

JFindMyFiles

Relatório Final de Projecto Informático n.º AP01 / 2007-8

Patrícia Monteiro, n.º 12435

Sérgio Lopes, n.º 10635

Relatório final submetido para avaliação parcial da unidade curricular de Projecto Informático, do curso de Engenharia Informática. Projecto sob a orientação da Professora Ana Filipa Nogueira, do Departamento de Engenharia Informática da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria.



Departamento de Engenharia Informática da
Escola Superior de Tecnologia e Gestão do
Instituto Politécnico de Leiria

22 de Julho de 2008

Agradecimentos

À orientadora, Ana Filipa Delgado Nogueira, por ter aceite e tornado possível a realização deste projecto. E pelo empenho demonstrado durante o desenvolvimento do projecto.

Tom Wheeler e Zane Cahill, por terem ido além do seu papel com a ajuda prestada nos momentos onde o conhecimento nos faltava.

A todos os utilizadores das listas de discussão de desenvolvimento e de utilizadores da plataforma NetBeans, que se nos ajudaram com as suas opiniões e enorme receptividade às perguntas colocadas.

Proposta de Projecto

Área Temática:

DA Desenvolvimento de Aplicações

Descrição:

Com o crescente aumento na capacidade de armazenamento dos suportes digitais e com o igual aumento na utilização de meios digitais no tratamento de informação surge a necessidade de aplicações que permitam facilmente catalogar, gerir e encontrar os ficheiros necessários ao utilizador.

Pretende-se desenvolver uma aplicação desktop que possa ser alternativa a algumas aplicações existentes com o mesmo objectivo, Where Is It™^I e Argentum MyFiles™^{II}, e que permita a fácil gestão de ficheiros digitais em diversos suportes, possibilitando a pesquisa e visualização de algumas propriedades mesmo que os ficheiros não estejam disponíveis.

Com o *JFindMyFiles* o utilizador poderá guardar a informação sobre a estrutura e conteúdo de determinado directório a que tenha acesso. Com essa informação ser-lhe-á possível efectuar a catalogação e gestão da informação, bem como a pesquisa e obtenção de dados sobre os ficheiros sem necessidade de aceder aos suportes onde os mesmos se encontram guardados.

O software terá algumas características, consideradas base como:

- Permitir a inserção, remoção, alteração e edição das informações sobre ficheiros;
- Oferecer a capacidade de leitura de determinado suporte e importar a informação de todos os ficheiros contidos;
- Permitir guardar a informação em bases de dados remotas ou usar um sistema de persistência local;
- Permitir criar relatórios da informação e exportar a informação em formatos que possam ser lidos por aplicações externas;
- Permitir a adição de novos tipos de informação a manter, seja de formatos de ficheiro existentes seja de formatos desconhecidos na altura do desenvolvimento da aplicação, através de um sistema de *plugins*;

^I <http://www.whereisit-soft.com>

^{II} <http://www.argentuma.com/myfiles.html>

- Pesquisa por ficheiros, visualização e listagem de ficheiros mesmo que o suporte não esteja disponível. Pode incluir pré-visualização, *thumbnail* no caso de imagens, etc.

Pretende-se usar a plataforma Java no desenvolvimento da aplicação de forma a garantir a capacidade multi-plaforma.

No final serão entregues os seguintes elementos: relatório de projecto, aplicação desenvolvida, sistema de ajuda ao utilizador e manual de programador para sistema de *plugins*.

Requisitos:

Para a realização do projecto são necessários conhecimentos em programação orientada a objectos, linguagem SQL e de utilização de sistemas de gestão de bases de dados relacionais.

Parcerias: “sem parcerias”

Elementos do grupo: Patrícia Monteiro, aluna ei12435 e Sérgio Lopes, alunos ei10635

Orientadores: Ana Filipa Nogueira

Docente responsável pela proposta: Rui Ferreira

Resumo

O projecto tem como objectivo o desenvolvimento de uma aplicação para computador pessoal, capaz de obter e catalogar informação sobre ficheiros de computador, o que vulgarmente se designa por *Digital Media Assessment*[46], de modo a colmatar a falta de uma aplicação de uso livre e multi-plataforma que permita maior mobilidade entre diferentes sistemas operativos.

No sentido de conhecer o estado de desenvolvimento actual na área, foi feita uma análise de aplicações concorrentes, de forma a avaliar as suas características e pontos fracos.

Abordando o problema através da metodologia de Programação Extrema (XP), foram usadas ferramentas e recursos que, após leve análise, se revelaram adequadas, como Subversion para controlo de versões, Project Pier para gestão de projectos, entre outros. Foram aplicados testes de unidade ao código desenvolvido, testes de aceitação às várias versões criadas e testes de usabilidade ao produto final.

Foi, também, escolhida uma plataforma de desenvolvimento de aplicações ricas (RCP), após terem sido estudadas quatro opções. Escolha esta que condicionou a linguagem de programação e o ambiente de desenvolvimento integrado (IDE) usado.

Do desenvolvimento resultou a criação de cinco versões do produto, cada uma incrementando as funcionalidades implementadas nas versões anteriores, culminando, na versão 1.0, com um aplicação funcional e usável, que cumpre os objectivos propostos.

Índice

1	Introdução e Motivação	14
1.1	<i>Objectivos Atingidos</i>	<i>15</i>
2	Metodologia de Desenvolvimento.....	15
3	Estudo e Selecção de Tecnologias.....	17
3.1	<i>Plataforma RCP.....</i>	<i>17</i>
3.2	<i>Motores de Bases de Dados Relacionais</i>	<i>17</i>
3.3	<i>Tecnologias ORM</i>	<i>18</i>
3.4	<i>Outras Bibliotecas e Tecnologias</i>	<i>19</i>
4	Ferramentas Utilizadas	19
4.1	<i>Gantt Project.....</i>	<i>19</i>
4.2	<i>Project Pier.....</i>	<i>20</i>
4.3	<i>NetBeans IDE.....</i>	<i>21</i>
4.4	<i>Sistema de Controlo de Versões.....</i>	<i>21</i>
4.5	<i>Notepad++ e GEdit</i>	<i>21</i>
4.6	<i>Sistemas de apoio BerliOS.....</i>	<i>21</i>
4.7	<i>XTest e JUnit.....</i>	<i>22</i>
5	Desenvolvimento do Projecto	23
5.1	<i>User Stories.....</i>	<i>23</i>
5.2	<i>Planeamento Inicial.....</i>	<i>23</i>
5.2.1	<i>Reunião de Planeamento.....</i>	<i>23</i>
5.3	<i>Iteração 1</i>	<i>24</i>
5.3.1	<i>Avaliação</i>	<i>24</i>
5.4	<i>Iteração 2.....</i>	<i>28</i>
5.4.1	<i>Avaliação</i>	<i>31</i>
5.5	<i>Iteração 3.....</i>	<i>33</i>
5.5.1	<i>Avaliação</i>	<i>34</i>
5.6	<i>Iteração 4.....</i>	<i>37</i>

5.6.1	Avaliação	38
5.7	Iteração 5.....	38
5.7.1	Avaliação	40
5.8	Iteração 6.....	40
5.8.1	Avaliação	41
5.9	Iteração 7.....	41
5.9.1	Avaliação	42
5.10	Iterações 8 e 9.....	43
5.11	Spike Solutions	43
5.11.1	CRC.....	43
5.11.2	ExecutarFicheiro	44
5.11.3	PluginAPI.....	44
5.11.4	SimpleExport.....	44
5.11.5	SPKlerDados	45
5.11.6	TemplateExport.....	45
5.12	Testes de Usabilidade	46
5.13	Testes de unidade.....	48
5.14	Conclusões	49
6	Produto Criado	50
7	Desenvolvimentos Futuros	51
8	Anexos.....	52
8.1	Estudo da Metodologia	52
8.2	Estudo da Plataforma RCP.....	52
8.3	Sistema de Controlo de Versões.....	56
8.3.1	Clientes SVN	56
8.4	Avaliação do Estado da Arte	58
8.5	User Stories.....	60
8.5.1	Lista de User Stories.....	60
8.5.2	Distribuição das User Stories por Iterações	62
8.6	Convenções de Código.....	65
8.7	Testes de Usabilidade	66
8.7.1	Questionários Apresentados	66
8.7.2	Resultados dos Testes de Usabilidade	69

8.8	<i>Imagens do Produto final</i>	71
9	Bibliografia	78

Índice de figuras

Figura 1.	Fluxograma da metodologia de Programação Extrema.....	16
Figura 2.	Ecrã inicial do sistema Project Pier, visão geral do projecto.....	20
Figura 3.	Gráfico de Gantt com o planeamento do projecto	24
Figura 4.	Windows XP, à esquerda, e aplicação de teste, à direita.	25
Figura 5.	Ubuntu 8.04, à esquerda, e aplicação de teste, à direita.	25
Figura 6.	Evolução do consumo de memória com algoritmo recursivo.....	26
Figura 7.	Evolução do consumo de memória com algoritmo iterativo	27
Figura 8.	Sistema Where Is It™.....	29
Figura 9.	Modelo de domínio.....	30
Figura 10.	Representação visual dos dados	32
Figura 11.	Representação da Nodes And Explorer API[45]	33
Figura 12.	Package “radingfiles”	35
Figura 13.	Package “pluginapi”	36
Figura 14.	Package “plugins”	36
Figura 15.	Sistema JFindMyFiles, ecrã principal.	39
Figura 16.	Sistema JFindMyFiles, menu de edição.	39
Figura 17.	Template criado	42
Figura 18.	Aplicação de teste criada para a spike solution SPKlerDados.....	45
Figura 19.	Formulário de Teste de Usabilidade no sistema Google Docs	47
Figura 20.	Resultados dos testes de unidade apresentados no IDE.....	48
Figura 21.	Resultados dos testes de unidade apresentados em HTML	49

Figura 22.	Sistema JFindMyFiles em execução.....	51
Figura 23.	Integração do TortoiseSVN na shell do MS Windows XP.....	57
Figura 24.	Janela principal do SmartSVN	58
Figura 25.	Sistema sem catálogo aberto.....	71
Figura 26.	Janela que permite abrir um catálogo.....	72
Figura 27.	Configuração das extensões existentes.....	73
Figura 28.	Adição de um grupo de discos.....	74
Figura 29.	Janela que permite iniciar a leitura de um disco.....	75
Figura 30.	Processo de leitura de um disco.....	76
Figura 31.	Janela principal do sistema com catálogo aberto.....	77

Índice de tabelas

Tabela 1.	Comparação entre aplicações	59
Tabela 2.	User stories	60
Tabela 3.	Lista de user stories agrupadas	63
Tabela 4.	Resultados do primeiro questionário	69
Tabela 5.	Resultados do segundo questionário.....	70

Dicionário de Acrónimos

Acrónimo	Significado
API	Application Programming Interface
AMC	Ant Movie Catalog
CD-ROM	Compact Disc-Read Only Memory
CSV	Comma Separated Value
CVS	Concurrent Versions System
DVD-ROM	Digital Versatil Disc – Read Only Memory
EXIF	Exchangeable Image File Format
HDD	Hard Disk Drive
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
JAR	Java Archive
JNI	Java Native Interface
JPA	Java Persistence API
JPG	Joint Photografic Experts Group
JTA	Java Transition API
JVM	Java Virtual Machine

MD5	Message Digest 5
MFC	Microsoft Foundation Classes
ODS	OpenOffice Spreadsheet
ORM	Object-relational Mapping
RCP	Rich Client Platform
RFC	Request for Comments
SAX	Simple API for XML
SHA-1	Secure Hash Algorithm 1
SQL	Structured Query Language
SO	Sistema Operativo
SVN	Subversion
SWT	Standard Widegt Toolkit
URL	Uniform Resouce Locator
XLS	Microsoft Excel Spreadsheet
XML	Extensible Markup Language
XP	Extreme Programming
XSD	XML Schema Definition

1 Introdução e Motivação

O cada vez mais fácil acesso à informação, aliado à expansão da capacidade dos sistemas de armazenamento e das velocidades das ligações de Internet, bem como da maior utilização de sistemas digitais de fotografia e vídeo, faz com que, um utilizador comum de computador pessoal, possua uma enorme variedade de ficheiros, espalhados por discos rígidos, CD-ROMs, DVD-ROMs, discos Blu-Ray, discos FLASH, entre outros.

Devido a esta situação, e principalmente no que toca a fotografias digitais, a oferta de *software* que permita catalogar e gerir toda a informação, tem vindo a aumentar consideravelmente. No entanto, essa oferta acaba por se manter apenas no campo da fotografia digital e negligenciar outros tipos de informação, como sendo *e-books*, documentos de escritório, facturas electrónicas, entre outros documentos em formato digital.

É objectivo dos autores deste projecto, a criação de um sistema de catalogação e gestão de ficheiros, de utilização livre e portátil, que venha suprimir a necessidade evidenciada. Desta forma, como resultado do projecto, pretende-se um produto de *software* funcional e usável, com características multi-plataforma, de uso livre através de uma licença *Open Source*, e que possa ser útil, tanto a utilizadores domésticos como a empresas, na gestão dos seus bens digitais.

Para esse efeito, e após um estudo das metodologias que se adaptavam às características do projecto, foi escolhida uma metodologia que privilegiasse o desenvolvimento em detrimento de análises mais extensas, neste caso a metodologia XP.

Foi, também, feito um estudo sobre a possibilidade de se usar uma das várias *frameworks*^{III} de desenvolvimento de aplicações ricas, de forma a aliviar a carga a nível de programação. Foi assim escolhida a plataforma NetBeans, com base na qual se desenvolveu a aplicação. A escolha de uma plataforma RCP implicou um enorme esforço de aprendizagem, uma vez que o desenvolvimento foi feito em paralelo com o período de adaptação e conhecimento da plataforma.

Após escolhida a plataforma, foram determinadas, as *user stories*, feito o planeamento inicial, a determinação das iterações e suas durações, e foram criadas as diversas *spike solutions*, que pretenderam eliminar algumas dúvidas, de carácter técnico, que surgiram.

^{III} Uma *framework*, neste contexto, define um esqueleto de uma aplicação que pode ser adaptado a diferentes propósitos por um programador.

O conteúdo deste relatório compreende, por ordem de apresentação, os pontos de, metodologia escolhida, tecnologias e ferramentas usadas, desenvolvimento do projecto, que inclui a informação referente às diversas iterações, como documentos resultantes da análise, desenvolvimentos futuros, onde se indicam funcionalidades não implementadas, correcções a fazer e opções para a melhoria da aplicação. Apresenta-se ainda, uma secção final onde se incluem todos os anexos.

Acompanham o relatório, 3 manuais. O de instalação do sistema, o de utilizador e o de programador de *templates* e extensões.

1.1 Objectivos Atingidos

A aplicação desenvolvida permite criar catálogos com informação sobre ficheiros, lidas de discos CD-ROM, HDD, DVD-ROM, FLASH ou qualquer outra pasta específica, acessível ao computador.

Possibilita a organização de discos, visualização do seu conteúdo, pesquisa de ficheiros existentes no catálogo, pesquisa de ficheiro duplicados e abertura de ficheiros, quando acessíveis ao programa, nas suas aplicações pré-definidas.

O sistema permite a exportação de dados para formatos de texto, nomeadamente XML e CSV, e a importação de dados de ficheiros XML, criados previamente através do mecanismo de exportação. Além de ficheiros de texto, foi criada a funcionalidade para exportar dados usando *templates* HTML.

É também fornecida uma API que permite a criação de extensões ao sistema de leitura, de forma a serem fornecidas outras informações diferentes das obtidas por omissão.

2 Metodologia de Desenvolvimento

Existem diversas metodologias de desenvolvimento, cada uma com as suas vantagens, desvantagens, e adequação a determinado projecto ou objectivo, pelo que, após avaliados os objectivos a atingir e condicionantes inerentes, foi escolhida, como metodologia de desenvolvimento, a metodologia XP.

A metodologia XP é uma metodologia ágil que como tal valoriza, segundo o *Agile Manifesto*[9]:

- Pessoas e interacções, sobre processos e ferramentas;

- Sistemas funcionais, sobre documentação compreensiva;
- Colaboração com o cliente, sobre negociação de contractos;
- Adaptação à mudança, sobre o seguir um plano.

A metodologia XP, indicada para equipas pequenas, em projectos com requisitos vagos, indeterminados, ou em constante mudança, oferece um modo eficaz de planeamento e controlo do desenvolvimento do projecto, permitindo assim, garantir a qualidade dos produtos criados[7].

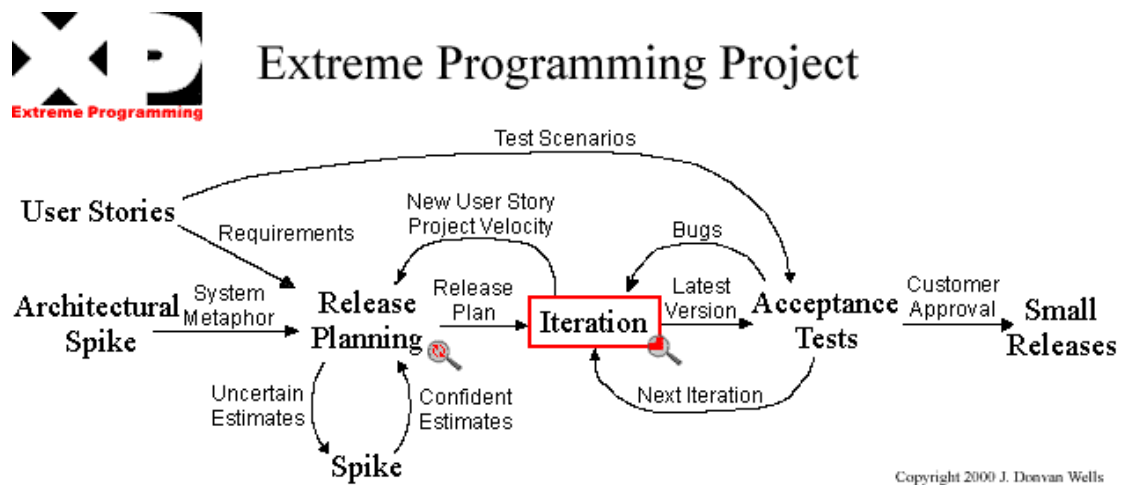


Figura 1. Fluxograma da metodologia de Programação Extrema

Como se pode observar no fluxograma da metodologia[7], Figura 1, após o planeamento de distribuição e desenvolvimento de *spike solutions* iniciais, toma lugar o desenvolvimento iterativo da aplicação, durante o qual é mantido o planeamento de iterações e são encorajadas as práticas que dão suporte à metodologia, como:

- Código Colaborativo, onde todos os programadores da equipa são incentivados a partilhar o controlo do código, não existindo monopólio de um programador sobre determinado código;
- Aplicação de testes de unidade e de aceitação.

3 Estudo e Selecção de Tecnologias

3.1 Plataforma RCP

Cada vez mais se verifica que existem diversas tarefas repetidas no desenvolvimento de aplicações, facto que se torna ainda mais evidente quando se pensa em aplicações para *desktop*.

As plataformas de desenvolvimento de aplicações ricas, RCP, são *frameworks* que tratam de fornecer ao programador, todas as tarefas comuns a aplicações *desktop*. Tarefas como a criação de menus, gestão da selecção, gestão de janelas, persistência de definições, entre outros.[47]

Consideradas quatro plataformas, a MFC, Eclipse RCP, a Spring Rich Client Platform (Spring Rich) e a NetBeans Platform, a escolha recaiu sobre a NetBeans Platform, em detrimento da MFC devido à necessidade uma aplicação multi-plataforma, e em detrimento da Spring Rich devido à infância desta última e ao facto de estar a sofrer uma reestruturação que estará completa em Outubro de 2008. Devido a alguns erros encontrados no sistema SWT[1][2][3], e devido ao mesmo não ser *standard* na tecnologia Java, a Eclipse RCP foi também eliminada.

A análise completa das plataformas pode ser consultada no anexo 8.2.

3.2 Motores de Bases de Dados Relacionais

Pretende-se que a aplicação não seja dependente do motor de bases de dados, usado no apoio à persistência dos dados. No entanto, não é tecnicamente possível manter uma completa independência e, ao mesmo tempo, proporcionar ao utilizador uma forma simples de escolher o motor a usar.

