 18, subsecção 4.2.1), para activar um determinado componente deve-se clicar no *link start*, quando o mesmo está no estado de *Not Running*. Em condições normais, o novo estado passará a *Running* e o *link start* será substituído pelo *link stop*, que serve para desactivar o componente, passando este novamente para o estado *Not Running*. O componente *Darkstar Server* ao estar no estado *Running*, tem o *link restart*, que serve para reiniciar o componente numa acção de desactivação, imediatamente seguida de activação, esse *link* só é visível quando um componente está no estado *Running*. A configuração dos componentes é acedida a partir do *link edit*, que dá acesso à página da Figura 46, onde se podem adicionar, modificar e remover propriedades de configuração. A título de exemplo, na figura referida está a configuração do componente *Darkstar Server*, sendo a configuração dos outros componentes semelhante, onde apenas mudam o nome e o valor das propriedades de configuração.

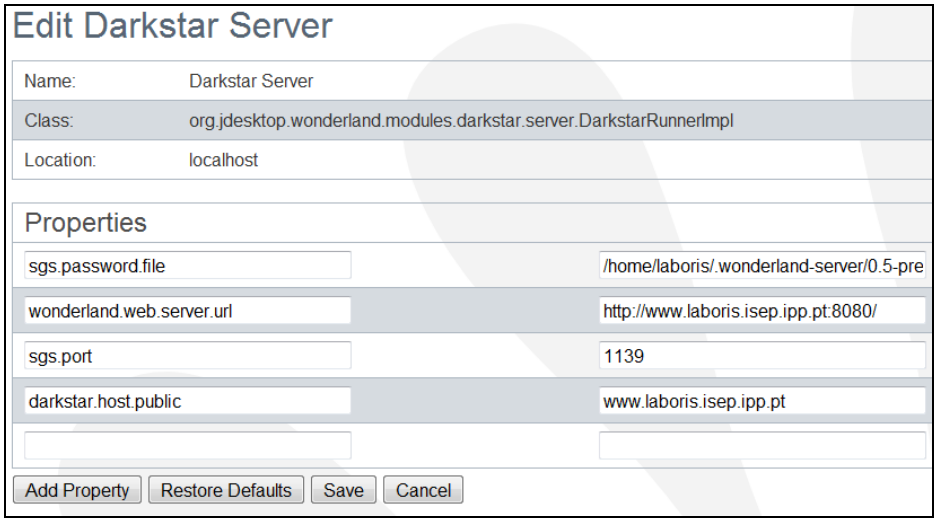


Figura 46 - Propriedades de configuração do *Darkstar Server*

De acordo com a figura anterior, nos campos debaixo da indicação *Properties*, os da esquerda são o nome da propriedade e os da direita o valor da mesma. No caso do *Darkstar Server*, está definido o ficheiro de palavra-chave na propriedade *sgs.password.file*, a propriedade *wonderland.web.server.url* especifica o URL de acesso ao servidor *Web* do OW, o porto de acesso é o valor da propriedade *sgs.port*, e a propriedade *darkstart.host.public* define o endereço *Web* que está disponível para a rede exterior (Internet). À excepção da propriedade *darkstar.host.public*, todas as outras vêm incluídas por omissão na configuração do *Darkstar Server*. Debaixo dos campos de configuração das propriedades encontram-se quatro botões:

- ***Add Property*:** permite adicionar uma propriedade de configuração, ou seja, adiciona uma linha de dois campos, caso os restantes estejam preenchidos.
- ***Restore Defaults*:** restabelece a configuração por omissão incluída no componente.

- ***Save*:** guarda a configuração, e volta à página de gestão dos componentes do servidor OW.
- ***Cancel*:** volta à página de gestão dos componentes sem efectuar alterações.

O *link log* na coluna *Actions* (Figura 18), permite visualizar os *logs* (eventos registados) de cada componente do servidor OW.

Anexo 4 - Gestão de módulos

O acesso à página de gestão de módulos é feito pelo clique no *link Manage Modules*, da lista de *links* do lado esquerdo (Figura 16, secção 4.1).

A instalação ou adição de um módulo ao OW faz-se clicando no botão *Browse* (Figura 25, secção 4.4), que abre uma janela para indicar o caminho no qual se encontra o ficheiro JAR, esse caminho é depois inserido no campo de texto à esquerda do referido botão. Depois de especificado o ficheiro JAR, clica-se no *link Install*, sendo adicionada uma linha à tabela *Installed Modules* com o novo módulo. Se for adicionado um módulo já existente, o anterior é sobreposto pelo último adicionado. A remoção de um módulo faz-se seleccionando o módulo a remover, marcando o pequeno quadrado à esquerda da coluna *Module Name*, na página de gestão de módulos (Figura 25, secção 4.4). Depois, basta navegar pelo *scroll* do *browser* até ao fundo da tabela *Installed Modules*, e clicar no *link Removed selected modules* (Figura 47), por baixo do fundo da tabela referida.