Tal se deve ao facto, de ser necessário adicionar uma biblioteca externa que possibilite a comunicação com o motor escolhido e de, ser também necessário, configurar cada uma das bibliotecas presentes.

Foi assim decidido, escolher alguns dos motores de bases de dados relacionais mais comuns, com particular ênfase em motores *Open Source*, e criar suporte na aplicação para apenas esses motores, sendo que, a nível de programação, não existirá qualquer impedimento ou dificuldade, para se aumentar o número de motores suportados.

Os motores escolhidos foram: Firebird[55][12], a partir da sua versão 2.1, Microsoft SQL Server, a partir da versão 2005, MySQL[13], a partir da versão 5, PostgreSQL[14], a partir da versão 8.

Dado que será também necessário que a aplicação guarde os dados localmente e, para evitar configurações por parte do utilizador, facilitar a instalação da aplicação e o desenvolvimento e distribuição da mesma, foi escolhido incorporar um motor de bases de dados na aplicação. Desta forma, será transparente para o utilizador a forma como os dados são guardados, evitando também, a necessidade de escrever código diferente para o acesso à base de dados local.

Para este efeito foi escolhido o motor HSQLDB[15], na versão 1.8.9, uma implementação de um motor de bases de dados relacionais, completamente em Java, o que garante a independência do SO, e permite a fácil integração com a aplicação a desenvolver.

Para incorporar o motor HSQLDB, foi usado o modo de execução em processo. Este é um dos dois modos de execução que este motor disponibiliza, sendo o segundo modo, o de execução servidor.

O modo de execução em processo possibilita a utilização por parte de uma aplicação Java, sendo o motor iniciado na mesma JVM que a aplicação e controlado, em todos os pontos, através do código da aplicação. Este modo apenas permite comunicações que partam da JVM onde foi iniciado o motor e não aceita ligações de aplicações externas.

3.3 Tecnologias ORM

Para se conseguir uma independência do motor de base de dados é necessário que exista uma camada de abstracção que permita aceder aos dados sem escrever código que seja específico do motor a usar. Nesse sentido, e dado que a linguagem a usar é orientada a objectos, considera-se o uso de uma tecnologia ORM, que permita desenvolver a aplicação sem preocupações com o sistema de persistência usado.

Existem várias tecnologias ORM para a linguagem Java, e é possível encontrar algumas comparações de características oferecidas[16], mas das duas tecnologias conhecidas pela equipa, TopLink Essentials[17] e Hibernate[18], não existem diferenças significativas que influenciem o desenvolvimento do projecto, e, uma vez que a equipa de desenvolvimento possui alguma experiência com a tecnologia Hibernate, considerou-se suficiente esse conhecimento para a escolha da tecnologia.

3.4 Outras Bibliotecas e Tecnologias

- JCalendar[19], conjunto de componentes visuais para a escolha de datas, permite mostrar componentes de interacção com o utilizador para a escolha e validação de datas.
- JackSum[20], biblioteca que implementa vários algoritmos de *hash*, e que permite a criação, de forma simplificada, de códigos SHA-1^{IV}, para os ficheiros lidos.
- Metadata Extractor[21], possibilita a extracção de informação EXIF a partir de ficheiros JPG.

As bibliotecas MySQL Connector[22], jTDS[23], PostgreSQL JDBC Driver[24] e JayBird[25], implementam *drivers* de ligação aos motores de bases de dados suportados, respectivamente, MySQL, MS SQL Server, PostgreSQL e FireBird. De notar que, embora a Microsoft tenha criado um *driver* de ligação JDBC à base de dados MS SQL Server, a sua licença impediria a distribuição juntamente com a aplicação, especificamente a alínea c), do ponto 2 e o ponto 4, da licença de redistribuição[26].

4 Ferramentas Utilizadas

4.1 Gantt Project

De forma a criar o calendário de projecto foi escolhido o sistema Gantt Project[27], um *software* de gestão e planeamento de projectos baseado em gráficos e Gantt, que, por ser implementado usando a tecnologia Java, permite o seu uso em qualquer SO para o qual exista uma máquina virtual Java.

Um gráfico de Gantt permite representar o planeamento de um projecto, indicando as datas de início e de fim dos elementos que compreendem o projecto. Possibilita também perceber o grau de cumprimento do projecto.

^{IV} Uma função de *hash* criptográfica que permite obter uma mensagem de tamanho fixo, 160 *bits*, a partir de um valor de entrada de tamanho variável.

4.2 Project Pier

A utilização de um sistema de gestão de projectos baseado em tarefas ajudará a manter a equipa focada nos objectivos de cada iteração, bem como a agrupar e separar as tarefas referentes às diversas iterações.

Nesse sentido, foi escolhido o sistema Project Pier[28], como gestor de tarefas a usar, pela sua simplicidade de instalação e gestão, o que evitou a perda de tempo na configuração inicial e manutenção, pela facilidade de utilização e pelo facto de ser um sistema já conhecido da equipa, o que fez com que a sua utilização não acrescentasse dificuldades no desenvolvimento do projecto.

O sistema fornece funcionalidades de gestão de projectos por tarefas e etapas, gestão de utilizadores e permissões, gestão de ficheiros e documentação associada a cada projecto, entre outras.

The screenshot displays the Project Pier web application interface. At the top, there is a navigation bar with tabs for Overview, Messages, Tasks, Milestones, Files, Tags, Forms, and People. A user greeting 'Welcome back Knitter (Logout)' and links for Account, Projects, and Administration are visible. The main content area is titled 'Overview' and includes a project status section (Active), involved companies (none), and RSS feeds (Recent activities). A central section lists milestones, categorized into 'Late milestones / Today milestones' and 'Upcoming milestones (in next 30 days)'. A table at the bottom shows recent activities, including updates to milestones and a task.

Details	Taken on, by
MILESTONE 'Lançamento da quarta versão.' updated	Today 10:58, Knitter
MILESTONE 'Apresentação de Documentação' updated	Today 10:57, Knitter
MILESTONE 'Versão 1.0' updated	Today 10:57, Knitter
TASK 'Verificar o texto de avaliação de sistemas exist...' closed	Jul 01, 2008, Knitter

Figura 2. Ecrã inicial do sistema Project Pier, visão geral do projecto

O sistema de gestão poderá ser encontrado em <http://jfindmyfiles.berlios.de/pier>, no entanto, o mesmo é de acesso restrito.

4.3 NetBeans IDE

A escolha do Ambiente de Desenvolvimento Integrado (IDE), foi limitada pela escolha da plataforma RCP uma vez que, para usar a plataforma NetBeans, foi conveniente programar através do NetBeans IDE[29].

Seria possível desenvolver a aplicação usando outro IDE, ou mesmo um simples editor de texto, no entanto, tal escolha impediria a utilização dos diversos assistentes que pretendem ajudar na criação de aplicações usando a plataforma NetBeans.

4.4 Sistema de Controlo de Versões

Para o controlo de versões foi escolhido o sistema Subversion e como clientes o sistema TortoiseSVN[36], para sistemas operativos Windows, e o sistema SmartSVN[37], para sistemas operativos Linux.

Uma explicação da necessidade de um sistema de controlo de versões e das características dos dois clientes pode ser consultada no anexo 8.3.

4.5 Notepad++ e GEdit

Para a rápida edição de documentação, especialmente durante as diversas iterações efectuadas, foram utilizados como editores de texto, o Notepad++[30], para edição de texto em sistemas operativos MS Windows, e o GEdit[31], para sistemas GNU Linux com Gnome como gestor de janelas.

A escrita de documentação intermédia em ficheiros de texto, permite que a equipa se foque no conteúdo e não na apresentação e possibilita a sua fácil edição, independentemente do SO usado. Ao que se acrescenta o facto de, ao ser usado um sistema de controlo de versões, ser mais fácil a resolução de eventuais conflitos em ficheiros de texto, que em ficheiros binários, como os usados pelas diversas aplicações de produtividade, como MS Office, OpenOffice, entre outros.

4.6 Sistemas de apoio BerliOS

Sendo um dos objectivos a criação de um produto *Open Source* e, dadas as exigências a nível de ferramentas, como o uso de um sistema de controlo de versões e consequente necessidade

de um servidor, considerou-se a submissão do projecto a um dos vários repositórios de projectos *Open Source*.

Escolheu-se o repositório BerliOS[48] pelas versões de *software* e políticas de utilização de serviços, que permitem que a instalação do *software* Project Pier, seja possível.

Como recursos usados, são de referir:

- Página do projecto: <http://developer.berlios.de/projects/jfindmyfiles>;
- Sítio do projecto: <http://jfindmyfiles.berlios.de> e ferramentas associadas;
- Listas de discussão disponíveis:
 - Lista de *commits*: jfindmyfiles-commits@lists.berlios.de;
 - Lista de utilizadores: jfindmyfiles-users@lists.berlios.de;
- URL do repositório SVN: <http://svn.berlios.de/svnroot/repos/jfindmyfiles/trunk>.

4.7 XTest e JUnit

XTest[32] é uma *framework* de testes genérica, para uso em projectos Java. A *framework* fornece um conjunto de utilitários, *templates*, configurações exemplo e recomendações de como criar um ambiente de teste flexível e fácil de usar.

O suporte para esta *framework* foi introduzido no IDE NetBeans, de forma a ser a base para todos os testes no desenvolvimento, tanto da plataforma NetBeans como do IDE NetBeans.

A *framework* tem por base ferramentas como o sistema Ant[33] ou o sistema de testes JUnit[34] e providencia a base para a utilização de um conjunto de tecnologias de teste.

JUnit é uma *framework* para a criação e execução de testes repetíveis e, dado a necessidade de execução de testes de unidade, foi a ferramenta escolhida para os criar.

Estas duas ferramentas foram escolhidas pelo suporte oferecido no IDE usado, por serem tecnologias bastante usadas e com boa documentação, bem como por serem, elas mesmas, usadas pela plataforma RCP escolhida.

5 Desenvolvimento do Projecto

Nesta secção é apresentado o planeamento, desenvolvimento e avaliação do desenvolvimento do projecto. Encontra-se organizada para que surjam, pela ordem em que aparecem no fluxograma da metodologia XP (Figura 1), os elementos que compõem o desenvolvimento. Uma vez que a metodologia XP o divide por iterações, cada uma com a sua fase de análise, execução e conclusão, as iterações são descritas com um texto introdutório, onde se indicam os objectivos pretendidos, e com uma avaliação, onde se verifica se os ditos objectivos foram atingidos.

5.1 User Stories

A determinação de *user stories* é feita em conjunto com os clientes que, neste caso, são a equipa de desenvolvimento. Assim, as *user stories* apresentadas, são o resultado da experiência que os elementos da equipa possuíam com *software* similar, e do conjunto de funcionalidades que a equipa considerou úteis, do ponto de vista de utilização.

5.2 Planeamento Inicial

A fase de planeamento inicial compreendeu a avaliação das *user stories* encontradas, a sua distribuição por várias iterações, e a criação de um planeamento para a execução do projecto.

5.2.1 Reunião de Planeamento

Na reunião de planeamento, as *user stories* foram separadas em grupos de acordo com o grau de dependência e proximidade. Para cada um desses grupos foi determinado o tempo de desenvolvimento ideal^V, que foi posteriormente transformado em tempo de desenvolvimento real através de um peso^{VI} de valor 1^{VII}.

Na Figura 3, está presente o gráfico de Gantt que representa o calendário elaborado e a sequência de iterações desenvolvidas.

^V De acordo com a metodologia escolhida, o tempo de desenvolvimento ideal corresponde ao tempo que um programador demoraria a completar uma tarefa sem interrupções, distrações ou outro atraso[7].

^{VI} Um peso indica a que período de tempo real corresponde o tempo ideal indicado pelo programador. Tipicamente uma semana de desenvolvimento ideal tem uma correspondência entre uma semana e meia a três semanas de desenvolvimento real[7].

^{VII} Embora o peso escolhido se afaste do que é normal usar, não seria possível compreender o período de desenvolvimento no tempo existente para o projecto se escolhêssemos um peso superior.

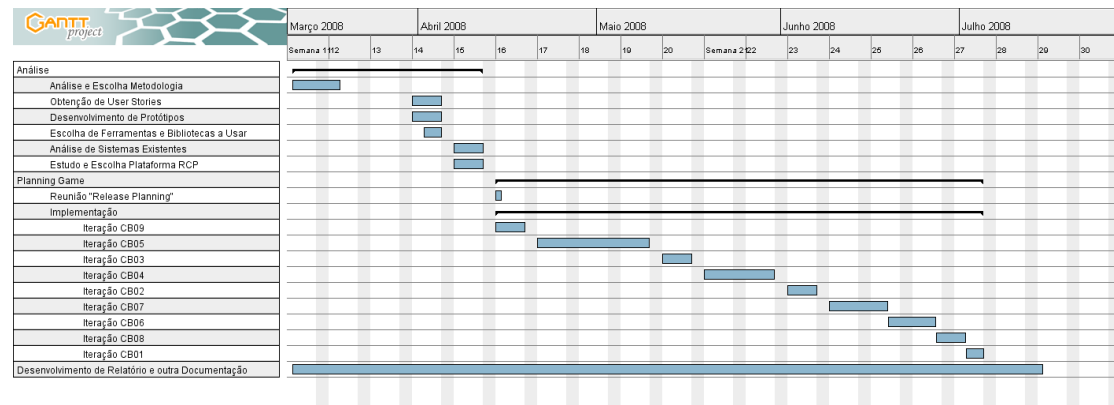


Figura 3. Gráfico de Gantt com o planeamento do projecto

5.3 Iteração 1

Esta primeira iteração foi condicionada por ter visto parte do seu tempo usado para a reunião de planeamento e pelo facto de ser o início de desenvolvimento com um conjunto de tecnologias novas que levaram à necessidade de um período de adaptação.

Para esta iteração considerou-se suficiente o desenvolvimento de código de teste, que avalie a viabilidade de ler uma pasta do disco ou uma *drive*, de forma a obter informações sobre os ficheiros contidos, bem como as limitações e dificuldades que possam existir.

Decidiu-se ser necessário obter, pelo menos, as seguintes informações sobre a lista de ficheiros existentes:

- Tamanho dos ficheiros, incluindo o tamanho das pastas contidas;
- Datas de criação e modificação dos ficheiros;
- Atributos base, como a indicação de ficheiros escondidos;
- Os nomes e caminhos completos dos ficheiros;
- No caso de *drives*, obter o nome atribuído à *drive* e o seu número de identificação.

5.3.1 Avaliação

Os testes efectuados verificaram que a linguagem usada, não permite obter alguns dos dados inicialmente propostos, como sejam as datas de criação de ficheiros, os nomes das *drives* e os seus números de identificação.

No caso das datas de criação dos ficheiros, considerou-se que, por ser um atributo dependente do SO, tal atributo não será guardado. No caso da obtenção dos nomes e dos números de identificação das *drives*, e dado que a tecnologia usada não permite obter esse tipo de informação, a única solução encontrada será a criação de bibliotecas usando JNI.

A utilização de JNI obriga à criação código em linguagem C ou C++ que faça uso directo da API do SO, o que implica a criação de tantas bibliotecas quantos os sistemas operativos nos quais a aplicação irá correr. Pelos problemas associados, decidiu-se não seguir pela via JNI.

Na obtenção do tamanho de ficheiros, foi detectado um contratempo: o valor reportado para o tamanho de uma pasta varia de SO para SO, nem sempre sendo o valor de todo o seu conteúdo. Por esta razão, se o mesmo catálogo for usado para guardar dados de ficheiros lidos em sistemas operativos diferentes, é possível que existam ligeiras diferenças nos valores totais para os tamanhos das pastas. As figuras seguintes, Figura 4 e Figura 5, obtidas em dois sistemas operativos diferentes, Windows XP SP2 e Ubuntu 8.04, respectivamente, demonstram esta situação.

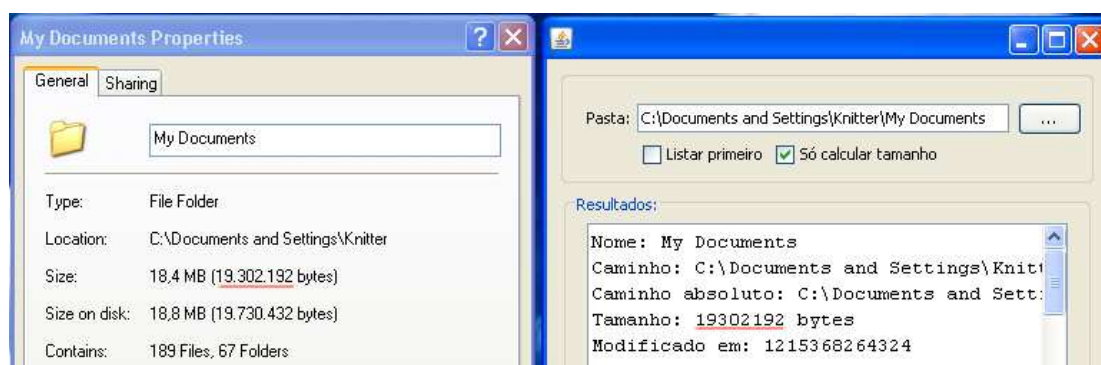


Figura 4. Windows XP, à esquerda, e aplicação de teste, à direita.

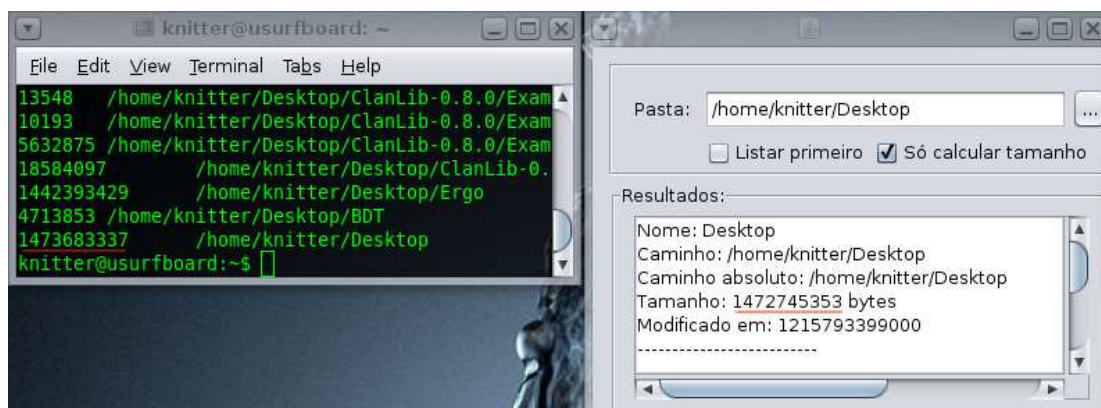


Figura 5. Ubuntu 8.04, à esquerda, e aplicação de teste, à direita.

No desenvolvimento do algoritmo de leitura de ficheiros, a primeira abordagem foi a de desenvolver um método recursivo, dado que, para a tarefa de ler ficheiros, indo pasta a pasta, a recursividade aparece de forma natural. Esta abordagem levou ao aparecimento de outra limitação da plataforma: sem cuidado é extremamente fácil usar demasiada memória, mais do que a disponível por omissão a uma JVM.

A pesquisa a uma pasta, com pouco mais de uma centena de ficheiros e subpastas, resultava no consumo total da memória disponível, neste caso 64MB, resultando no lançamento de excepções por falta de memória.

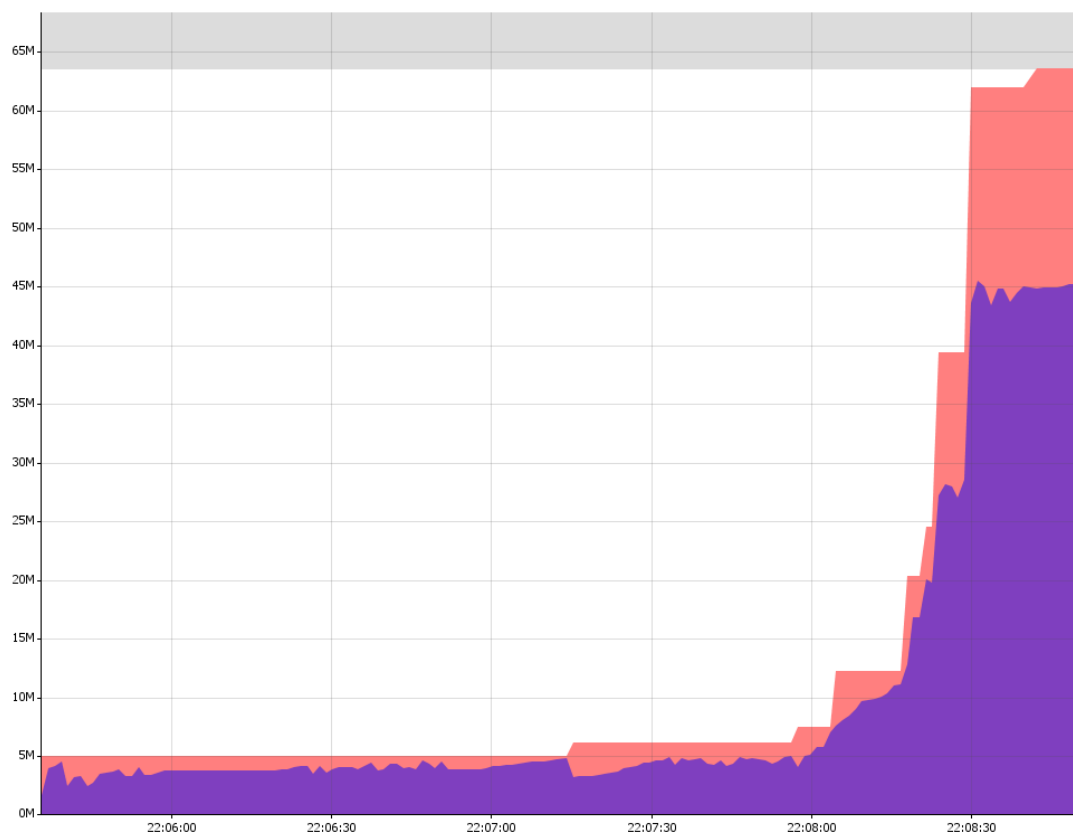


Figura 6. Evolução do consumo de memória com algoritmo recursivo.

Como se pode verificar pelo gráfico da Figura 6, que representa, no eixo das abcissas o tempo de execução e no das ordenadas, os valores de memória disponível, a pesquisa nas subpastas consome a memória da aplicação de forma muito rápida. De notar que, pela forma como está implementado o código, os ficheiros que representam pastas são lidos no fim, daí a forma do gráfico.

Para resolver este problema foi desenvolvido um segundo algoritmo, desta vez iterativo, e recorrendo ao uso de pilhas, que, mesmo na sua versão não optimizada, conseguia obter as

informações desejadas sem consumir toda a memória disponível, Figura 7. Este algoritmo permite também, e de forma mais simples, a obtenção de valores intermédios para o tamanho de pastas contidas na pasta raiz.

```
INICIAR a pilha de pastas
LISTAR conteúdo da pasta inicial
PARA CADA ficheiro
    SE ficheiro é uma pasta ENTÃO
        adicionar à pilha
    SENÃO
        ler e guardar dados
FIM

ENQUANTO a pilha NÃO estiver vazia e operação NÃO foi abortada
    OBTER pasta da pilha
    ler e guardar dados
    listar conteúdo da pasta
    PARA CADA ficheiro
        SE ficheiro é uma pasta ENTÃO
            adicionar à pilha
        SENÃO
            ler e guardar dados
FIM
```

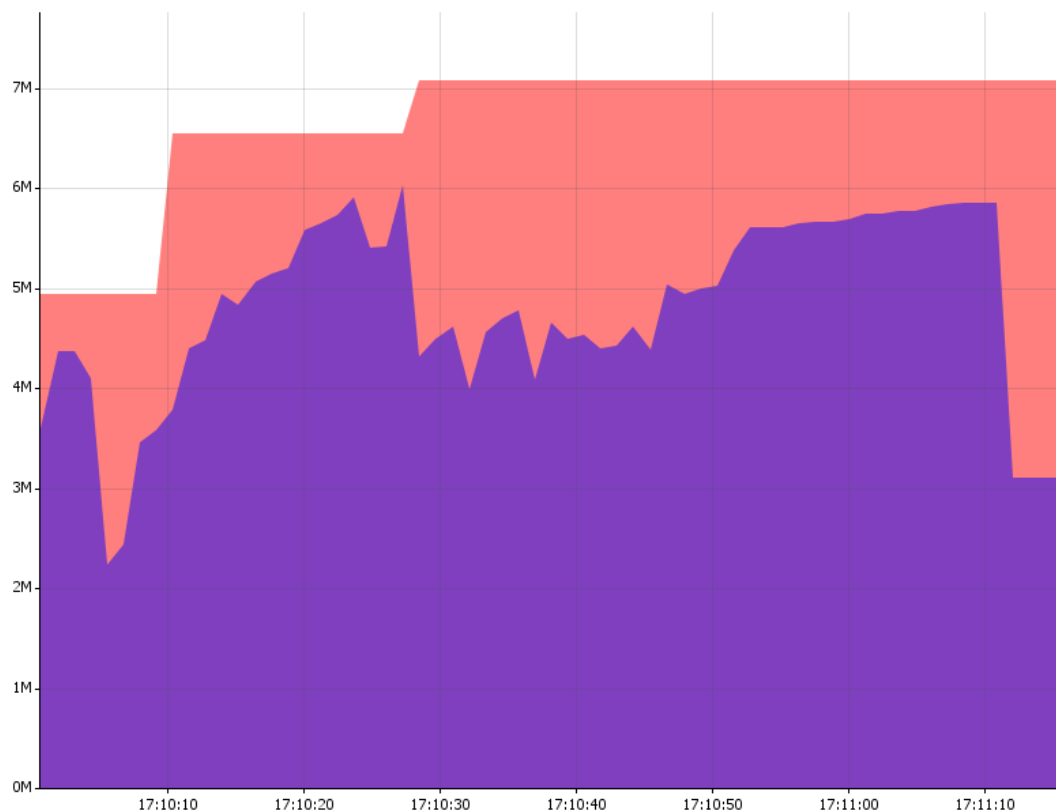


Figura 7. Evolução do consumo de memória com algoritmo iterativo

Embora a substituição por um algoritmo iterativo, tenha, em grande parte, resolvido o problema de consumo de memória, há que ter em consideração que o mesmo, por estar em testes, não se encontra a efectuar todas as operações que serão necessárias, como a determinação da relação entre diferentes pastas, tradução dos dados para uma representação que tenha significado para o sistema, entre outros. Não se encontra, também, incluído na plataforma RCP. Estas condições provocarão, certamente, alterações nos resultados finais.

Excepto no acima referido, foi possível ler todas as informações pretendidas. É de referir, no entanto, que o desenvolvimento de actividades paralelas, de adequação às tecnologias, provocou atrasos significativos que foram propagados para as iterações seguintes.

5.4 Iteração 2

Nesta iteração foram determinados, o modelo de dados e as funcionalidades de gestão e organização da informação, de acordo com as *user stories* e foram criadas as regras de código, necessárias à correcta aplicação do modelo de Código Colaborativo e disponíveis em anexo.

Foi também desenvolvida a parte da interface gráfica que permitia o acesso às funcionalidades criadas, tendo por base, um dos sistemas já existentes no mercado[43], Figura 8, e foi feita a escolha da tecnologia de persistência e a configuração de uma base de dados para testes.

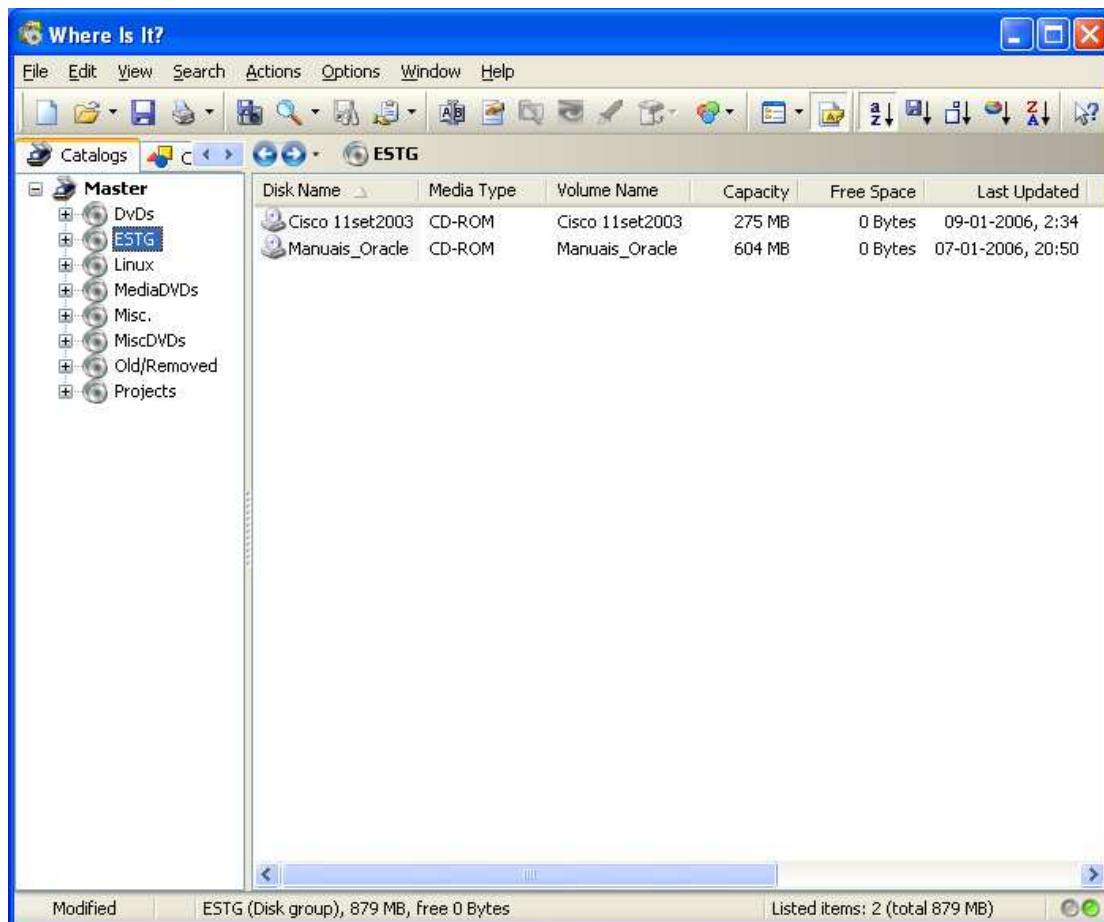


Figura 8. Sistema Where Is It™

A nível de domínio de informação, considera-se um catálogo o contentor de todos os dados, elemento que guarda os discos e pastas lidos, as suas informações e as do seu conteúdo.

Os catálogos são guardados usando bases de dados relacionais, sendo o nome do catálogo usado para o nome da base de dados. Foi vantajoso usar uma tecnologia ORM para diminuir a distância entre os objectos usados e a estrutura da base de dados, bem como remover da equipa de programadores as preocupações que o mapeamento entre objectos e as tabelas correspondentes impõe.

Além do nome do catálogo, serão guardadas informações da data de criação, uma descrição, o número total de discos catalogados e de pastas e ficheiros existentes.

Um disco, elemento que pode ter associado vários ficheiros, corresponde a uma *drive* ligada ao computador, a uma pasta, ou, a uma localização na rede, e pode conter ficheiros cujas descrições poderão ser importadas. Os discos podem ser agrupados em grupos de discos e estes podem conter outros grupos de discos.

Os discos contêm o número automático atribuído pelo programa, o tipo de suporte, se é DVD-ROM, CD-ROM, HDD, FOLDER, FLOPPY, um nome definido pelo utilizador, a localização física do disco, as etiquetas a que pertence, o espaço livre e o espaço ocupado e uma descrição.

Uma etiqueta é apenas uma designação que representa uma forma de organizar determinado disco ou grupo de discos. Da análise da informação, concluiu-se que etiquetas e categorias não possuem diferenças práticas, catalogar por categorias ou por etiquetas produz o mesmo resultado. Dessa forma, podemos considerar que categorias são um caso especial de etiquetas.

De um grupo de discos é guardado o nome, o número de discos associados, a capacidade total oferecida pelos discos, o espaço livre total que existe nos discos, o número total de pastas e de ficheiros, a lista de etiquetas atribuídas e uma descrição do grupo.

Com base nas descrições apresentadas acima, foi determinado o modelo de domínio, Figura 9, que indica as entidades criadas e as suas relações.

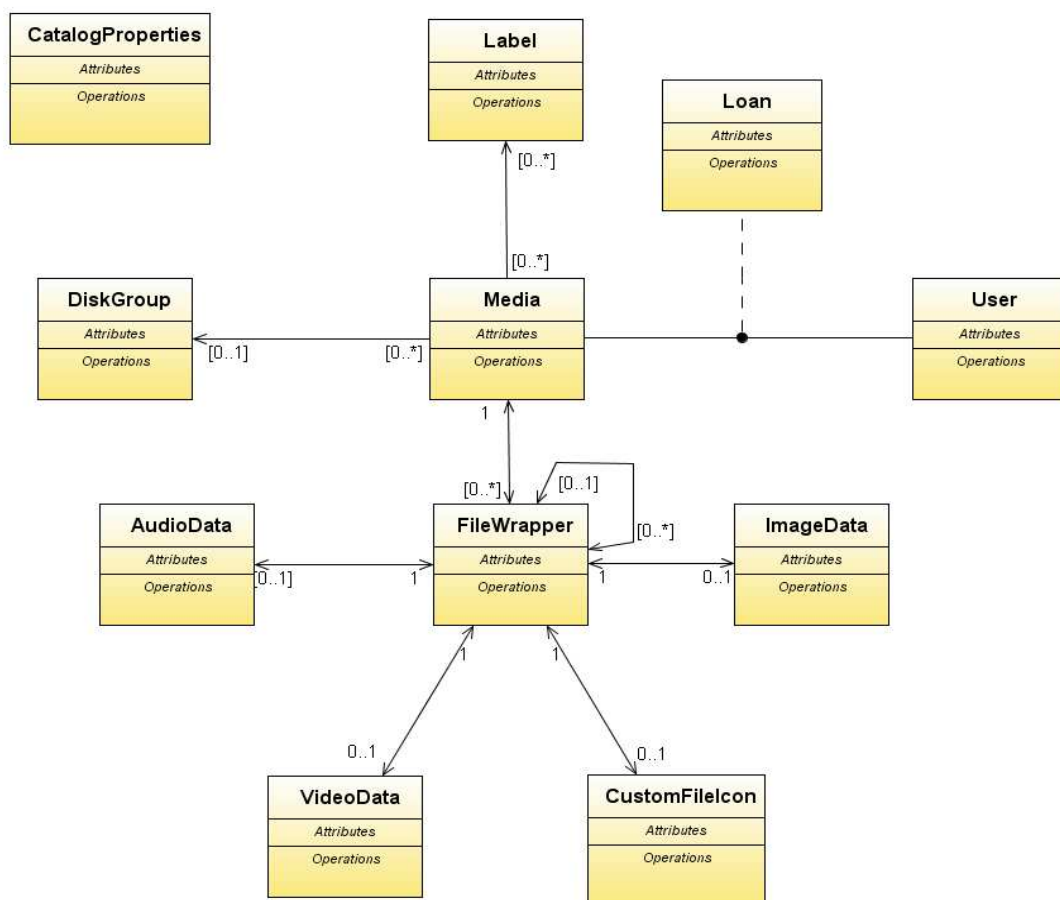


Figura 9. Modelo de domínio.

Um catálogo, ao ser apresentado na interface gráfica, possui, como elemento raiz, um nó com o nome do catálogo. Este é o elemento no qual todos os restantes elementos, sejam grupos ou apenas discos, podem ser colocados.

Na criação de um novo catálogo será necessário pedir o nome a dar ao catálogo, sendo que os dados de ligação à base de dados devem já estar preenchidos, usando os que tenham sido configurados anteriormente no painel de opções. No entanto deverá ser possível alterar os mesmos dados na janela de criação do catálogo.

Para efectuar uma ligação à base de dados é necessário indicar o URL para a ligação, o porto onde o servidor deverá ser contactado, o nome de utilizador, palavra-chave de acesso e o tipo de motor de bases de dados a usar.

5.4.1 Avaliação

Esta iteração passou o tempo previsto por quase três semanas. Foram encontradas várias dificuldades na correcta implementação dos métodos e objectos, para que fosse possível à componente visual ter acesso aos dados a mostrar.

Existiram dificuldades na criação do sistema de persistência, especificamente na criação dos ficheiros de mapeamento da tecnologia Hibernate, e na integração das bibliotecas do mesmo sistema com os restantes módulos e plataforma RCP.

Não foram encontrados tutoriais ou outra documentação sobre o assunto e, consultada a lista de discussão sobre desenvolvimento na plataforma RCP NetBeans, descobriu-se que apenas uma minoria de programadores tinham implementado ou estavam a implementar uma solução de *software* que envolvesse a integração do sistema de persistência, que usasse, como apoio, um motor de bases de dados relacionais, e até, que usasse uma tecnologia ORM.

Na componente visual, criar e mostrar nós que representassem os dados, Figura 10, foi também um processo demorado, tendo sido um dos pontos que mais pesou no atraso da iteração.

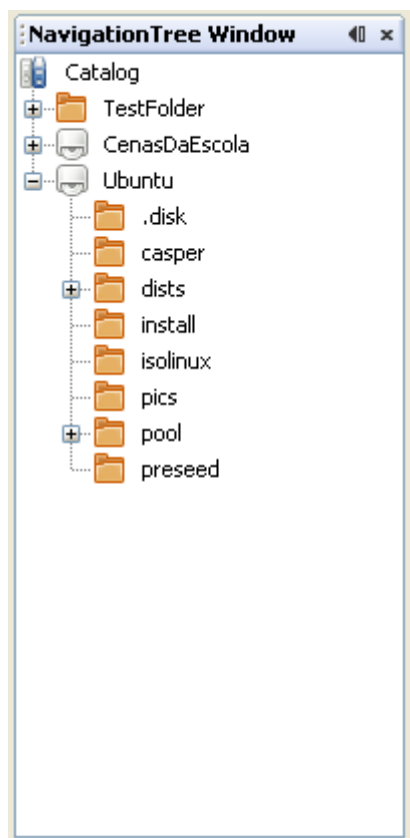


Figura 10. Representação visual dos dados

Uma vez que os dados são obtidos através de uma ligação a uma base de dados é um processo demorado, foi necessário implementar a criação dos nós^{viii}, que representam os dados, de forma assíncrona. Neste ponto, a API da plataforma referente à utilização de nós, está bastante virada para o uso dos mesmos para a representação de ficheiros, como pode ser visto na Figura 11, ou vistas de ficheiros como é o caso dos ficheiros JAR que contém no seu interior outros ficheiros[49]. Colocar essa API a representar dados que não estão presentes em disco, e que além de demorarem a ser obtidos, não podem ser facilmente manipulados como qualquer outro ficheiro, levou a que muito código, que de outra forma seria criado automaticamente, tivesse de ser criado manualmente. Muito desse código não era comum nos tutoriais e na documentação consultada e algumas tarefas não foram atingidas com sucesso, estando presentes, vários *bugs*^{ix}, para os quais não foi encontrada solução.

^{viii} Um Nó é o elemento de topo na *Nodes And Explorer API*. Um Nó não contém os dados mas é uma representação, acessível ao utilizador, de dados da aplicação, tipicamente ficheiros, embora a API possa ser usada para representar qualquer tipo de dados[49].

^{ix} Bug é termo usado para indicar uma falha ou defeito num sistema de computador.