☐	security-session-auth	v0.1	Wonderland Security Authorized Session Store
☐	servermanager	v0.5	Server manager library
☐	sharedstate	v0.5	Shared state API and library
☐	snapshot-manager	v0.5	Snapshot manager library
☐	stickynote	v1.0	[None]
☐	telepointer	v1.0	Tele Pointer
☑	textchat	v0.5	Text chat API and library
☐	viewproperties	v0.5	View properties configuration library
☐	voicebridge	v0.5	jVoicebridge server
☐	webdav	v0.5.1	Webdav library
☐	whiteboard	v1.0	Wonderland Whiteboard Module
☐	xapps-config	v0.5	Shared apps configuration library
☐	xremwin	v0.5	X11 shared application library

Removed selected modules

Figura 47 - Zona inferior da página de gestão de módulos

No caso da Figura 47 o módulo marcado para remoção é o *textchat*. A adição e remoção de módulos do OW, deve ser feita com o componente *Darkstar Server* desactivado.

Um módulo só funcionará correctamente se estiver correctamente configurado. No bloco de configuração seguinte, está um exemplo de configuração do módulo do MV, no seu ficheiro *my.module.properties*.

```
#
# Property: module.name (required)
# The unique name of the module
#
module.name=Multimetro

#
# Property: module.description (optional)
# A textual description of the module
#
module.description=Multimetro controlado remotamente

#
# Property: wonderland.dir (required)
# The location of the Wonderland source
#
wonderland.dir=/home/laboris/david/src/0.5/0.5-preview4/wonderland

#
# Property: module.plugin.src (optional)
# Beneath src/classes/, where is the module code located (common/, client/, server/)
#
module.plugin.src=org/jdesktop/wonderland/modules/multimetro
```

Bloco de configuração 2

A definição da versão de um módulo é exemplificada no Bloco de configuração 3, que contém parte de código XML do ficheiro de compilação *build.xml*, do módulo do MV.

```
        <module name="${module.name}" majorVersion="6" minorVersion="1" jarfile="${module.dist.dir}/${module.name}.jar"
moduleDescription="${module.description}" builddir="${build.dir}">
            <client dir="${current.dir}/lib">
                <include name="*.jar"/>
                <clientjar name="${module.name}-client" basedir="${build.classes.dir}">
                    <include name="${module.src}/client/**"/>
                    <include name="${module.src}/common/**"/>
                    <fileset dir="${current.dir}/src/classes">
                        <include name="${module.src}/client/resources/**"/>
                    </fileset>
                </clientjar>
            </client>

            <server>
                <include name="*.jar"/>
                <serverjar name="${module.name}-server" basedir="${build.classes.dir}">
                    <include name="${module.src}/server/**"/>
                    <include name="${module.src}/common/**"/>
                </serverjar>
            </server>
        </module>
```

Bloco de configuração 3

Anexo 5 - Segurança

No menu de edição de propriedades de objectos 3D (Figura 27, secção 4.5), clicando em *Properties*, surge o menu da Figura 48.