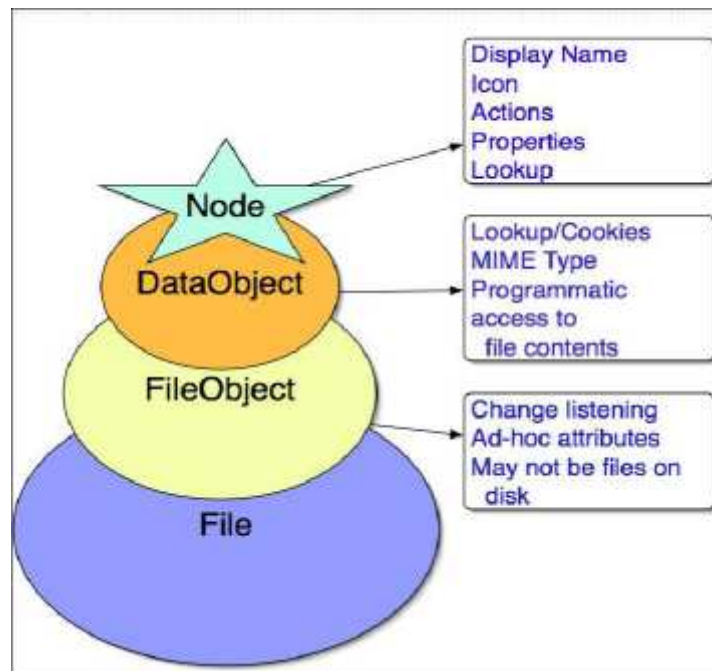


Figura 11. Representação da Nodes And Explorer API[45]

A funcionalidade de catalogar por etiquetas, embora contemplada no modelo de domínio e no código das entidades, não possui elementos de interface com o utilizador de forma a poder ser usada.

Considerou-se que a iteração não obteve o sucesso desejado, ficando por implementar o código que permitiria ver o conteúdo dos nós seleccionados, e que permitisse que alterações à base de dados fossem reflectidas na interface gráfica.

5.5 Iteração 3

O objectivo da terceira iteração foi a criação de uma API que permitisse implementar o sistema de extensões pretendido. Esta API faz parte do módulo que permite a leitura e introdução das informações na base de dados.

A API foi conseguida através de um conjunto de *interfaces*, que fornecem um contrato a cumprir pelas classes que adicionam novas descrições, e através de um conjunto de classes de ajuda, que permitam verificar quais as extensões existentes, quais as que se encontram activas, a que tipos de ficheiro corresponde cada extensão, e que permitem fornecer métodos de acesso à interface gráfica com o utilizador.

Consideraram-se como dados mais importantes, que podem ser fornecidos pelo sistema de extensões, as descrições textuais do ficheiro, não acessíveis através da API da plataforma

Java, e pequenos excertos do ficheiro, por exemplo, excertos de vídeo, miniaturas de imagens, entre outros.

Foram criadas algumas extensões, de forma a testar a API, que serão distribuídas juntamente com a aplicação, mas o utilizador poderá acrescentar extensões próprias. Para tal, o sistema irá pesquisar, em localizações pré-definidas, ficheiros Java compilados que implementem o contacto definido nas interfaces da API.

É dada oportunidade ao utilizador para definir onde pretende que as extensões sejam procuradas. Daqui se depreende que as extensões são definidas para um utilizador apenas, e não serão, por omissão, partilhadas por vários utilizadores com acesso ao mesmo computador. Uma vez que a aplicação não tem presente o conceito de vários utilizadores no mesmo computador, caso se pretenda a partilha de extensões entre utilizadores, essa partilha terá de ser configurada manualmente, usando as opções da aplicação.

A API permite que uma extensão do sistema de leitura forneça informações chave como o tipo de ficheiro que irá ler, o nome da extensão, a versão e o nome do autor. A identificação do tipo de ficheiro é feita através da extensão do ficheiro em causa.

Foi desenvolvida a interface gráfica que permite saber que extensões são reconhecidas pela aplicação, qual o seu estado, (activo/inactivo), e que possibilita a alteração desse estado.

5.5.1 Avaliação

Esta iteração revelou-se bastante mais simples que o inicialmente pensado. Muito devido ao facto de, após avaliação das *user stories* e a experiência que a equipa de programação tinha com o *software* de referência, se verificar que seria suficiente restringir o tipo de informação que é guardada.

O sistema de extensões permite criar extensões que obtenham informação sobre ficheiros na forma de excertos de áudio, excertos de vídeo, imagens e descrições textuais do ficheiro. Permite também que seja alterado pela extensão, o ícone a usar, substituindo assim o ícone fornecido pela aplicação.

Apesar de implementada com sucesso, foram detectadas duas limitações no desenho da API: verifica-se que não são permitidas duas extensões diferentes para o mesmo tipo de ficheiro, isto é, se duas extensões se registarem como fornecedoras de informações sobre ficheiros JPG, apenas a segunda será utilizada, não sendo a primeira, sequer, visível ao utilizador, e, uma vez que o código das extensões é executado sem qualquer validação de segurança, é possível que uma extensão que contenha código malicioso possa ser executada livremente,

por exemplo, código que apague ficheiros do computador onde a aplicação está a ser utilizada.

De seguida são apresentadas as *packages* que compõem o sistema de leitura de informações e a API de extensões incluída. A *package* “*readingfiles*”, Figura 12, representa o sistema de leitura de ficheiros, com as suas classes de apoio, e a relação com as outras duas *packages*. A *package* “*pluginapi*”, Figura 13, contém as classes e interfaces necessárias ao sistema de extensões e, finalmente, a *package* “*plugins*”, Figura 14, inclui algumas implementações de extensões que foram usadas para testar a API.

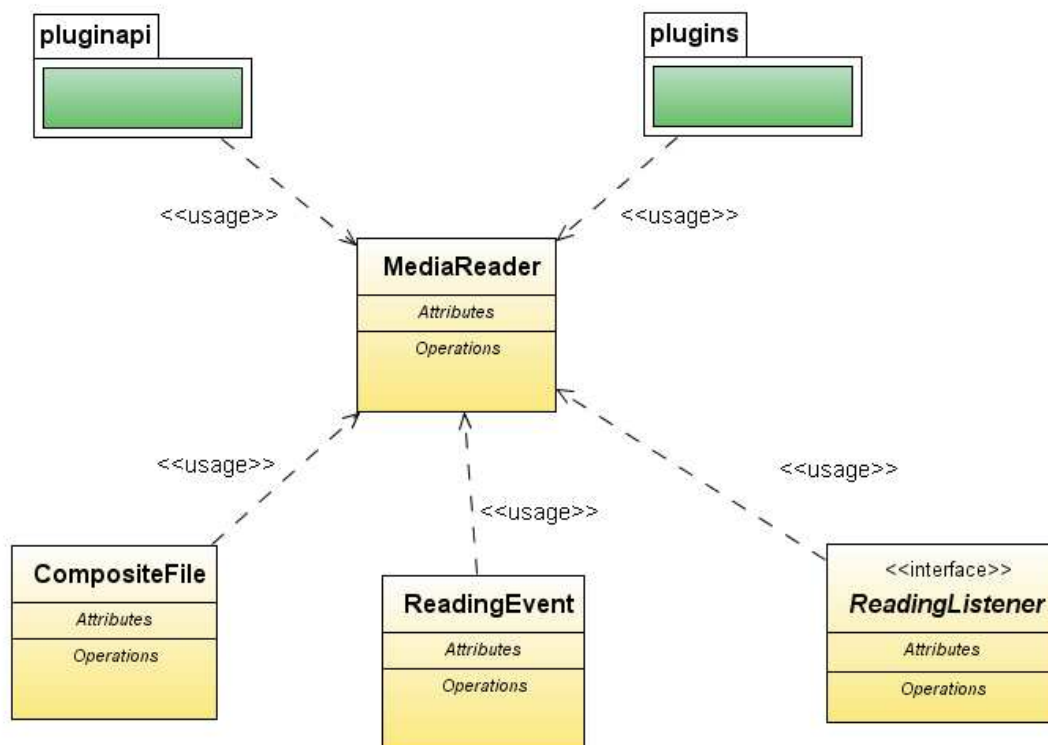


Figura 12. Package “*readingfiles*”

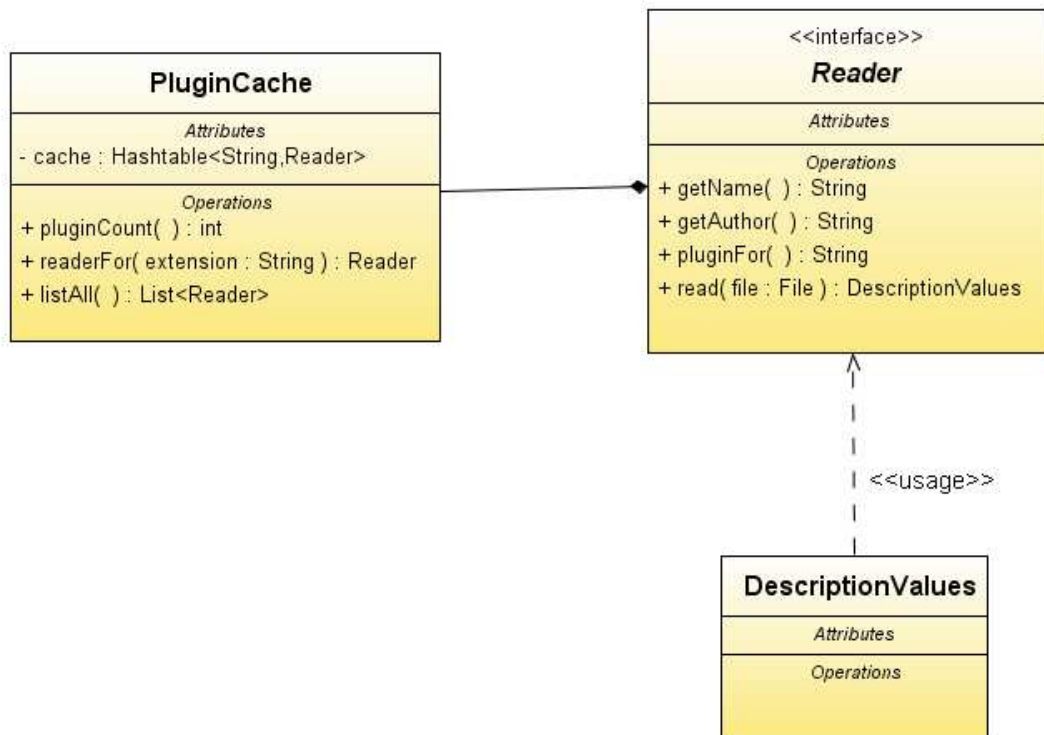


Figura 13. Package “pluginapi”

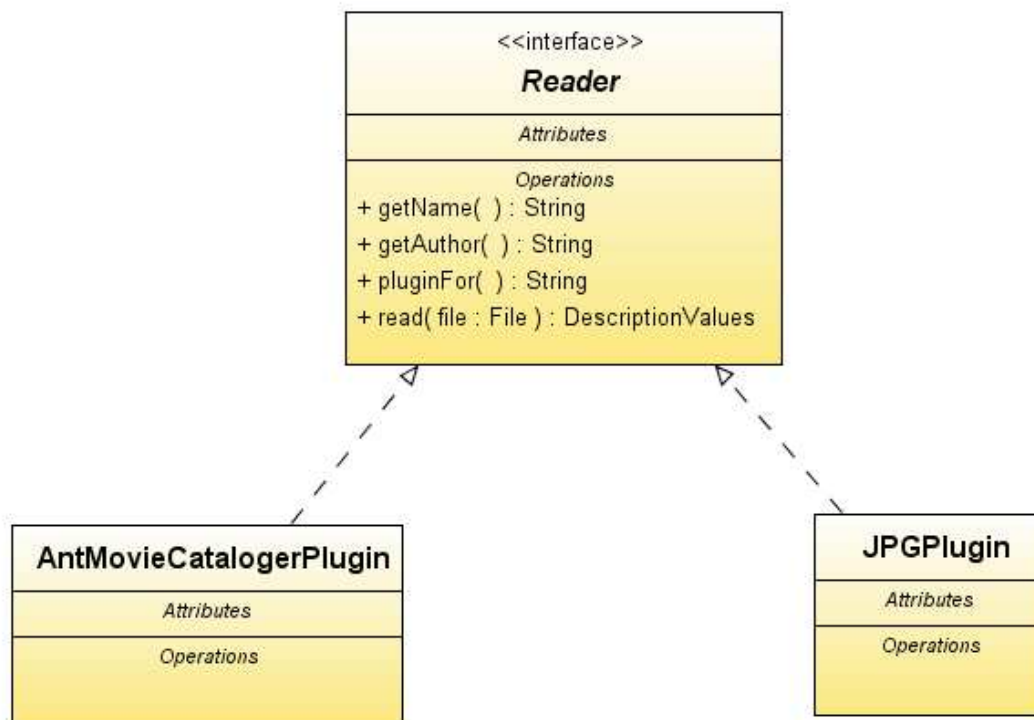


Figura 14. Package “plugins”

5.6 Iteração 4

Os objectivos da quarta iteração foram:

- A alteração ao modelo de dados de forma a permitir guardar a localização física do ficheiro no disco de origem;
- Implementação de mecanismos de pesquisa para localizar informações no catálogo, possibilitando o uso de expressões regulares, da definição do âmbito de pesquisa, da pesquisa nas descrições, além de nos nomes dos ficheiros, e de se fazer a distinção entre maiúsculas e minúsculas durante a pesquisa;
- Implementar um mecanismo que permita abrir os ficheiros, cujo suporte esteja disponível, nas suas aplicações pré-definidas. Por exemplo, abrir um documento HTML no *browser* definido do utilizador, ou um ficheiro PDF no Adobe Acrobat Reader.
- Implementar elementos de interface gráfica e módulos que permitam escolher o tipo de motor de base de dados a usar para guardar o catálogo;
- Implementar um sistema que permita identificar ficheiros duplicados no catálogo.

Na avaliação dos objectivos verificou-se que o requisito de pesquisas com opção de expressões regulares, seria limitado pela falta de suporte a esse tipo de pesquisas, tanto na linguagem SQL como no sistema Hibernate.

Para identificar ficheiros duplicados foi usado o código identificativo do ficheiro, já contemplado no modelo de domínio, dado esse ser o único campo que permite, com alguma segurança, determinar a identidade de um ficheiro. Para gerar esse código recorreu-se a um algoritmo de *hash* e, uma vez que existem vários algoritmos diferentes, a existência de bibliotecas que os implementam é, também, variada. Assim, após uma breve pesquisa, foi escolhida a biblioteca JackSum, pela sua simplicidade de utilização, diversidade de algoritmos oferecidos, licença *Open Source*, e principalmente por ser completamente implementada em Java, o que, em oposição à outra biblioteca avaliada, a FastMD5, garante o requisito de que a aplicação será multi-plataforma.

O algoritmo escolhido foi o SHA-1, devido ao bom compromisso entre unicidade dos códigos gerados e a penalização de performance associada que oferece.

5.6.1 Avaliação

Não foi possível resolver o problema das pesquisas com expressões regulares. Uma vez que se pretende independência do motor de bases de dados que apoia o sistema de persistência, não poderá ser usada nenhuma funcionalidade da linguagem SQL que não seja *standard*. Assim, não foi possível tirar partido do suporte para expressões regulares que alguns motores de bases de dados oferecem.

A introdução do algoritmo de cálculo do código *hash* provocou um aumento da memória consumida e do tempo necessário para efectuar a leitura dos dados dos ficheiros. Quando o suporte a ser lido contém muitos ficheiros e pastas é possível que a memória, definida por omissão para a JVM, não seja suficiente para a execução do processo de leitura. Este problema pode ser ultrapassado se a aplicação for iniciada com uma quantidade de memória base superior ao valor por omissão, tipicamente 64MB.

O sistema que permite abrir/executar um ficheiro, está dependente do SO. Apesar de ter sido usada uma funcionalidade da plataforma Java, presente desde o Java 6, não existe nenhuma garantia que a funcionalidade esteja disponível em todos os sistemas operativos e plataformas.

5.7 Iteração 5

A quinta iteração compreendeu alterações à iteração com o utilizador de forma a garantir alguns dos pedidos feitos nas *user stories*, nomeadamente questões de menus, barras de ferramentas e menus de contexto.

Nas figuras seguintes é apresentado o aspecto da aplicação após alterações à plataforma RCP. A Figura 15 apresenta a janela principal, sem um catálogo aberto, a Figura 16 apresenta um dos menus da aplicação, após remoção de todos os elementos que a plataforma RCP oferece e que foram considerados desnecessários.

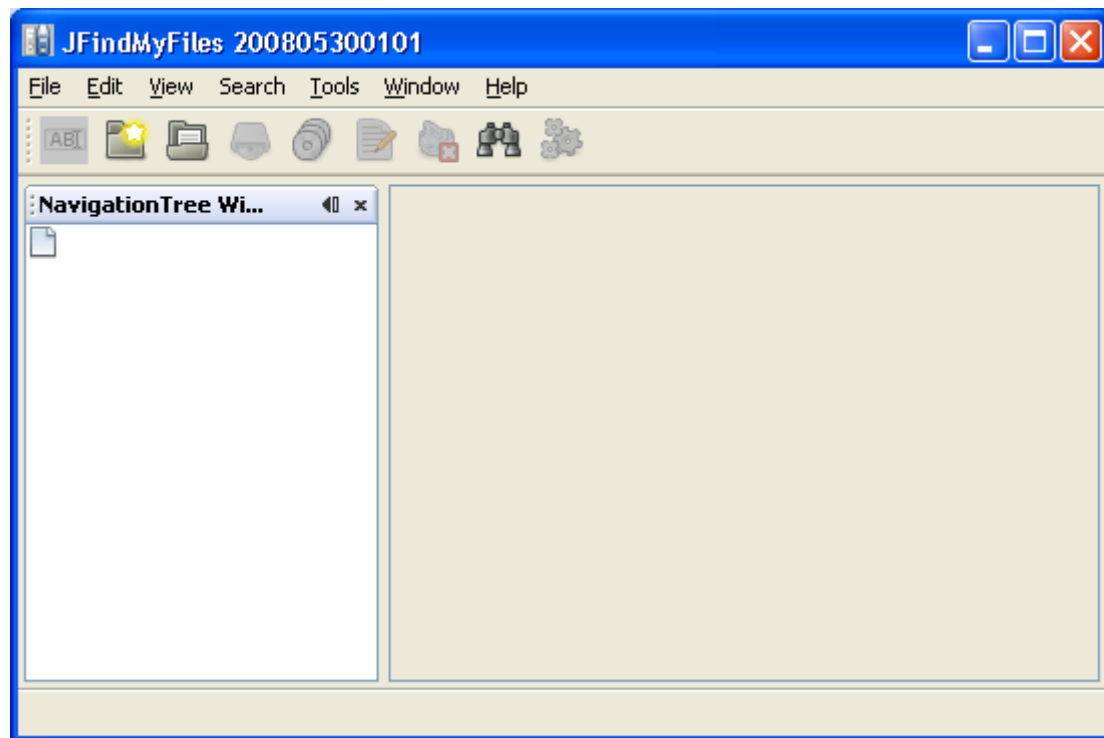


Figura 15. Sistema JFindMyFiles, ecrã principal.

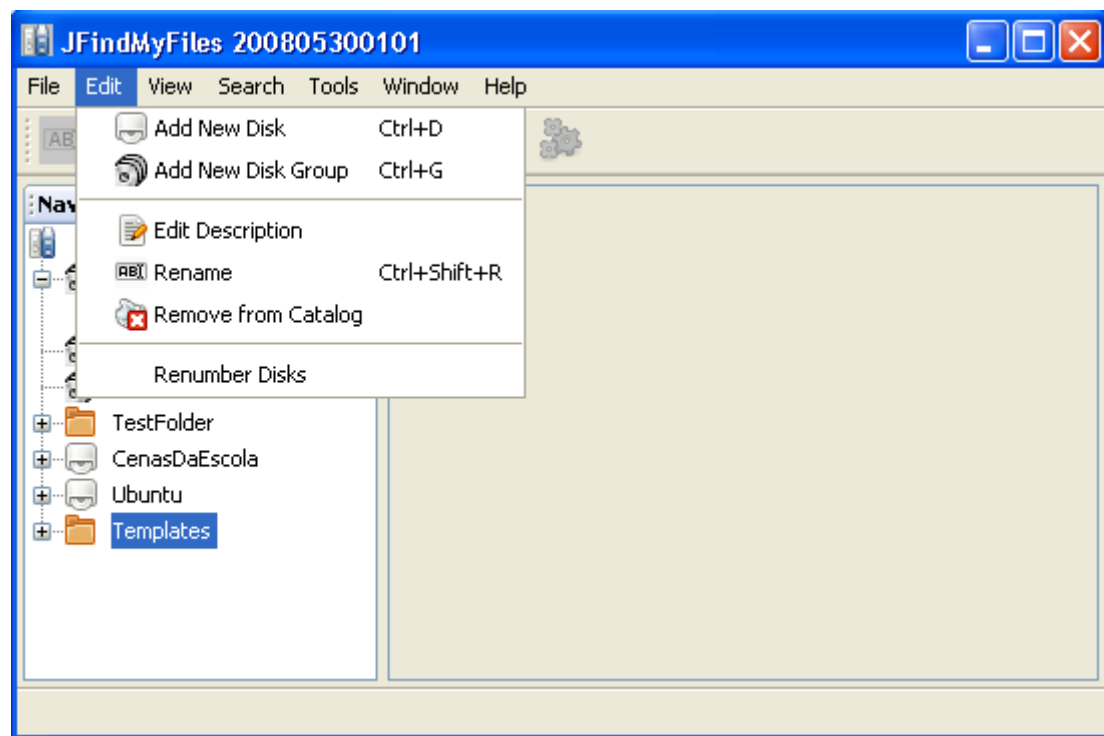


Figura 16. Sistema JFindMyFiles, menu de edição.

5.7.1 Avaliação

Embora esta iteração não tenha apresentado grandes dificuldades técnicas, algumas das escolhas efectuadas nas iterações anteriores, especificamente as que envolveram a criação de componentes de interacção, como menus, tanto os da barra de menus como os de contexto, dificultaram o desenvolvimento.

O insuficiente conhecimento sobre a plataforma, impediu que a equipa se tivesse apercebido que, apesar de os assistentes tornarem o desenvolvimento mais rápido, também faziam escolhas que nem sempre foram úteis. Neste caso, o tipo de acções^x associadas aos menus não era o tipo correcto e não permitia a partilha de acções já existentes entre menus diferentes que tivessem a mesma funcionalidade.

Várias classes tiveram de ser removidas e reescritas, foi necessário também editar o ficheiro XML que controlava as acções existentes o que provocou várias versões do sistema com erros que impediam a sua correcta utilização. Este ficheiro é criado automaticamente pelos assistentes do IDE e não existe documentação completa que pudesse ter sido consultada, apenas documentação referente a secções específicas.

5.8 Iteração 6

Dos vários formatos para os quais poderia ser possível exportar dados do sistema, os mais relevantes seriam os formatos que permitissem interoperabilidade com outras aplicações e sistemas operativos. Assim foram considerados os formatos: XML, que pelas suas características, nomeadamente por ser um formato de texto legível e portátil para qualquer outra plataforma, bem como de fácil leitura por sistemas informáticos, se apresenta como uma escolha natural; CSV, um formato largamente utilizado e suportado e que permite a fácil importação por folhas de cálculo; XLS e ODS, os formatos de folhas de cálculo do MS Excel e do OpenOffice Calc.

Tanto o formato XLS como o ODS exigem o uso de bibliotecas externas para a exportação de dados. A implementação deste formato no sistema de exportação foi considerada de baixa relevância pelo que a sua implementação foi relegada para o final da iteração.

Também com prioridade baixa, o sistema deverá permitir importar dados de ficheiros XML, exportados anteriormente e importar dados de outras aplicações, que deverão ser escolhidas,

^x Na plataforma NetBeans, uma acção de utilizador possui associada uma classe Java que implementa o código a executar, reage a selecções e outros gestos do utilizador. Existem vários tipos de acções, conforme a situação a que seja necessário reagir.

de entre as aplicações analisadas no início do desenvolvimento, tendo em conta as que se considerem mais relevantes. Na escolha dos sistemas deverão ser tidos em conta os seguintes critérios:

- Licença da aplicação;
- Quantidade de informação perdida ao ser efectuada a importação;

Na exportação para CSV foi decidido seguir o RFC4180[11], que sugere as regras a usar na criação deste formato de ficheiros.

5.8.1 Avaliação

A implementação do sistema de exportação para XML foi baseada na tecnologia SAX, existente na plataforma Java, que por ser um modelo baseado em eventos que não exige a representação, em memória, de toda a árvore XML, torna mais simples a exportação.

Foi também criado um *schema* XML, ficheiro XSD, que permitirá validar a estrutura dos ficheiros criados pelo mecanismo de exportação.

Para se seguirem as recomendações do RFC consultado, referente à criação de ficheiros CSV, foi necessário que, aquando da exportação, fosse criado, não um, mas vários ficheiros CSV, minorando assim a repetição de dados. Isto porque é necessário que os campos existentes numa linha do ficheiro sejam sempre fixos e iguais ao número de campos do cabeçalho. Para cumprir esta recomendação, e de forma a não criar demasiada informação repetida, decidiu-se separar os dados por 7 ficheiros. O primeiro ficheiro contém os dados gerais do catálogo, o segundo os dados referentes às etiquetas usadas, seguido dos grupos, discos, ficheiros lidos, lista de utilizadores e terminando com o ficheiro referente aos empréstimos.

5.9 Iteração 7

A sétima iteração compreendeu o desenvolvimento de um sistema que permitisse exportar os dados da aplicação, usando como base um ficheiro HTML.

Os *templates* são compostos por ficheiros HTML e recursos associados, que possuem marcadores específicos que indicam as áreas onde irão ser injectados os dados. Desta forma é possível exportar os dados da aplicação e personalizar o aspecto com que os mesmos são apresentados.

Uma lista completa dos marcadores criados encontra-se em anexo, no manual para o programador.

5.9.1 Avaliação

Esta iteração foi executada com sucesso, sem terem existido problemas técnicos ou dificuldades durante o seu desenvolvimento. Devido ao facto de ter sido, de todas as iterações, a que não dependeu do uso da plataforma RCP.

Foi criado um *template* simples, Figura 17, para testar o sistema.

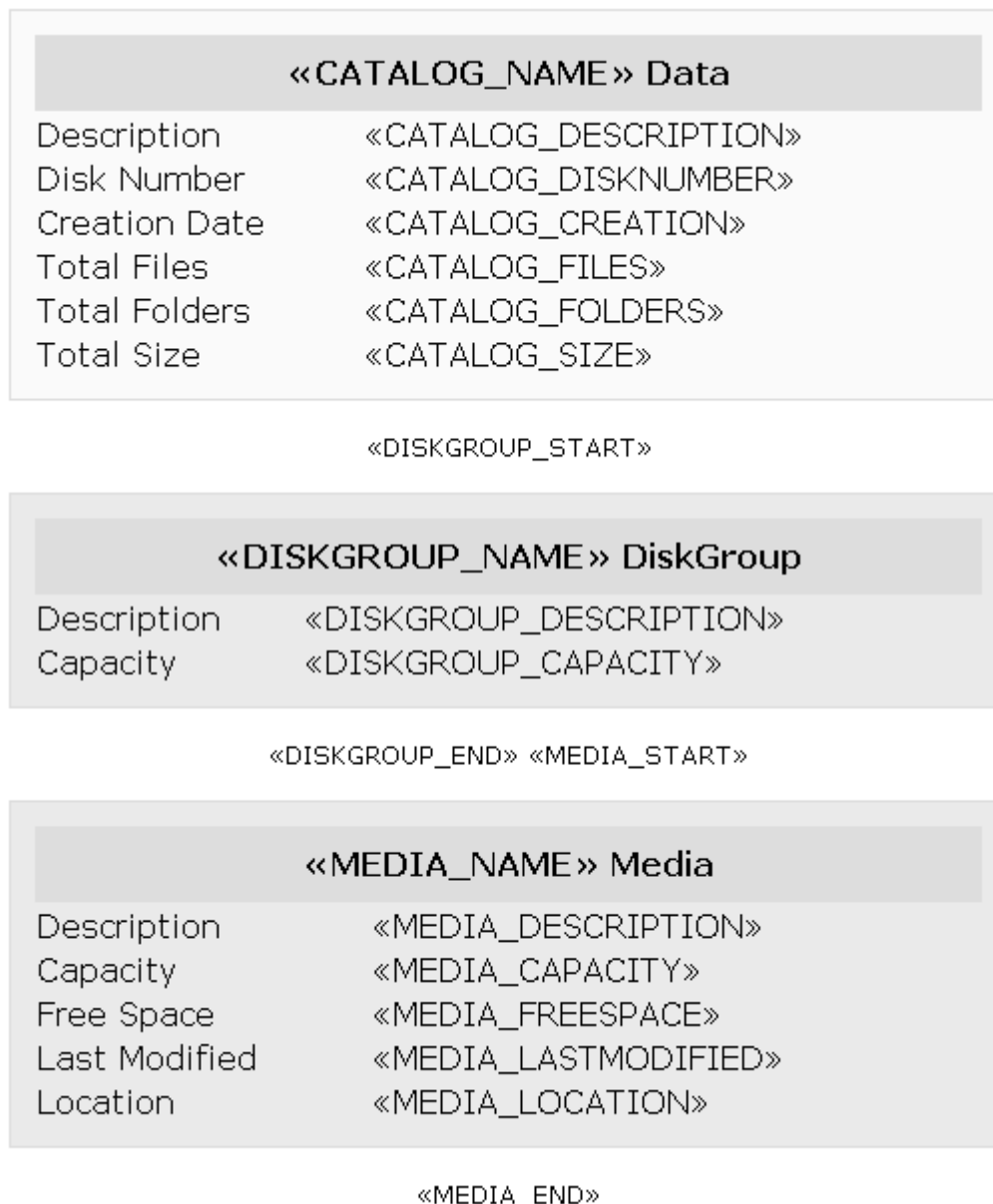


Figura 17. Template criado

5.10 Iterações 8 e 9

Devido aos atrasos verificados, e porque estas duas iterações compreendiam tarefas para a implementação de funcionalidades opcionais à aplicação, foi decidido que não seriam executadas, deixando os seus objectivos para desenvolvimentos futuros.

5.11 Spike Solutions

Uma *spike solution* pretende ser uma forma simples e rápida de resolver algumas dúvidas de carácter técnico que surjam durante a fase inicial do projecto ou durante as fases de análise de dada iteração.

5.11.1 CRC

Os testes desenvolvidos nesta *spike solution* permitiram verificar a validade e facilidade de se obter um código de *hash* a partir dos ficheiros lidos.

Determinou-se que seria mais simples e menos propício a erros, a utilização de uma biblioteca externa que fornecesse a funcionalidade desejada. Da pesquisa efectuada, foram encontradas duas bibliotecas candidatas, a Jacksum e a FastMD5. Uma vez que esta última recorre a código nativo, através de JNI, limita o número de plataformas, para a qual, a aplicação estará disponível, foi escolhida a biblioteca Jacksum.

Exemplo da criação de um código de *hash* MD5, usando a biblioteca FastMD5:

```
public void runnMD5() {
    try {
        System.out.println(MD5.asHex(MD5.getHash(new
File("C:\\System\\JFindMyFiles\\docs\\TODO"))));
    } catch (IOException ex) {
        //ignore
    }
}
```

Exemplo da criação de um código *hash* MD5, usando a biblioteca Jacksum:

```
public void runnMD5() {
    AbstractChecksum checksum = null;
    try {
        checksum = JacksumAPI.getChecksumInstance("md5");
    } catch (NoSuchAlgorithmException nsae) {
        // algorithm doesn't exist, ignore
    }
    // updates the checksum with the content of a file
    try {
        checksum.readFile("C:\\System\\JFindMyFiles\\docs\\TODO");
    } catch (IOException ioe) {
        .....//ignore
    }
    System.out.println(checksum);
}
```

Como se pode verificar, a utilização da biblioteca Jacksum, obriga à escrita de uma quantidade maior de código, no entanto, esta biblioteca possui como vantagens, o facto de permitir vários algoritmos, em vez de apenas o MD5 e, para além disso, proporciona um maior controlo de erros.

5.11.2 ExecutarFicheiro

Permitiu confirmar a possibilidade de abrir um ficheiro na sua aplicação pré-definida no SO.

Esta funcionalidade criou dúvidas pela fraca integração existentes entre a plataforma Java e o *desktop* dos vários sistemas operativos. Esta é uma operação dependente do SO onde a aplicação estará a correr, e até à versão 5 do Java, teria sido necessário recorrer a bibliotecas externas, possivelmente implementadas usando JNI, o que aumentaria a dependência entre a aplicação e o SO.

Após criado o código de teste verificou-se que, usando a versão 6 do Java, é possível, embora de forma ainda limitada, criar algumas funcionalidades comuns a aplicações *desktop*, nomeadamente a capacidade de abrir os ficheiros nas aplicações definidas no SO.

Excerto de código que permite abrir um documento na aplicação definida para esse efeito:

```
if(Desktop.isDesktopSupported()) {  
    Desktop d = Desktop.getDesktop();  
    try {  
        d.open(new File("C:\\\\System\\\\JFindMyFiles\\\\docs\\\\TODO.txt"));  
    } catch (IOException ex) {  
        //ignore  
    }  
}
```

5.11.3 PluginAPI

A criação de um sistema de extensões previa ser um dos pontos que mais dificuldades técnicas implicaria. Os testes de código preliminares revelaram uma dificuldade que, mais tarde se verificou errada.

Esta *spike solution* pretendeu resolver a questão e mostrar que as ideias iniciais estariam erradas. Com os resultados obtidos, o planeamento foi refeito de forma a ajustar melhor o tempo de desenvolvimento atribuído à terceira iteração.

Os resultados deste teste, foram a estrutura preliminar da API de extensões.

5.11.4 SimpleExport

Nesta *spike solution* foi testado o sistema de exportação para XML e CSV, tendo sido avaliadas as APIs de escrita de ficheiros XML presentes no Java.

Desta *spike solution* resultou o código, que foi incorporado no sistema, para atingir a funcionalidade de exportação de dados para XML e CSV.

5.11.5 SPKlerDados

Esta *spike solution* permitiu confirmar que informações, relativas aos ficheiros, poderiam ser lidas e permitiu testar alguns algoritmos possíveis de usar na obtenção das informações.

Como resultado foi criada uma pequena aplicação, Figura 18, que permite obter, entre outros, o tamanho dos ficheiros e pastas e pesquisar usando os algoritmos testados.

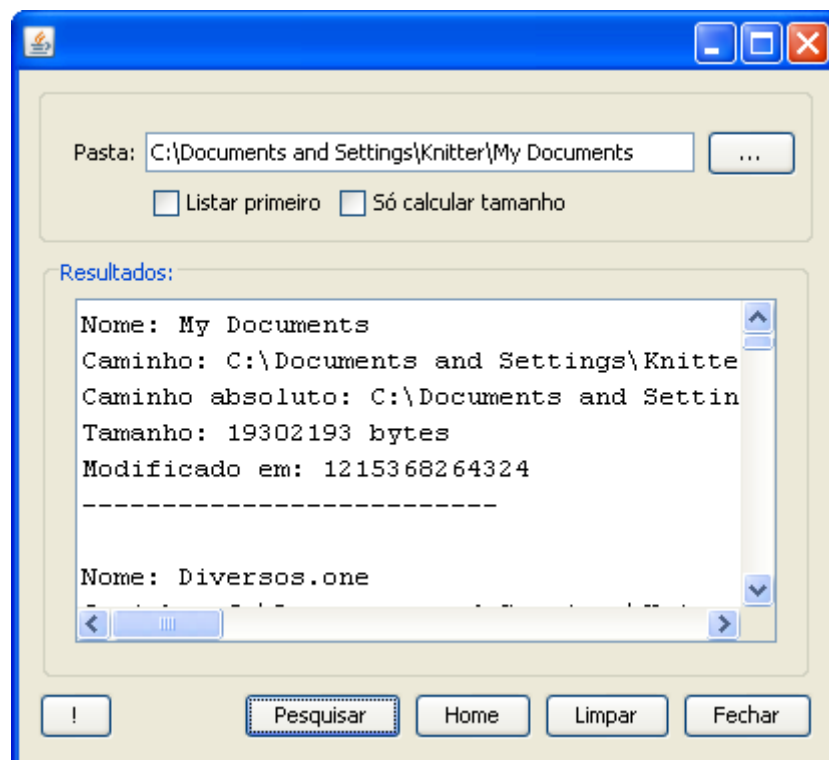


Figura 18. Aplicação de teste criada para a *spike solution* SPKlerDados

5.11.6 TemplateExport

Spike solution que pretendia avaliar a forma de criar um sistema de exportação que se baseasse num *template* HTML.

Nesta *spike solution* apenas foi criado código genérico que permitiu ter uma ideia de como estruturar a pesquisa e leitura dos ficheiros que compõem os *templates* e as informações necessárias para os mesmos, como título, recursos associados, autor, entre outros.

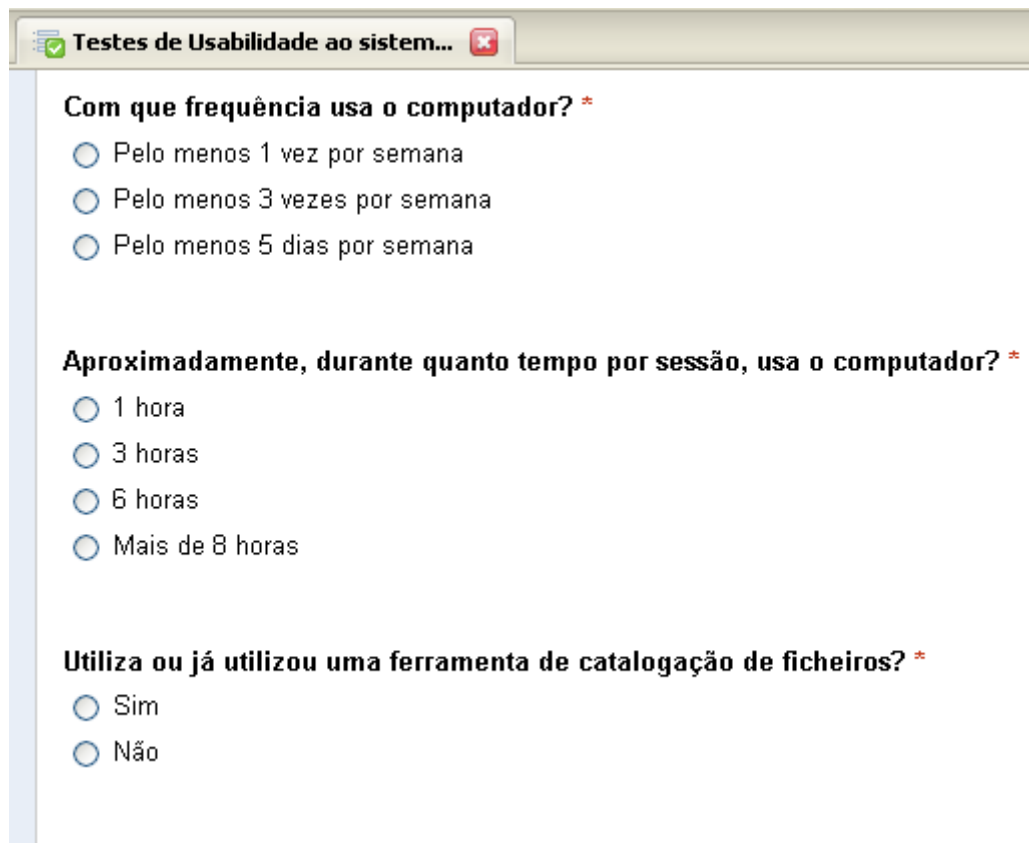
5.12 Testes de Usabilidade

Sendo usabilidade um termo demasiado vago, com várias definições que, por vezes se completam e noutras se opõem, é difícil definir correctamente o termo, no entanto, usabilidade pode ser vista como o quão satisfeitos estão os utilizadores com o sistema. Um sistema poderá ser complicado, mas ao não interferir de forma negativa, com o trabalho do utilizador, ser considerado um bom sistema[8].

Dessa forma, os testes de usabilidade pretendem dar a conhecer o grau de satisfação dos utilizadores e a facilidade com que estes usam o sistema. São um método fundamental de determinar os problemas que possam existir na interface do sistema.