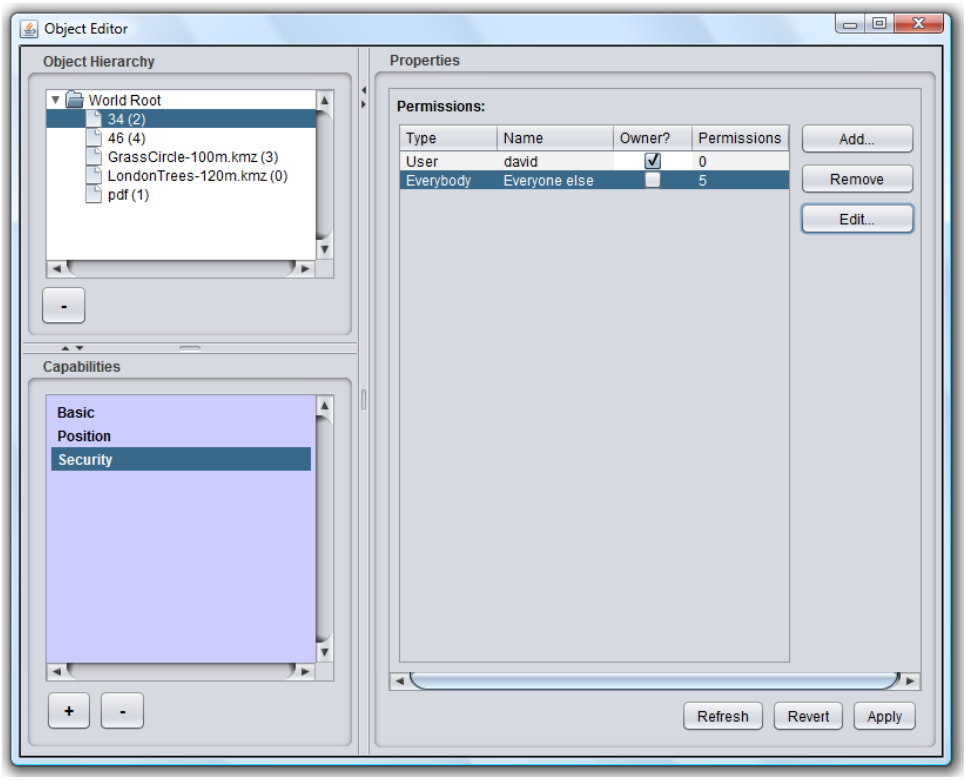


Figura 48 - Menu geral de edição de propriedades de objectos 3D

O menu anterior também pode ser usado para alterar outras propriedades dos objectos 3D, tal como a sua posição. Por exemplo, clicando no objecto que se pretende alterar na caixa à esquerda, por baixo da indicação *Object Hirarchy*, e depois clicando em *Position* também à esquerda, por baixo da indicação *Capabilities*. Esse menu serve também para remover objectos, clicando no botão "-" por baixo do quadro com a listagem dos objectos (*Object Hirarchy*).

A única forma de adicionar permissões a objectos 3D, é através do menu da Figura 48. Estas permissões tratam-se de uma capacidade extra do objecto, adicionada com o botão "+", por baixo da caixa *Capabilities*, situado à esquerda. Depois, surge o menu de adição de capacidades de objectos 3D, exibido na Figura 49.