No entanto a aplicação de testes de usabilidade acarreta diversos riscos, como a fiabilidade dos testes, isto é, saber se ao serem repetidos os testes são obtidos os mesmos resultados, a validade do testes, e a questão dos resultados reflectirem correctamente os problemas de usabilidade que se pretendem avaliar. Como resultado, uma avaliação de usabilidade, frequentemente, exige que o avaliador tome decisões sobre dados de confiança reduzida.

Para a avaliação das respostas e para apresentação do questionário, foi usada a funcionalidade do Google Docs, que permite a criação de formulários com questões, que podem ser posteriormente preenchidos através de um *browser*, Figura 19, e cujos resultados são, automaticamente, enviados para folhas de cálculo.



Testes de Usabilidade ao sistem...

Com que frequência usa o computador? *

- ☐ Pelo menos 1 vez por semana
- ☐ Pelo menos 3 vezes por semana
- ☐ Pelo menos 5 dias por semana

Aproximadamente, durante quanto tempo por sessão, usa o computador? *

- ☐ 1 hora
- ☐ 3 horas
- ☐ 6 horas
- ☐ Mais de 8 horas

Utiliza ou já utilizou uma ferramenta de catalogação de ficheiros? *

- ☐ Sim
- ☐ Não

Figura 19. Formulário de Teste de Usabilidade no sistema Google Docs

No anexo 8.7.2, são apresentados os resultados dos testes de usabilidade.

Os testes foram realizados em dois sistemas operativos diferentes, Windows XP e Ubuntu Linux, e a distribuição dos utilizadores pelos dois sistemas operativos foi aleatória.

Da análise dos testes, e da avaliação feita durante a execução dos mesmos, verificaram-se vários erros de usabilidade que deverão ser corrigidos.

As operações mais simples, como criar catálogos, abrir catálogos existentes, adicionar discos e trabalhar com os dados gerais da aplicação, foram executados sem problemas. No entanto, as acções referentes às opções da aplicação revelaram-se difíceis de executar por parte dos participantes.

Nenhum participante conseguiu alterar os dados referentes ao autor do catálogo, todos demonstraram alguma confusão entre os elementos de acesso ao sistema de extensões, presente na janela de opções, e o sistema de *plugins* proveniente da plataforma RCP NetBeans. Todos os utilizadores que testaram a aplicação acederam primeiro à janela de *plugins* da plataforma e não pensaram na existência da janela de opções para o sistema de extensões.

A funcionalidade de pesquisa de ficheiros duplicados provocou, também, alguma confusão, com cerca de metade dos utilizadores a tentarem executar a tarefa correspondente a partir da janela de pesquisa geral.

Dois utilizadores não tinham qualquer experiência com o tipo de aplicação e foi necessário explicar alguns dos objectivos e alguns conceitos, como o conceito de catalogação de discos. Estes utilizadores não tinham, também, noção do que seriam *mount points*, em sistemas operativos Linux, e na execução da última tarefa não entenderam o que era pedido.

5.13 Testes de unidade

Durante o desenvolvimento da aplicação, foram escritos vários testes de unidade, permitindo manter um controlo sobre a quantidade de *bugs* existentes. Os testes foram escritos e executados usando as ferramentas disponíveis no IDE, neste caso a plataforma de testes XTest, que permitem a execução de testes e análise dos resultados de forma simples, quer dentro do IDE, Figura 20, quer através da geração de relatórios e a sua apresentação em HTML, Figura 21

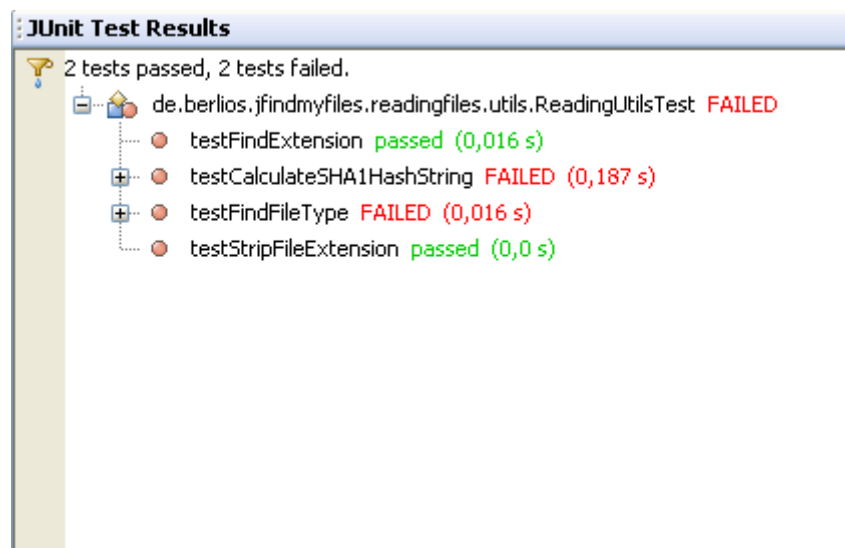


Figura 20. Resultados dos testes de unidade apresentados no IDE

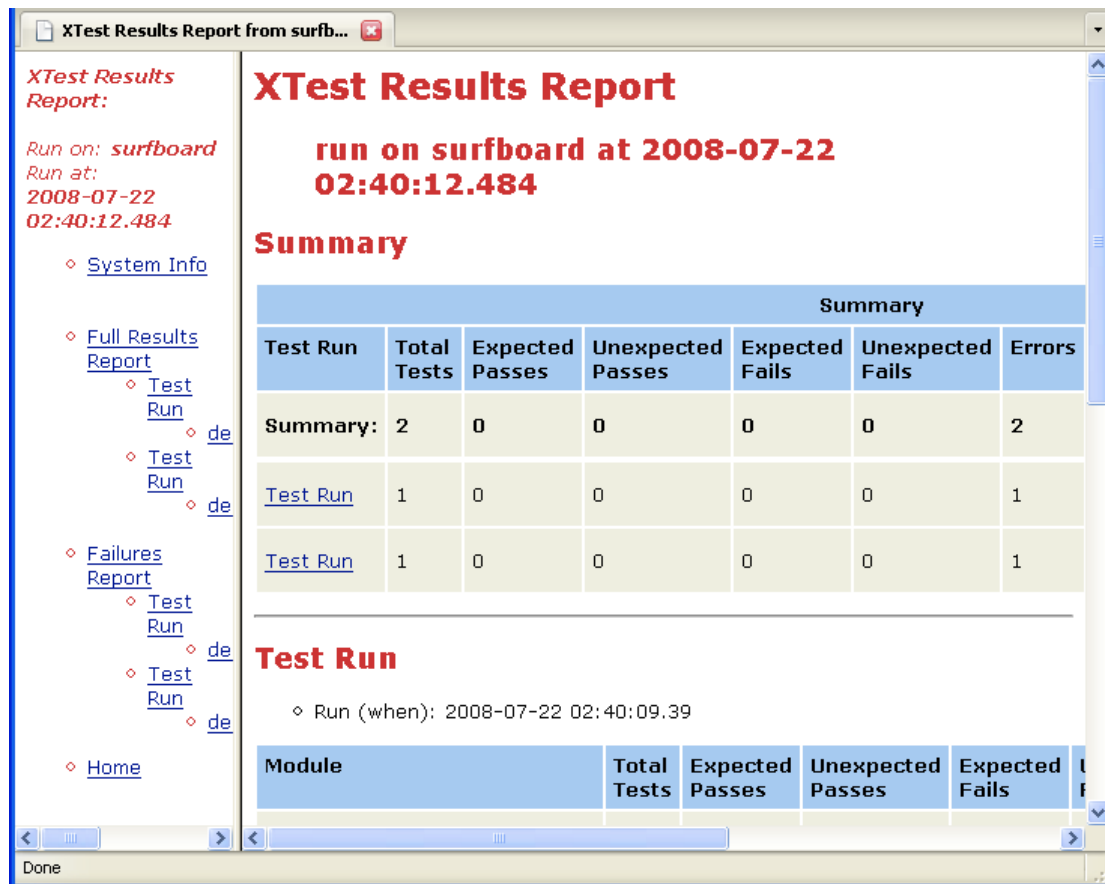


Figura 21. Resultados dos testes de unidade apresentados em HTML

Como exemplo, é apresentado o código de teste do método que permite obter a extensão de um ficheiro.

```
/**
 * Test of findExtension method, of class ReadingUtils.
 */
@Test
public void testFindExtension() {
    System.out.println("findExtension");
    String filename = "dados.txt";
    String expResult = "txt";
    String result = ReadingUtils.findExtension(filename);
    assertEquals(expResult, result);
}
```

5.14 Conclusões

Os objectivos iniciais foram atingidos em quase toda a sua totalidade, no entanto, alguns *bugs* persistem na aplicação, derivados principalmente das escolhas da equipa, que ao não possuir o conhecimento suficiente sobre a plataforma RCP usada, nem sempre decidiu pelas melhores implementações.

O desenvolvimento do projecto expôs a equipa a uma quantidade elevada de novas tecnologias e ferramentas, abrindo portas a novo conhecimento, e a uma visão diferente da programação de aplicações para computador pessoal. O uso da plataforma RCP escolhida e o inevitável contacto com alguns dos programadores que a produzem e/ou usam, mostrou à equipa um conjunto de técnicas, padrões, e formas de resolução de problemas, que o ambiente académico não proporciona.

Foi conseguido o desenvolvimento de uma aplicação multi-plataforma e de utilização livre, que possibilita assim que um utilizador possa aceder aos seus catálogos independentemente do SO que usa, e que numa empresa, independentemente do SO usado aquando a criação do catálogo, o mesmo pode ser acedido por outros computadores com outros sistemas operativos, oferecendo assim uma maior versatilidade e utilidade

O projecto, com origem nas necessidades pessoais dos elementos da equipa, e cuja continuação, aperfeiçoamento e divulgação são alguns dos objectivos a seguir após concluída o projecto, esteve sujeito a fortes restrições, impostas pelo curto tempo disponível à execução do projecto, mas permitiu, contudo foi criada uma aplicação funcional, com as características inicialmente pretendidas, e que serão melhoradas num futuro próximo.

6 Produto Criado

Nesta secção é apresentada uma visão geral da aplicação para que se perceba qual o resultado final. A Figura 22 exhibe a janela principal da aplicação durante a sua execução normal.

No anexo 8.8 é possível ver um conjunto alargado de imagens que permitem obter uma ideia mais precisa do produto resultante.

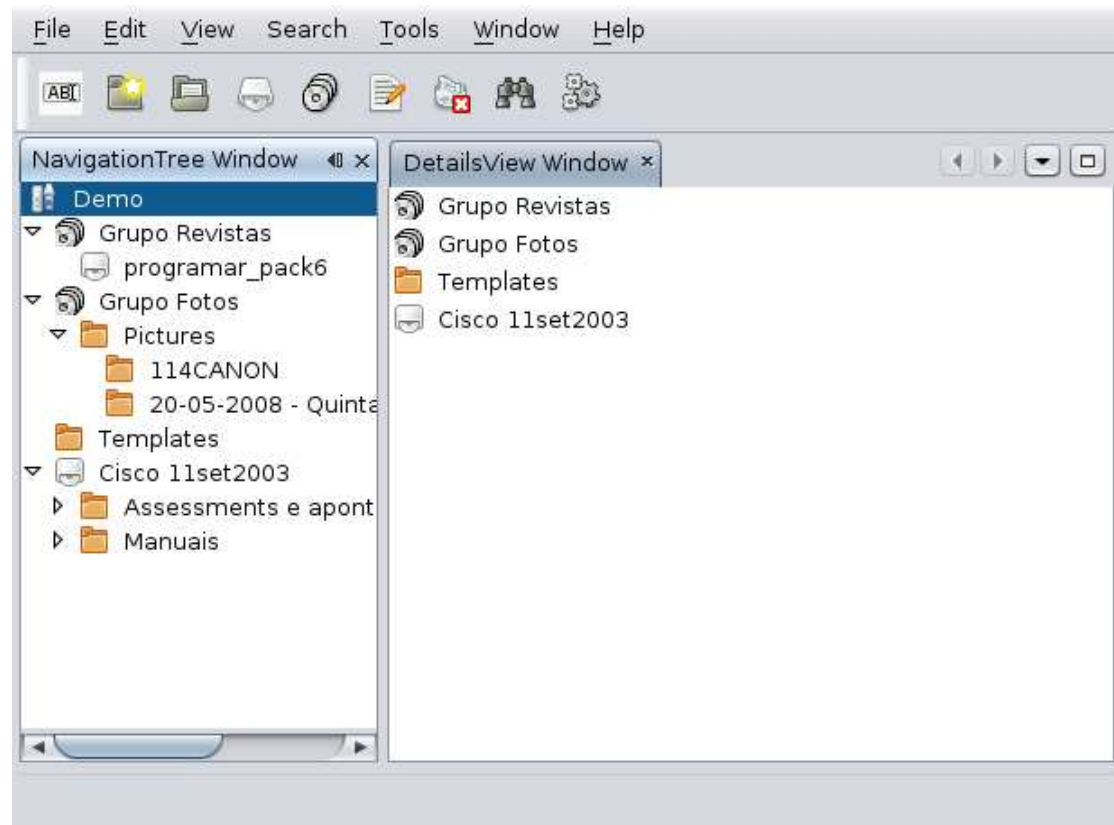


Figura 22. Sistema JFindMyFiles em execução

7 Desenvolvimentos Futuros

Devido ao facto de a iteração 8 ter ficado incompleta e a 9 não ter sido implementada, estas deverão ser concluídas e incorporadas na aplicação.

Deverão ser abordados os problemas da API de extensões, de forma a permitir mais que uma extensão para o mesmo ficheiro e garantir que as extensões executam num contexto seguro.

Actualmente, o código de persistência obriga à dependência da tecnologia ORM escolhida, neste caso ao uso de Hibernate. Em desenvolvimentos futuros, esse código deverá ser alterado, de forma a usar o *standard* de acesso a dados relacionais, JPA. Dessa forma será possível obter alguma independência do sistema ORM utilizado, podendo o mesmo, caso se verifique necessário, ser substituído por qualquer outra tecnologia que implemente a API JPA.

A interface gráfica deverá ser melhorada de acordo com o resultado dos testes de usabilidade.

8 Anexos

8.1 Estudo da Metodologia

Uma avaliação das metodologias existentes, objectiva e consciente dos requisitos e características do projecto, e da equipa de desenvolvimento, especialmente o tempo disponível para a execução do projecto e o tamanho da equipa, leva a que a escolha recaia sobre metodologias ágeis ou adaptadas de forma ágil e que as metodologias restantes sejam imediatamente descartadas.

A teoria por detrás de métodos ágeis é a de que se forem investidos meses a trabalhar toda a especificação do sistema no início do projecto, e depois for colocada em prática, isso não garantirá a qualidade do produto, satisfação do cliente e cumprimento dos prazos estipulados[10].

Os métodos ágeis respondem a esta questão de formas variadas, mas o princípio comum é o de dividir todo o projecto em pedaços mais pequenos e não decidir nada no início, especialmente a especificação[9].

A aplicação da metodologia ICONIX ao desenvolvimento ágil seria uma solução para se tirar partido das vantagens associadas a processos ágeis e ao mesmo tempo o rigor e segurança no desenvolvimento de *software* que uma metodologia, como a ICONIX oferece. No entanto assim, a aplicação da metodologia ICONIX ágil não é o mesmo que aplicar uma metodologia que tenha sido criada com a agilidade em mente, e não deixa de ter as regras e requisitos a nível de documentação e desenho de especificações, que tem quando usada sem adaptação às metodologias ágeis.

A opção da metodologia XP, por ser uma das mais conhecidas e faladas, surge como a que mais se adequa às características do projecto e características da equipa de desenvolvimento pelo que foi essa a seguida durante o processo de gestão e desenvolvimento do projecto.

8.2 Estudo da Plataforma RCP

No desenvolvimento aplicações *desktop*, existem várias tarefas que são repetidas de aplicação para aplicação, e que distraem o programador dos requisitos que o seu sistema possui[47]. A rápida prototipagem e avaliação de conceitos é também limitada por esse tipo de tarefas que, embora não estejam associadas ao problema a testar, são necessárias implementar para que a ideia possa ser correctamente avaliada.

Plataformas de desenvolvimento de aplicações ricas, RCP, são *frameworks* que permitem evitar tarefas repetitivas, comuns as todas as aplicações *desktop*. Estas *frameworks* fornecem serviços como gestão de janelas, menus, persistência definições, sistemas de actualizações automáticas, entre outras.

Inicialmente, foram consideradas quatro plataformas de desenvolvimento, a MFC[52][53], a Eclipse RCP[51], a Spring Rich[54] e a NetBeans Platform[47].

Dada a necessidade de uma aplicação multi-plataforma a plataforma MFC não poderá ser considerada como opção, uma vez que se destina, única e exclusivamente a sistemas operativos MS Windows. Se for acrescentado o facto de que, esta *framework* se afasta dos objectivos e funcionalidades oferecidas pelas restantes opções, aproximando-se mais de uma biblioteca de apoio ao desenvolvimento e não de uma plataforma RCP, facilmente se percebe que não é uma opção válida para o desenvolvimento do projecto.

A Spring RCP, é uma plataforma recente, implementada pela equipa que criou a plataforma de desenvolvimento Web para a tecnologia Java, Spring Framework. Sendo assim a Spring RCP revelou-se uma opção inviável.

Esta plataforma está ainda em desenvolvimento, com uma versão estável, de estatuto diferente de alpha e/ou beta, lançada a 17 de Março de 2008, e em fase de actualização, com código a ser reescrito, sendo prevista uma primeira versão, sob um novo rumo, para Outubro de 2008.

A documentação é escassa e baseada numa abordagem "document-as-needed", em que a documentação será criada conforme a necessidade dos programadores que usem a plataforma e a disponibilidade dos que a estão a desenvolver. Esta abordagem, impõe algumas restrições no acesso a informação relevante, em tempo útil.

Das quatro plataformas iniciais restam a Eclipse RCP e a NetBeans Platform, no entanto, a correcta avaliação das plataformas deveria ser feita recorrendo à implementação de um sistema comum nas duas alternativas concorrentes, de forma a ser possível comparar o impacto que as diferentes arquitecturas, funcionalidades e características, terão no desenvolvimento. Esta é, no entanto, uma opção não disponível devido ao curto tempo existente para desenvolver todo o sistema. Para contornar este problema, recorreu-se ao estudo feito por outro profissional da área, que desenvolveu uma aplicação nas duas plataformas e publicou o seu estudo na revista alemã *Eclipse Magazin*[6]. Após contactado o autor, foi possível obter uma versão em Inglês.

Estudado o artigo, concluí-se que ambas as plataformas possuem características similares, sendo os pormenores de implementação e arquitectura pouco relevantes para a escolha, isto é, através da comparação das capacidades técnicas das duas plataformas o resultado é um empate. Sendo assim, e tendo em conta os requisitos da aplicação efectuou-se uma comparação baseada nos seguintes aspectos:

- Disponibilidade de documentação oficial;
- Existência de exemplos práticos e exemplos criados por utilizadores da tecnologia;
- Existência de comunidade, que possa, nas situações em que a documentação se mostre insuficiente, responder às questões levantadas;
- IDEs que se podem usar no desenvolvimento da aplicação;
- Sistema gráfico utilizado pela plataforma;
- Portabilidade da plataforma;
- Conhecimento que os elementos do grupo possuem sobre a plataforma, os IDEs e o sistema gráfico imposto.

Na disponibilidade de documentação oficial e existência de exemplos práticos, ambas as plataformas mostraram possuir uma sólida documentação oficial, com bastantes exemplos e caso de uso sobre o desenvolvimento de aplicações ricas. A documentação existente é actualizada e facilmente acessível, sendo que, neste ponto, a plataforma NetBeans leva uma ligeira vantagem, na medida em que providencia pequenos tutoriais onde se desenvolvem aplicações completas sobre a plataforma e também pelo facto de existir um módulo para o IDE NetBeans que fornece acesso directo à documentação da API em modo *offline*.

As comunidades das duas plataformas são também bastante vastas, com listas de discussão, fóruns, e sistemas de *Wikis*^{XI}, onde as dúvidas poderão ser respondidas rapidamente.