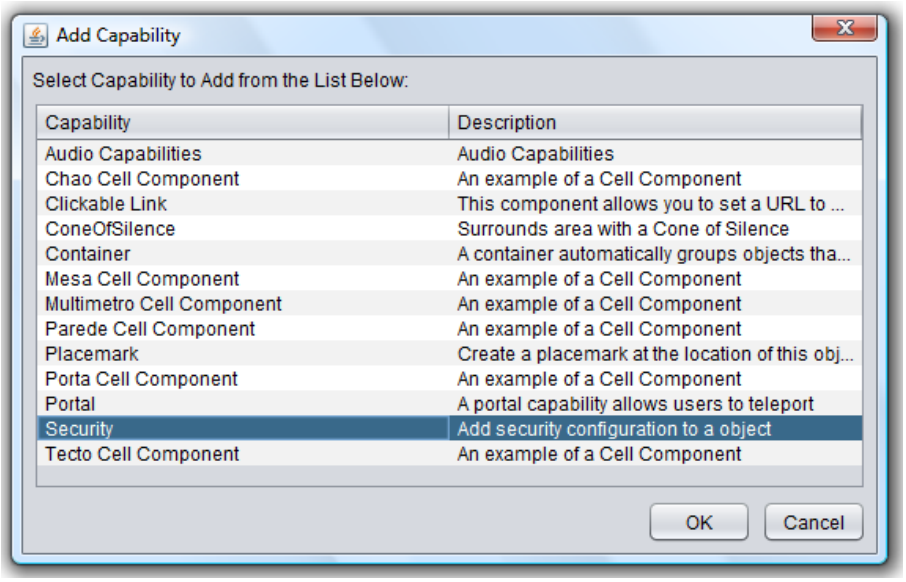


Figura 49 - Menu de adição de capacidades a objectos 3D

As permissões de um objecto 3D estão na capacidade *Security*, que está seleccionada, e será adicionada à caixa *Capabilities* da Figura 48. Outras capacidades da lista do menu, podem ser adicionadas quando clicadas sobre a sua indicação, e depois confirmadas com o botão *OK* (Figura 49). Cada capacidade adicionada à caixa *Capabilites* da Figura 48 é retirada da lista do menu anterior. Com a capacidade *Security* adicionada, é possível estabelecer as permissões que se pretendem, clicando em *Security* na caixa *Capabilities* (Figura 48). Depois, surgem as propriedades da capacidade do lado direito da Figura 48, na tabela *Permissions* (por baixo da indicação *Properties*), que mostra os utilizadores ou tipos de utilizador a quem se pretende aplicar permissões. Nessa tabela, cada linha representa um utilizador ou tipo de utilizador especificado na coluna *Type*, que pode assumir os valores *User* para um utilizador, *group* para um grupo de utilizadores e o tipo especial *Everybody*, que representa todos os restantes utilizadores ou grupos de utilizadores, não especificados na tabela. O nome dos utilizadores ou grupos está representado na coluna *Name*. A coluna *Owner* define o dono ou os donos de um objecto 3D, para o definir basta marcar o quadrado dessa coluna. No caso da Figura 48, o dono do objecto 3D é o utilizador *david*. Finalmente, a coluna *Permissions* representa o número de permissões aplicadas ao objecto para um determinado utilizador. Cabe ao dono do objecto especificar as permissões para os restantes utilizadores, para isso, pode usar as funcionalidades dos três botões situados à direita da tabela *Permissions*. O botão *Add* serve para adicionar um utilizador à tabela e o botão *Remove* retira o utilizador marcado da mesma. Com o botão *Edit* alteraram-se as permissões do utilizador seleccionado, só se aplica a utilizadores que não sejam donos do objecto 3D, pois o dono não pode alterar as permissões para si próprio, este, como espectável tem o controlo total do objecto. Para cada utilizador ou grupo de utilizadores, o botão *Edit* abre o seguinte menu da Figura 50.

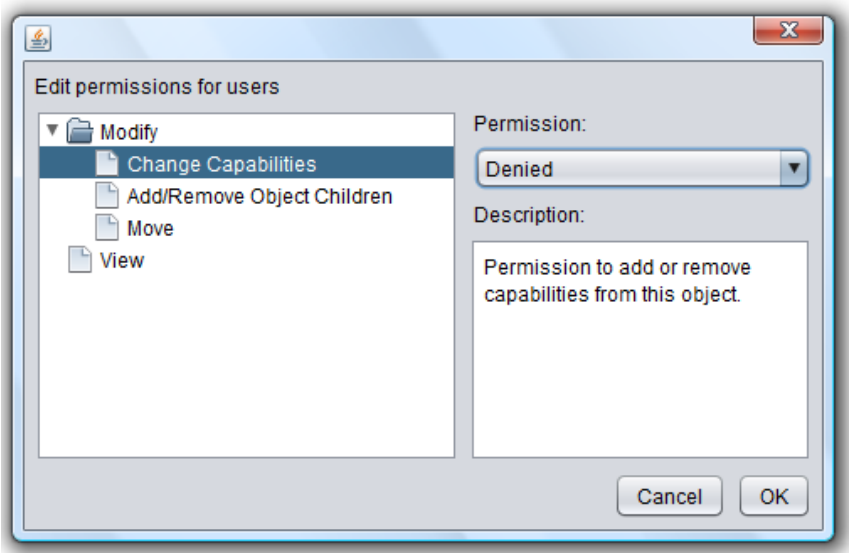


Figura 50 - Menu de edição de permissões de objectos 3D

No menu da figura anterior, na caixa do lado esquerdo estão os vários tipos de permissões. Para modificar uma permissão, basta clicar por cima do seu tipo, ficando sombreado a azul. É mostrado do lado direito, por baixo da indicação *Description*, a descrição correspondente ao tipo de permissão clicado. O valor para cada tipo de permissão é definido na caixa de selecção, do lado direito da Figura 50, por baixo da indicação *Permission*.

Existe um grupo de três botões no canto inferior direito do menu da Figura 48: *Refresh*, *Revert* e *Apply*. O mais importante e mais usado é o *Apply*, que serve para aplicar/guardar as permissões ao objecto 3D. O clique neste botão faz com que o próprio, e o botão *Revert* fique desabilitado. O botão *Refresh* limpa as selecções feitas pelo rato (linhas ou palavras seleccionadas sombreadas a azul), o botão *Revert* serve para desfazer a última alteração efectuada, desde que as definições não tenham sido aplicadas/guardadas com o botão *Apply*. O clique no *Revert* desabilita o próprio e o botão *Apply*.

Anexo 6 - Arranque automático da máquina Servidor Wonderland

Foi referido na secção 4.1, que o servidor OW é arrancado usando o comando *ant run-server* na linha de comandos, no directório onde se encontra o código fonte do servidor OW. Para automatizar este procedimento, em primeiro lugar, é necessário fazer com que o terminal da linha de comandos seja executado no arranque do SO Ubuntu 9.10. Para tal, acede-se à configuração das preferências das aplicações de arranque (Figura 51) através do menu *Sistema* →*Preferências* →*Aplicações de Arranque*.

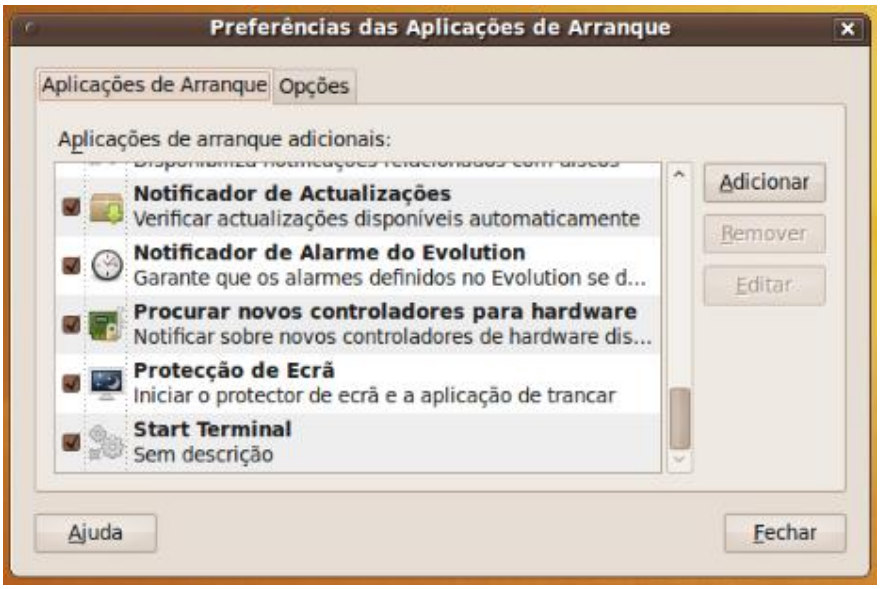


Figura 51 - Menu de configuração de aplicações de arranque

A adição de uma nova aplicação para arrancar automaticamente, é feita através do clique no botão *Adicionar* do menu da figura anterior, que faz surgir o menu de edição de aplicações de arranque da Figura 52.

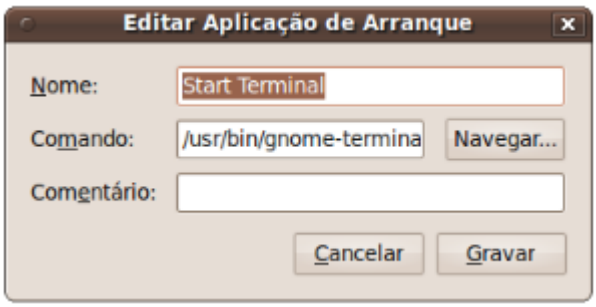


Figura 52 - Menu de edição de aplicações de arranque

O único campo obrigatório do menu da Figura 52 é o campo *Comando*, onde se deve colocar o caminho do ficheiro executável da aplicação, que pretende-se executar no arranque do SO. O conteúdo dos campos *Nome* e *Comentário*, serve para ser apresentado na lista de aplicações do menu da Figura 51, onde no campo *Nome* se coloca o nome da aplicação, e no campo *Comentário* se coloca uma

pequena descrição da aplicação. As definições da aplicação de arranque, são depois guardadas através do botão *Gravar* da Figura 52. Depois, deve-se criar um ficheiro de comandos Linux, que vai conter os comandos necessários ao arranque do servidor OW. O ficheiro é o *shell script loadw*, mostrado no Bloco de código 1.

```
#! /bin/bash

ps -e | grep "java" > temp

if [ ! -s temp ] ; then
        sudo /etc/init.d/ssh start
        rm -r temp
        sleep 5
        cd /home/laboris/david/src/0.5/0.5-preview4/wonderland
        ant run-server
else
   rm -r temp
fi
```