A escolha da plataforma irá impor o tipo de IDE que se poderá usar, isto é, a escolha da plataforma Eclipse RCP irá condicionar à utilização do IDE Eclipse, e do mesmo modo, a escolha da plataforma NetBeans irá condicionar à utilização do IDE NetBeans. Nenhum outro IDE poderá ser usado se o objectivo é tirar partido das funcionalidades e vantagens que o uso

^{XI} Uma *Wiki* é uma colecção de páginas de Internet, desenhadas de forma a permitir o acesso livre, contribuição e alteração de conteúdo através de uma linguagem de marcação simplificada[5].

de uma plataforma deste género oferece. Embora seja teoricamente possível programar usando outros IDEs e outras ferramentas, na prática, o resultado será a perda de todas as facilidades existentes, como assistentes de geração automática de código, assistentes de criação de funcionalidades, assistentes de testes e sistemas de compilação automáticos.

A plataforma NetBeans usa o sistema gráfico Swing^{xii}, este sistema está disponível com todas as edições da plataforma Java e é o *standard* nesta tecnologia. Questões de velocidade que assombravam este sistema estão, actualmente, ultrapassadas, sendo que este sistema cumpre o requisito de portabilidade imposto. A plataforma Eclipse RCP usa o sistema gráfico SWT que, apesar de ser anunciado como um sistema nativo, rápido e multi-plataforma, mostrou alguns problemas que contradizem esse anúncio.

A tecnologia SWT não faz parte do *standard* da plataforma Java, e por esse mesmo motivo não está tão rapidamente disponível em novos sistemas. A utilização desta tecnologia irá também expor os programadores a *bugs* específicos da plataforma aumentando assim a complexidade de desenvolvimento.

Em suma, no que diz respeito à portabilidade da plataforma, a plataforma NetBeans não apresenta problemas de portabilidade além dos existentes na tecnologia Java. A plataforma Eclipse RCP, por seu lado, introduz alguns problemas relacionados com o sistema gráfico que usa.

Ambas as plataformas são desconhecidas para os programadores, embora estes tenham já alguma experiência com o sistema gráfico da plataforma NetBeans, o Swing, e com IDE NetBeans. O sistema gráfico da plataforma Eclipse RCP, o SWT, é completamente desconhecido para os programadores e o uso do IDE Eclipse também não é comum na equipa de desenvolvimento.

Em adição aos pontos anteriores, foi também considerada a facilidade e capacidade de integração de bibliotecas de componentes, criadas em Swing, com o resto da plataforma. O que se verificou foi que, devido a erros existentes no sistema SWT, a correcta integração de bibliotecas Swing com tecnologia SWT, só será possível em sistemas operativos MS Windows e GNU Linux, deixando de fora sistemas operativos Mac OS X. A plataforma NetBeans, por ser desenvolvida em Swing, não terá problemas de integração com outras bibliotecas, também desenvolvidas em Swing.

^{xii} O projecto Swing implementa um conjunto de componentes gráficos com aspecto personalizável. É uma biblioteca *standard* na tecnologia Java[4].

A conclusão a que se chegou, e tendo em conta todos os pontos apresentados, é que a plataforma NetBeans deverá ser escolhida para o desenvolvimento da aplicação. Esta é a plataforma que satisfaz todos os requisitos apresentados, e é aquela que reduz as dificuldades técnicas associadas ao projecto.

Além dos aspectos apresentados, foram também tidas em conta, embora apenas como referência pontual as opiniões deixadas por profissionais da área, que desenvolveram *software* usando as duas plataformas e que apresentaram os problemas encontrados durante os vários processos de desenvolvimento, na lista de discussão nusers@netbeans.org, sob o tópico "Choosing Netbeans platform or Eclipse RCP".

8.3 Sistema de Controlo de Versões

Dado que na prática de Código Colaborativo é fundamental o uso de um sistema de controlo de versões para a fácil integração, partilha e actualização do código desenvolvido, foi escolhido, de entre os disponíveis nos serviços oferecidos pelo sistema BerliOS, o uso de SVN como controlador de versões.

A escolha de SVN em detrimento de CVS prende-se com o facto de o primeiro ser uma evolução natural do CVS na tentativa de eliminação dos problemas que esse sistema apresenta. Por outro lado, os elementos da equipa possuíam maior conhecimento do sistema SVN, sendo uma vantagem o uso de um sistema já conhecido.

8.3.1 Clientes SVN

Dos vários clientes SVN existentes, preferiu-se a escolha por um cliente que não tivesse qualquer integração com o ambiente de desenvolvimento. Desta forma, garantimos que, além do código desenvolvido, é possível criar uma estrutura que organize coerentemente os documentos de projecto sem se ficar dependente dos mesmos serem visíveis ao IDE usado.

O cliente TortoiseSVN, cliente com interface gráfica, foi usado em sistemas operativos MS Windows, pela sua simplicidade de instalação e utilização, integração na shell do SO, e conjunto de funcionalidades extensas e completas. A resolução de conflitos e a visualização de diferenças está disponível no cliente, não sendo necessário, outro *software* de apoio.

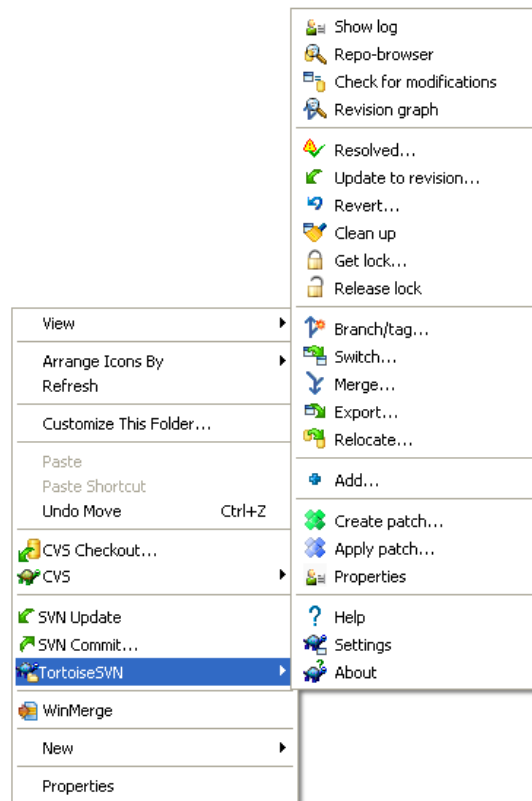


Figura 23. Integração do TortoiseSVN na shell do MS Windows XP

O segundo cliente usado, SmartSVN, provou ser o cliente mais completo, a nível de funcionalidades, e mais estável, que foi possível encontrar para sistemas operativos GNU Linux, especificamente para sistemas Ubuntu. Embora não seja um sistema *Open Source*, possui uma versão gratuita, apesar de limitada, que pode ser usada durante o desenvolvimento do projecto.

O facto de não se integrar facilmente com o SO provou criar algumas complicações, especialmente na detecção de alterações, adições e/ou remoção de ficheiros, bem como na resolução de conflitos que, por não existir um sistema de diferenças embutido, necessita de aplicações externas para essa funcionalidade.

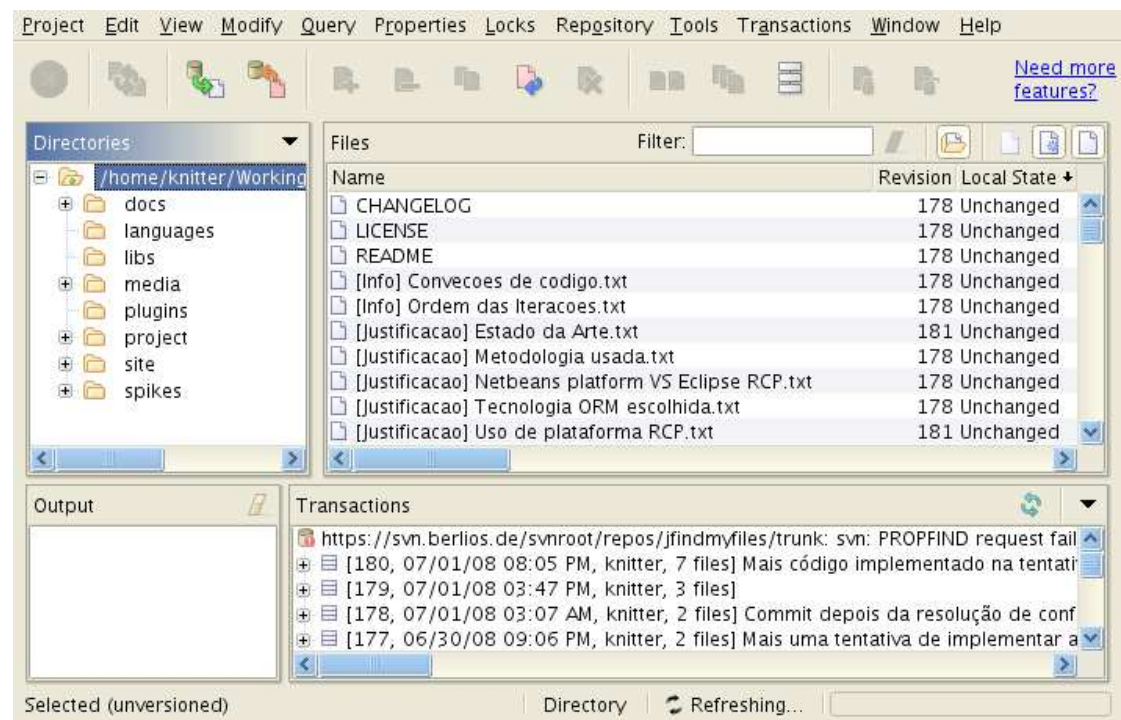


Figura 24. Janela principal do SmartSVN

8.4 Avaliação do Estado da Arte

Para conhecer a oferta existente no mercado, de forma a guiar as funcionalidades que foram implementadas, foi efectuada uma pesquisa por aplicativos concorrentes. Da pesquisa, resultou a recolha de informação sobre sete aplicações diferentes, cuja avaliação se apresenta de seguida, Tabela 1, e para a qual foram tidos em conta os aspectos que se consideraram relevantes, de acordo com os objectivos do projecto.

Tabela 1. Comparação entre aplicações

	Where Is It[43]	Argentum MyFiles[44]	cdCat[42]	DiskCatalogMaker[41]	CDFinder/CD Winder*[50]	Tellico[40]
Licença	Prop.**	Prop.**	GPL	Prop.**	Prop.**	GPL
Plataforma	Windows	Windows	Multi-plataforma, baseado em QT	Mac OS X	Mac OS/Windows	Linux
Permite pesquisas	Sim	Sim	Sim	Sim	Sim	Sim
Formato dos dados	Binário	Binário	XML	N/D	N/D	N/D
Extensível	Sim, sistema de extensões em Delphi	Não	Não	Não	Não	Sim, possibilidade de criação de <i>scripts</i>
Permite exportação	Sim, sistema de relatórios para exportar dados	Sim, sistema de <i>templates</i> HTML	Sim, para os formatos CSV, XML e HTML	N/D	Sim	Não
Permite importação	N/D	Não	Sim	N/D	Sim	Sim

*Duas versões da mesma aplicação, para sistemas operativos Mac OS e Windows, respectivamente.

**Licenciamento proprietário.

N/D, não se encontrava disponível no sítio da aplicação.

As aplicações escolhidas, apesar de serem apenas uma amostra do conjunto de aplicações existentes, são significativas no sentido em que demonstram a oferta existente actualmente. Na pesquisa não foram contempladas aplicações que se destinavam apenas a catalogar ficheiros de imagem e ficheiros de vídeo, uma vez que essa restrição vai contra os objectivos do projecto.

Embora a comparação apresentada tenha como alvo apenas sete aplicações, conseguiu-se verificar que o SO mais comum em computadores pessoais, possui uma disponibilidade maior de aplicações, não sendo essas aplicações compatíveis, e formato de ficheiro usado para persistência dos dados, com outros sistemas operativos.

A nível de funcionalidades, todas permitem a gestão de informações, a leitura dos vários dispositivos de armazenamento existentes no computador, bem como pesquisa de informação catalogada.

8.5 User Stories

8.5.1 Lista de User Stories

Segue a lista completa de *user stories* encontradas. As *user stories* apresentadas estão identificadas por um código, composto pelo prefixo ST, ao qual é adicionado o número da *user story*, desta forma identificando inequivocamente cada uma. A Tabela 2 lista as *user stories* encontradas.

Tabela 2. *User stories*

ST1	O programa tem de permitir que se possa indicar a pasta a ler e tem de ler o conteúdo dessa pasta e guardar as informações dos ficheiros.
ST2	Deve ser possível introduzir uma descrição para cada elemento através de uma janela que mostre as propriedades do ficheiro.
ST3	Quando se trata de um disco amovível deve existir a possibilidade de guardar a localização física do disco.
ST4	O programa tem de permitir criar catálogos, abrir catálogos, fechar e guardar catálogos.

ST5	Os discos e pastas podem ser catalogados usando etiquetas, o sistema deve ser independente das categorias e grupos de discos e permitir que o mesmo disco tenha mais que uma categoria.
ST6	É necessário encontrar um ficheiro no catálogo, para isso deve ser possível escolher a categoria, grupo de discos, ou todo o catálogo para restringir a pesquisa.
ST7	A aplicação deve permitir ser distribuída em mais que uma língua.
ST8	Sistema de estatísticas que mostre dados relevantes sobre os ficheiros existentes como o número de ficheiros de determinado tipo, o tamanho médio dos ficheiros catalogados, o tamanho total ocupado, etc.
ST9	Opções para permitir exportar os dados existentes para outros formatos, principalmente para formatos portáteis como XML e CSV.
ST10	Opções para importar dados de outras bases de dados criadas anteriormente com o programa e de outros formatos providenciados por programas concorrentes.
ST11	Dever ser possível exportar um catálogo completo ou um subconjunto para um conjunto de ficheiros HTML que possam ser colocados on-line ou distribuídos livremente. Devem existir personalizações de aspecto que se possam aplicar aos ficheiros criados.
ST12	A pesquisa de ficheiros deve conter opções para pesquisar usando expressões regulares.
ST13	As opções mais comuns devem aparecer numa barra acima da lista de ficheiros.
ST14	Deve existir um menu na árvore com a lista de categorias/grupo de discos/etiquetas com opções de pesquisa, introdução de descrições, alteração do nome, exportação e importação.

ST15	Se a localização do ficheiro estiver acessível deve ser possível abri-lo.
ST16	O sistema de leitura de informações de ficheiro tem de permitir ser expandido para ler ficheiros desconhecidos ou para que a forma como as informações são lidas seja alterada.
ST17	Deve ser possível configurar as extensões que estão activos, tanto numa janela de opções como na altura em que se vai ler os discos ou pastas.
ST18	Deve ser possível usar base de dados local ou remota, e o programa não pode ficar preso a apenas um tipo de base de dados.
ST19	Deve ser possível procurar ficheiros duplicados, o nome não deve ser limitação.
ST20	Deve ser possível comparar a informação guardada de um disco ou pasta com o seu conteúdo actual, caso o disco ou pasta estejam disponíveis ao programa.
ST21	O programa deve oferecer um sistema de gestão de saídas. Deve permitir gerir um conjunto de utilizadores e os empréstimos feitos a esses utilizadores.
ST22	Deve ser possível adicionar, remover e editar os dados de utilizadores, bem como ver os empréstimos actuais e passados feitos a esse utilizador.
ST23	A aplicação deverá correr em qualquer um dos sistemas operativos mais usados, desde Windows XP/Vista, Linux a Mac OS X.
ST24	São necessárias, algumas funções de manutenção, como a possibilidade de alterar a numeração dos discos catalogados.

8.5.2 Distribuição das User Stories por Iterações

Agrupar as *user stories*, permite que *user stories* semelhantes ou dependentes sejam implementadas juntas e que, iterações que se previam demasiado pequenas possam ver o seu tempo desenvolvimento aumentar.

Ao apresentar um grupo de *user stories*, o mesmo será precedido por um código, composto pelo prefixo CB, ao qual foi adicionado o número do grupo. A Tabela 3 apresenta a distribuição das *user stories* pelas iterações determinadas.

Tabela 3. Lista de user stories agrupadas

CB01	<p>O programa deve oferecer um sistema de gestão de saídas. Deve permitir gerir um conjunto de utilizadores e os empréstimos feitos a esses utilizadores (ST21).</p> <p>Deve ser possível adicionar, remover e editar os dados de utilizadores, bem como ver os empréstimos actuais e passados feitos a esse utilizador (ST22).</p>
CB02	<p>As opções mais comuns devem aparecer numa barra acima da lista de ficheiros (ST13).</p> <p>Deve existir um menu na árvore com a lista de categorias/grupo de discos/etiquetas com opções de pesquisa, introdução de descrições, alteração do nome, exportação e importação (ST14).</p>
CB03	<p>O sistema de leitura de informações de ficheiro tem de permitir ser expandido para ler ficheiros desconhecidos ou para que a forma como as informações são lidas seja alterada (ST16).</p>
CB04	<p>Quando se trata de um disco amovível deve existir a possibilidade de guardar a localização física do disco (ST3).</p> <p>É necessário encontrar um ficheiro no catálogo, para isso deve ser possível escolher a categoria, grupo de discos, ou todo o catálogo para restringir a pesquisa (ST6).</p> <p>A pesquisa de ficheiros deve conter opções para pesquisar usando expressões regulares (ST12).</p> <p>Se a localização do ficheiro estiver acessível deve ser possível abri-la (ST15).</p> <p>Deve ser possível usar base de dados local ou remota, e o programa não pode ficar preso a apenas um tipo de base de dados (ST18).</p> <p>Deve ser possível procurar ficheiros duplicados, o nome não deve ser limitação</p>

	(ST19).
CB05	<p>O programa tem de permitir criar catálogos, abrir catálogos, fechar e guardar catálogos (ST4).</p> <p>Os discos e pastas podem ser catalogados usando etiquetas, o sistema deve ser independente das categorias e grupos de discos e permitir que o mesmo disco tenha mais que uma categoria (ST5).</p> <p>A aplicação deve permitir ser distribuída em mais que uma língua (ST7).</p> <p>São necessárias, algumas funções de manutenção, como a possibilidade de alterar a numeração dos discos catalogados (ST24).</p>
CB06	<p>Deve ser possível exportar um catálogo completo ou um subconjunto para um conjunto de ficheiros HTML que possam ser colocados on-line ou distribuídos livremente. Devem existir personalizações de aspecto que se possam aplicar aos ficheiros criados (ST11).</p>
CB07	<p>Opções para permitir exportar os dados existentes para outros formatos, principalmente para formatos portáteis como XML e CSV (ST9).</p> <p>Opções para importar dados de outras bases de dados criadas anteriormente com o programa e de outros formatos providenciados por programas concorrentes (ST10).</p>
CB08	<p>Sistema de estatísticas que mostre dados relevantes sobre os ficheiros existentes como o número de ficheiros de determinado tipo, o tamanho médio dos ficheiros catalogados, o tamanho total ocupado, etc. (ST8).</p>
CB09	<p>O programa tem de permitir que se possa indicar a pasta a ler e tem de ler o conteúdo dessa pasta e guardar as informações dos ficheiros (ST1).</p> <p>Deve ser possível comparar a informação guardada de um disco ou pasta com o seu conteúdo actual, caso o disco ou pasta estejam disponíveis ao programa (ST20).</p>

8.6 Convenções de Código

Para que alguns princípios da metodologia XP, tais como o Código Colaborativo, possam funcionar correctamente, é necessário estabelecer algumas regras, tais como regras para a escrita de código. Estas regras são fundamentais quando qualquer elemento da equipa é encorajado a trabalhar no código, mesmo que não tenha sido criado por si.

As seguintes regras são fortemente baseadas nas convenções de escrita de código Java[39], criadas pela Sun, sendo que, no que se refere à forma como o código é estruturado, considerou-se suficiente o que é já indicado nas convenções da linguagem Java.

O código deverá ser escrito usando inglês para permitir o fácil acesso por programadores externos aquando da abertura total do *software*.

Todos os métodos têm de incluir um comentário usando a estrutura do sistema Javadoc[38] para que seja possível exportar documentação sobre o código usando ferramentas próprias para o efeito.

Os comentários devem conter, pelo menos, uma breve descrição do objectivo do método ou classe a que se referem, a descrição dos parâmetros de entrada e de saída e a indicação de qualquer excepção que possam lançar. Embora não seja obrigatório, a referência a classes relacionadas que sejam relevantes pode constituir uma ajuda na percepção da estrutura do código e deve ser feita sempre que possível.