Bloco de código 1

Criou-se este ficheiro, para os comandos nele contidos serem executados quando se abre o terminal da linha de comandos. O comando *ps –e | grep "java" > temp* verifica se existem processos Java a executar no SO, e redirecciona o seu resultado para o ficheiro *temp*, criado pelo próprio comando se o ficheiro não existir. A finalidade deste comando, é impedir o arranque do servidor OW de cada vez que se executa o terminal da linha de comandos, já que o mesmo servidor está representado por processos Java, quando está em execução. Depois, a condição *if* vai verificar se o ficheiro *temp* está vazio, ou seja, se há processos Java a executar, caso o ficheiro *temp* não esteja vazio, é executado o comando *rm* no bloco *else* da condição *if*, para eliminar o ficheiro *temp*. Se este ficheiro estiver vazio, vão ser executados os comandos dentro do bloco *if*, onde começa por arrancar o servidor SSH no comando *sudo /etc/init.d/ssh start*. Depois, é apagado o ficheiro *temp* no comando *rm*. Utiliza-se um *sleep* para causar um atraso de cinco segundos, para dar tempo para os comandos anteriores executarem. Finalmente, executam-se os comandos directamente relacionados com o servidor OW, *cd* para navegar para o directório do código fonte, e executar *ant run-server* para arrancar o servidor. Para garantir que o ficheiro *loadw* é executado no arranque da linha de comandos, este ficheiro deve estar guardado num directório que faça parte da variável de ambiente *PATH*, e ser adicionado ao ficheiro *bash.bashrc* no caso do Ubuntu. Um ficheiro executável, pode ser livremente executado em toda a linha de comandos, quando o seu directório faz parte da variável de ambiente *PATH*. Quanto ao ficheiro *bash.bashrc*, é onde se encontram os comandos, que são executados no arranque da linha de comandos.

Um *shell script* Linux é executado pelo nome do ficheiro que o constitui, assim, basta adicionar *loadw* à última linha do ficheiro *bash.bashrc*, para o *script* executar no arranque do terminal da linha de comandos. Para se editar o ficheiro *bash.bashrc*, de qualquer directório da linha de comandos no Ubuntu, deve-se digitar o comando *sudo gedit /etc/bash.bashrc*, já que é um ficheiro com permissão

de edição só para administradores do SO. No Bloco de configuração 4, encontra-se uma amostra das linhas finais do ficheiro *bash.bashrc*.

```
export ANT_HOME=/usr/local/apache-ant-1.8.0
export JAVA_HOME=/usr/local/jdk1.6.0_18
export PATH=/usr/local/apache-ant-1.8.0/bin:$PATH
export ANT_OPTS="-XX:MaxPermSize=900m -Xmx900m"
umask 002
loadw
```

Bloco de configuração 4

A preparação do código fonte para interpretar a nova propriedade *wonderland.darkstar.autostart*, foi feita no ficheiro de código *RunManager.java*. Em primeiro lugar, na implementação da propriedade *wonderand.darkstar.autostart*, adiciona-se a variável *D_START_PROP*, inicializada com o nome da nova propriedade, de acordo com o Bloco de Código 2.

```
private static final String START_PROP = "wonderland.runner.autostart";
private static final String D_START_PROP = "wonderland.darkstart.autostart";
```

Bloco de código 2

Depois, dentro da função *doInit*, adiciona-se a variável booleana *d_start* (Bloco de código 3), que serve para captar o valor *true* ou *false*, que se encontra definido na nova propriedade do ficheiro *my.run.properties*.

```
boolean start = Boolean.parseBoolean(SystemPropertyUtil.getProperty(START_PROP));
boolean d_start = Boolean.parseBoolean(SystemPropertyUtil.getProperty(D_START_PROP));

if(start==true && d_start==true)
    d_start=false;
```

Bloco de código 3

Em ambas as variáveis booleanas do bloco de código anterior, a função *parseBoolean* converte a *string* proveniente do ficheiro *my.run.properties*, para *boolean*, através função *getProperty*. A condição *if*, serve para forçar a variável *d_start* a *false*, caso as variáveis *start* e *d_start* estejam a *true*, devido a uma configuração incorrecta de ambas as propriedades *wonderland.runner.autostart* e *wonderland.darkstar.autostart* estarem a *true*. Assim, evita-se conflito de configurações. Finalmente, ainda na função *doInit*, dentro do ciclo *for* coloca-se a seguinte condição *if*, do Bloco de código 4.

```
if (start==false && d_start==true && r.getLocation().equals(getLocation()) && r.getStatus() == Status.NOT_RUNNING)
{
    if(r.getName().equals("Darkstar Server"))
    {
        startList.add(r);
    }
}
```

Bloco de código 4

O arranque exclusivo do componente *Darkstar Server*, é garantido pela condição *if* exterior, onde as variáveis *start* e *d_start* são verificadas se estão a *false* e a *true*, respectivamente. Estas correspondem aos valores das propriedades *wonderland.runner.autostart* e *wonderland.darkstar.autostart*. A

condição *if* interior, verifica se o nome do componente do servidor a arrancar, é igual a *Darkstar Server*, caso se verifique, é executada a instrução *startList.addr(r)*, que adiciona o componente referido, à lista de componentes a arrancar. Assim, carrega-se só componente *Darkstar Server*, sem necessidade de aceder à página de gestão de componentes do servidor OW.

Anexo 7 - Instalação/Configuração do *driver* da UPS

No Ubuntu, o primeiro procedimento é instalar o pacote de software *nut*, com o comando *sudo apt-get install nut*, o pacote é automaticamente descarregado da Internet e imediatamente instalado. Depois de instalado, deve-se aceder ao ficheiro *nut* no directório */etc/default* e editá-lo de forma idêntica ao Bloco de configuração 5.

```
# start upsd
START_UPSD=yes

# start upsmon
START_UPSMON=yes
```

Bloco de configuração 5

As instruções do bloco anterior são para os processos *upsd* e *upsmon*, encarregues do funcionamento e monitorização da UPS, arrancarem automaticamente no arranque do SO.

O *driver mge-shut* da UPS é especificado no ficheiro de configuração *ups.conf*, com o conteúdo do Bloco de configuração 6, no directório */etc/nut*. Todos os ficheiros de configuração do *nut* estão neste directório.

```
[ups1]
driver=mge-shut
port=/dev/ttyS0
```

Bloco de configuração 6

No bloco anterior, a primeira linha *[ups1]* especifica o nome da UPS para o SO, a instrução *driver* define o *driver* a ser utilizado, e a instrução *port* corresponde à porta da máquina onde a UPS está ligada. No caso do Ubuntu, a porta série RS-232 é representada pelo ficheiro *ttyS0* no directório */dev*. Para o ficheiro *ttyS0* ser acedido pelo *driver*, é necessário adicionar o utilizador *nut*, incluído no pacote de software, ao grupo de utilizadores *dialout*, com o comando *useradd -G dialout nut*. Depois, deve-se usar o comando *upsdrvctl start*, para testar a detecção do *driver* da UPS pelo SO, se o teste for positivo, o comando deve ter o *Output* 5, indicando que o SO detectou a UPS com o *driver mge-shut*.

```
Network UPS Tools - UPS driver controller 2.4.1
Network UPS Tools - MGE UPS SYSTEMS/SHUT driver 0.68 (2.4.1)
Detected Pulsar Evolution 1500 [unknown] on /dev/ttyS0
```

***Output* 5**

Com o *driver* detectado, passa-se à configuração das funcionalidades de rede, neste caso, é para especificar, que apenas a máquina local acede à UPS e ao *driver* respectivo. Essa configuração é feita no ficheiro *upsd.conf*, com a propriedade de configuração seguinte.

```
ACL localhost 127.0.0.1/32 ACCEPT localhost
```

Propriedade de configuração 6

A configuração do *nut* também define os utilizadores do próprio *driver*. Estes apenas dizem respeito ao *nut*, não é obrigatório que sejam iguais aos do SO, pois existem comandos do *driver* que obrigam a

especificar o utilizador e a sua palavra-chave. Os utilizadores do *nut* configuram-se no ficheiro *upsd.users*, conforme o bloco de configuração seguinte.

```
[laboris]

  password = laborisalunos
  allowfrom = localhost
  actions = set
  upsmon master
```

Bloco de configuração 7

Na primeira linha do bloco anterior, define-se o utilizador com a instrução *[laboris]*, as instruções seguintes para o utilizador são: a palavra-chave (*password*) a utilizar em alguns comandos do *driver*, a instrução *allowfrom* especifica a máquina de onde o utilizador acede ao *driver*, que neste caso é da máquina local. O utilizador *laboris* pode modificar variáveis que façam parte da própria UPS, usando o valor *set* na instrução *actions*, e finalmente, a instrução *upsmon*, significa que este utilizador monitoriza a UPS em modo *master*, este modo de monitorização facilita a funcionalidade de encerramento automático do SO. Para o *driver* poder efectuar acções, como desligar a máquina automaticamente, num limite mínimo da carga das baterias da UPS, deve-se usar o Bloco de configuração 8, no ficheiro *upsmon.conf*.

```
MONITOR ups1@localhost 1 laboris laborisalunos master
SHUTDOWNCMD "/sbin/shutdown -h 1"
```

Bloco de configuração 8

O Bloco de configuração 8 inclui instruções de monitorização do estado da UPS. A instrução *Monitor* especifica, que se está a monitorizar a *ups1* na máquina local, com apenas uma fonte de alimentação, através do utilizador *laboris* no modo *master*. A instrução *SHUTDOWNCMD*, especifica o comando que deve ser executado, quando a carga das baterias da UPS atingir um valor mínimo, neste caso irá ser executado o comando *shutdown*, causando o encerramento do SO, depois de decorrido um minuto.

Depois destas configurações, deve-se activar a monitorização da UPS no SO, com o comando *sudo /etc/init.d/nut start*, resultando em condições normais no *Output* 6. Todos os ficheiros de configuração do pacote *nut* aqui referidos, caso não existam, devem ser criados de raiz.

```
* Starting Network UPS Tools                                    [ OK ]
```

***Output* 6**

A verificação da configuração do *nut* faz-se com o comando *upsc ups1*. Caso tudo esteja correctamente instalado e configurado, surge o *Output 7*, que é parte do resultado do comando referido.

```
laboris@laboris-salaestudo:/etc/default$ upsc ups1
battery.charge: 100
battery.charge.low: 75
battery.runtime: 1625
driver.name: mge-shut
driver.parameter.pollinterval: 2
driver.parameter.port: /dev/ttyS0
```

***Output* 7**

Para especificar o valor mínimo da carga das baterias da UPS, usa-se o Comando 4.

```
upsrw -s battery.charge.low=75 -u laboris -p laborisalunos ups1
```

Comando 4

No caso deste último comando, é estipulado o valor mínimo da carga das baterias a 75%, ou seja, quando é executado o comando da instrução *SHUTDOWNCMD*, no ficheiro *upsmon.conf* do Bloco de configuração 8, para provocar o encerramento automático do SO.

Alguns comandos são precedidos da instrução *sudo*, cuja função é executar comandos Linux em modo de administrador de sistema, quando os mesmos têm permissões restritas. Quando se executam comandos com a instrução *sudo*, é requerida a palavra-chave do utilizador.

Anexo 8 - Estrutura do DVD

O DVD a ser entregue junto com este trabalho, contém o software desenvolvido e utilizado no desenvolvimento da solução proposta. Na figura seguinte encontra-se a sua estrutura de directórios.

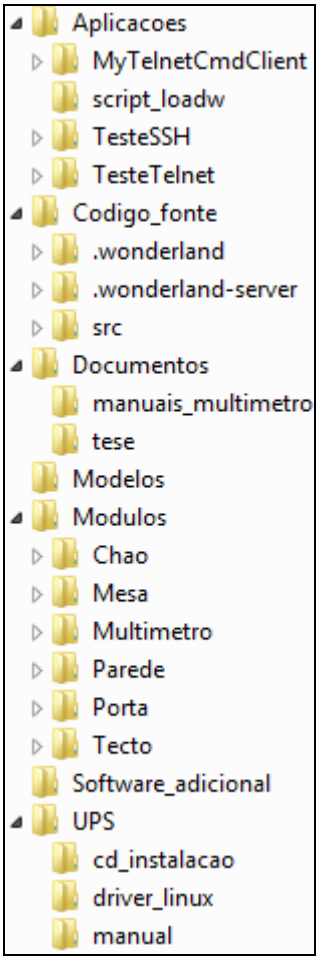


Figura 53 - Estrutura de directórios do DVD

Cada directório da raiz do DVD agrupa software ou ficheiros, de acordo com a sua especificidade. Os principais directórios são:

- **Aplicacoes:** contém aplicações auxiliares ao funcionamento da solução. Destas destacam-se, a aplicação *MyTelnetCmdClient* e o *script loadw*. Neste directório, também estão armazenadas aplicações de teste à comunicação dos protocolos *telnet* e SSH, estas últimas não são utilizadas no funcionamento da solução.

- **Codigo_fonte:** armazena todo o código fonte do servidor OW desta solução, e os seus módulos incluídos de raiz.

- **Documentos:** contém este documento em formato digital e os manuais do MR.

- **Modelos:** neste directório estão alguns modelos KMZ, que foram utilizados para construir os vários mundos virtuais referidos anteriormente. Podem ser utilizados para construir outros mundos virtuais.

- **Modulos:** contem o código dos módulos utilizados na solução, dos quais se destaca o módulo do MV.

- **Software_adicional:** armazena o software requerido à instalação e compilação do código fonte do servidor OW.

- **UPS:** tudo o que é específico da UPS encontra-se neste directório, desde a imagem do seu CD de instalação, driver para Linux, até ao seu manual de instruções.