Métodos que não sejam implementados terão de ter no seu corpo um comentário de linha com uma mensagem do propósito do método bem como deverão lançar uma excepção que indique a não implementação do código. Ex:

```
//TODO: <mensagem>  
throw new UnsupportedOperationException("Not implemented yet");
```

Caso seja necessário acrescentar notas no código, por exemplo para sinalizar linhas de código que se pensam mudar mais tarde ou remover, deverá ser colocado um comentário de linha com o formato:

```
//NOTE: <mensagem>
```

8.7 Testes de Usabilidade

8.7.1 Questionários Apresentados

Aos utilizadores que testaram a aplicação foram apresentados dois questionários, o primeiro pretendia obter algum conhecimento sobre a população de teste, o segundo incluía avaliações às tarefas efectuadas durante os testes.

Foi fornecido aos utilizadores uma versão impressa do manual da aplicação e um folha com todos os passos a executar. Apenas foram dadas explicações sobre onde se encontravam os recursos necessários para a normal execução do teste, como por exemplo, catálogos previamente criados.

De seguida são apresentados, de forma resumida, os dois testes efectuados. Uma vez que foi usado um sistema de formulários preenchidos no navegador de *Internet*, os questionários aqui replicados não possuem a mesma apresentação que a encontrada pelos utilizadores.

Testes de Usabilidade ao sistema JFindMyFiles - I

Este questionário tem por objectivo a avaliação de usabilidade do sistema JFindMyFiles. As informações serão usadas apenas para garantir que o sistema vai de encontro às necessidades dos utilizadores e não serão fornecidas a terceiros.

Os dados pessoais pedidos, servem para fins meramente estatísticos, pretendendo dar uma ideia dos indivíduos que preencheram os testes.

* Denota perguntas de resposta obrigatória.

Idade: *

- Menos de 18 anos
- Menos de 25 anos
- 35 anos ou menos
- Mais de 35 anos

Sexo *

- Masculino

- Feminino

Habilitações *:

Profissão *:

Com que frequência usa o computador? *

- Pelo menos 1 vez por semana
- Pelo menos 3 vezes por semana
- Pelo menos 5 dias por semana

Aproximadamente, durante quanto tempo por sessão, usa o computador? *

- 1 hora
- 3 horas
- 6 horas
- Mais de 8 horas

Utiliza ou já utilizou uma ferramenta de catalogação de ficheiros? *

- Sim
- Não

Se respondeu "Sim" na pergunta anterior, por favor indique qual. Se respondeu "Não", passe à pergunta seguinte.

Sente necessidade de uma aplicação que permita gerir os seus ficheiros digitais? *

- Sim
- Não

O sistema JFindMyFiles é de fácil utilização. *

- 1- Concordo Plenamente/ 2 - Concordo/ 3 - Sem opinião/ 4 - Discordo/ 5 - Discordo Plenamente

O sistema JFindMyFiles é difícil de aprender. *

- 1- Concordo Plenamente/ 2 - Concordo/ 3 - Sem opinião/ 4 - Discordo/ 5 - Discordo Plenamente

As mensagens do sistema JFindMyFiles são de fácil compreensão. *

- 1- Concordo Plenamente/ 2 - Concordo/ 3 - Sem opinião/ 4 - Discordo/ 5 - Discordo Plenamente

A ajuda do sistema JFindMyFiles é útil. *

- 1- Concordo Plenamente/ 2 - Concordo/ 3 - Sem opinião/ 4 - Discordo/ 5 - Discordo Plenamente

Se pudesse mudar 3 coisas no JFindMyFiles, o que mudaria?

Tem algum comentário e/ou sugestão a fazer?

Testes de Usabilidade ao sistema JFindMyFiles - II

Neste questionário serão pedidas as avaliações das tarefas executadas no sistema JFindMyFiles.

Classifique cada uma das tarefas de acordo com o seu grau de dificuldade, tendo em conta que o valor 1 significa "Muito Difícil" e valor 5, "Muito Fácil".

Todas as perguntas são de preenchimento obrigatório.

- Abra um catálogo existente, (encontra um no seu ambiente de trabalho).
- Verifique que extensões se encontram activas e desactive-as todas.
- Adicione um grupo de discos ao catálogo.
- Adicione um disco ao grupo que criou na tarefa anterior.
- Verifique se existem ficheiros iguais no catálogo aberto.
- Exporte os dados, usando um dos templates HTML disponíveis.
- Procure ficheiros com o nome "JFindMyFiles". Use as opções que achar relevantes.

- Crie um catálogo novo usando os dados fornecidos.
- Na sua folha estarão um conjunto de dados que permitirão ligar a uma base de dados.
- Altere os dados pessoais referentes ao autor do catálogo.
- Adicione a pasta onde habitualmente se encontra montado o seu CD-ROM. Esta tarefa deverá ser executada apenas por utilizadores em Linux

8.7.2 Resultados dos Testes de Usabilidade

Tabela 4. Resultados do primeiro questionário

Idade	Menos de 25 anos	35 anos ou menos	35 anos ou menos	Menos de 25 anos	Menos de 25 anos
Sexo	Masculino	Feminino	Masculino	Feminino	Feminino
Habilitações	Bacharel	12	12º	12º	Licenciatura
Profissão	Estudante	Estudante	estudante	estudante	estudante
Com que frequência usa o computador?	Pelo menos 5 dias por semana	Pelo menos 5 dias por semana	Pelo menos 5 dias por semana	Pelo menos 5 dias por semana	Pelo menos 5 dias por semana
Aproximadamente, durante quanto tempo por sessão, usa o computador?	6 horas	Mais de 8 horas	Mais de 8 horas	Mais de 8 horas	Mais de 8 horas
Utiliza ou já utilizou uma ferramenta de catalogação de ficheiros?	Não	Não	Sim	Não	Não
Se respondeu "Sim" na pergunta anterior, por favor indique qual. Se respondeu "Não", passe à pergunta seguinte.			desconhecido		
Sente necessidade de uma aplicação que permita gerir os	Sim	Sim	Sim	Sim	Sim

seus ficheiros digitais?					
O sistema JFindMyFiles é de fácil utilização.	2	1	2	2	2
O sistema JFindMyFiles é difícil de aprender.	5	4	4	3	4
As mensagens do sistema JFindMyFiles são de fácil compreensão.	2	1	2	2	3
A ajuda do sistema JFindMyFiles é útil.	2	1	3	1	1
Se pudesse mudar 3 coisas no JFindMyFiles, o que mudaria?			botões da toolbar maiores com texto por baixo a opção "não perguntar novamente" no popup de saída		
Tem algum comentário e/ou sugestão a fazer?					

Tabela 5. Resultados do segundo questionário

Abra um catálogo existente, (encontra um no seu ambiente de trabalho).	4	5	5	4	5
Verifique que extensões se encontram activas e desactive-as todas.	2	4	2	3	1
Adicione um grupo de discos ao catálogo.	5	5	5	5	5
Adicione um disco ao grupo que criou na tarefa anterior.	5	5	4	5	3
Verifique se existem ficheiros iguais no catálogo aberto.	4	5	4	3	2

Exporte os dados, usando um dos templates HTML disponíveis	3	5	5	4	5
Procure ficheiros com o nome "JFindMyFiles". Use as opções que achar relevantes.	4	5	5	4	5
Crie um catálogo novo usando os dados fornecidos.	5	5	5	4	3
Altere os dados pessoais referentes ao autor do catálogo.	1	4	1	1	1
Adicione a pasta onde habitualmente se encontra montado o seu CD-ROM.	3	3		3	1

8.8 Imagens do Produto final

As imagens seguintes pretendem ser uma mostra da aplicação criada, oferecendo uma visão geral da sua utilização. Foram obtidas executando a aplicação no SO Ubuntu Linux.

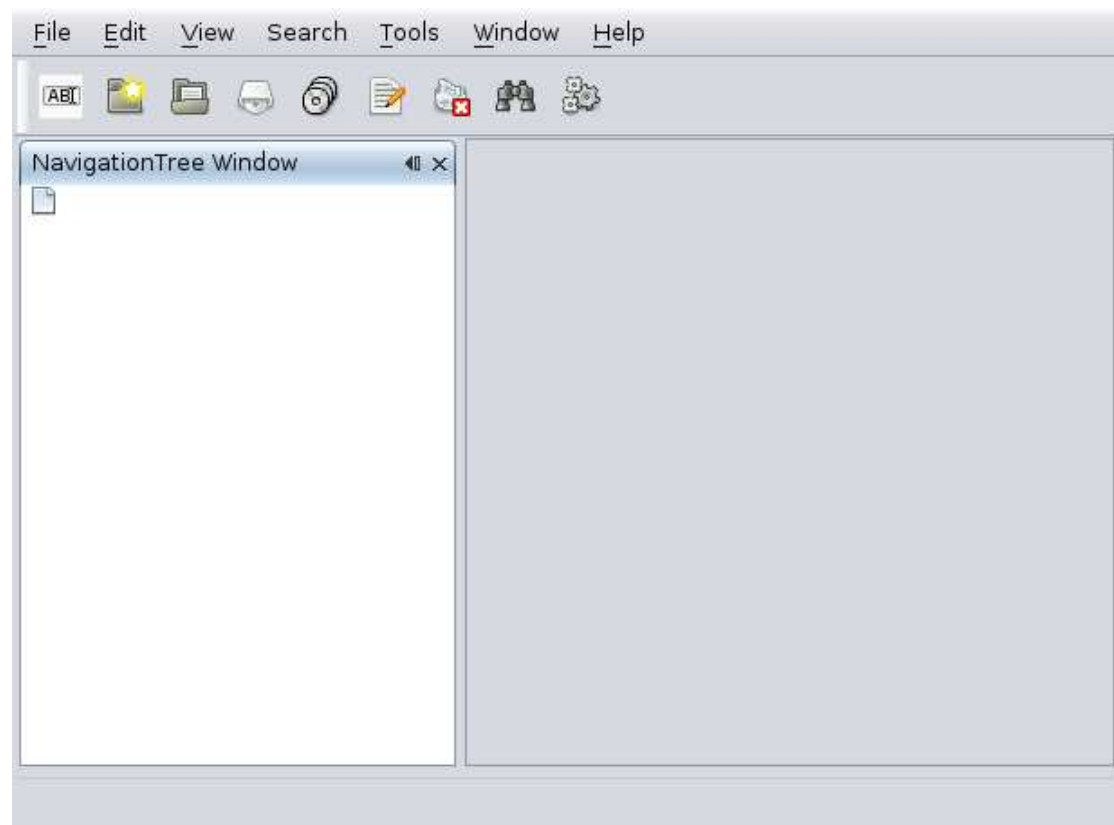


Figura 25. Sistema sem catálogo aberto

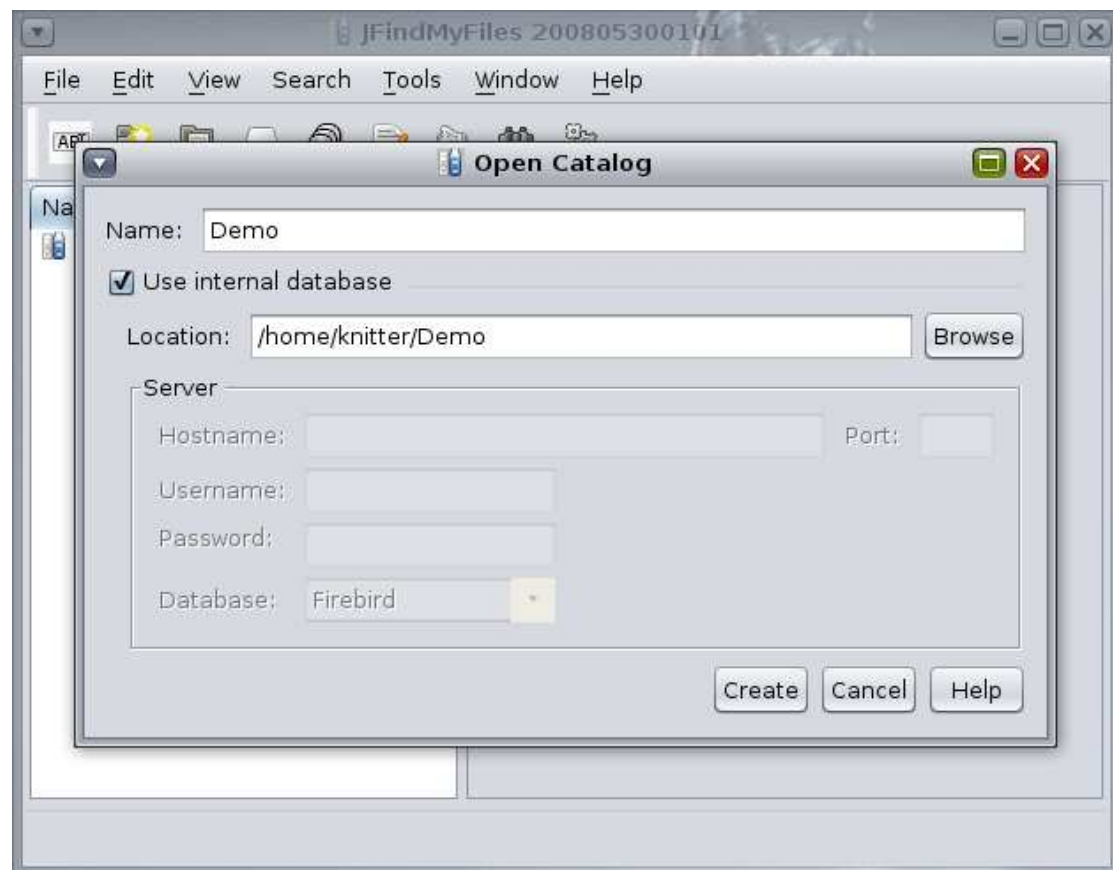


Figura 26. Janela que permite abrir um catálogo

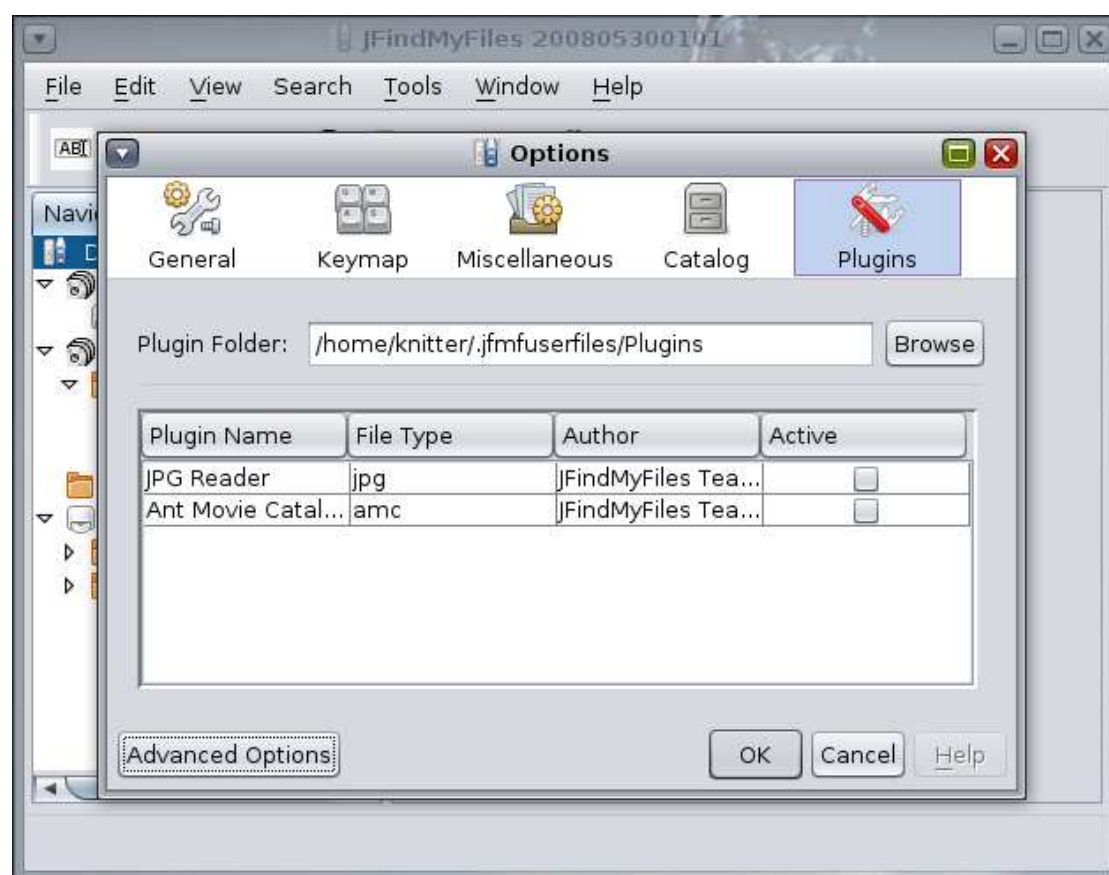


Figura 27. Configuração das extensões existentes

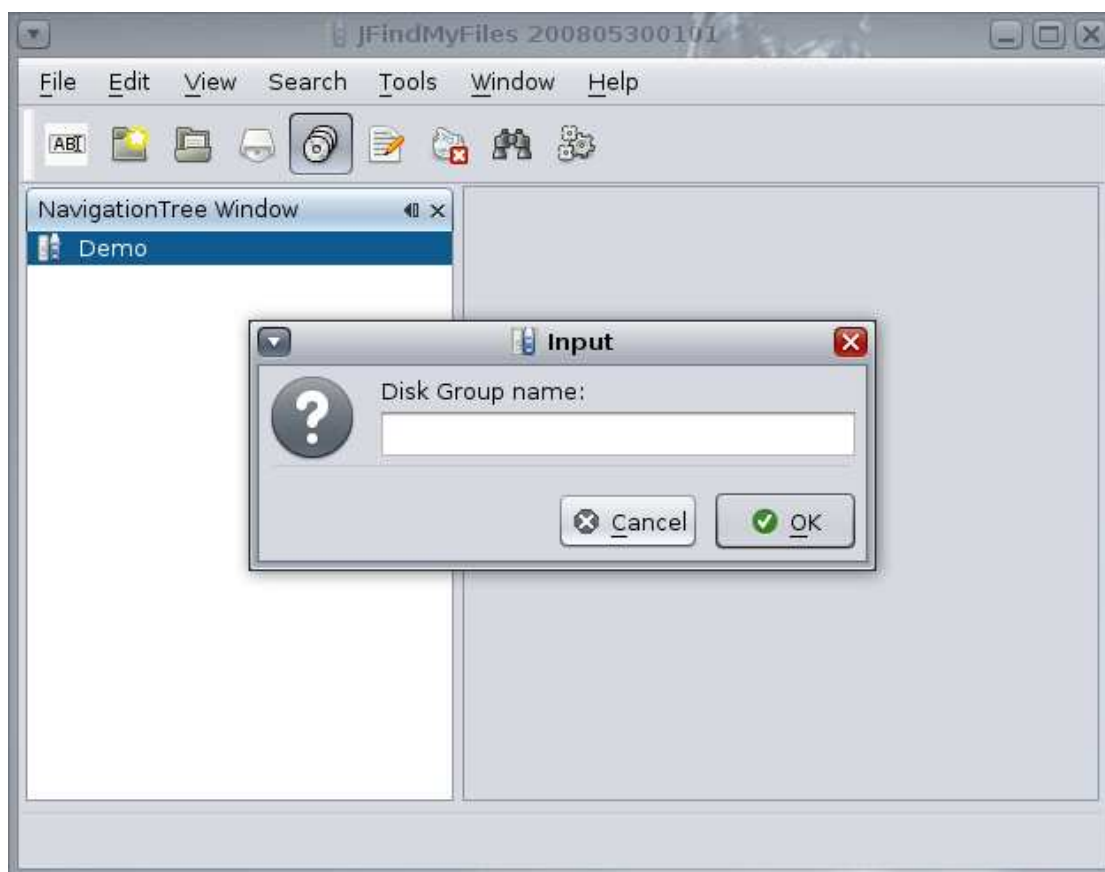


Figura 28. Adição de um grupo de discos

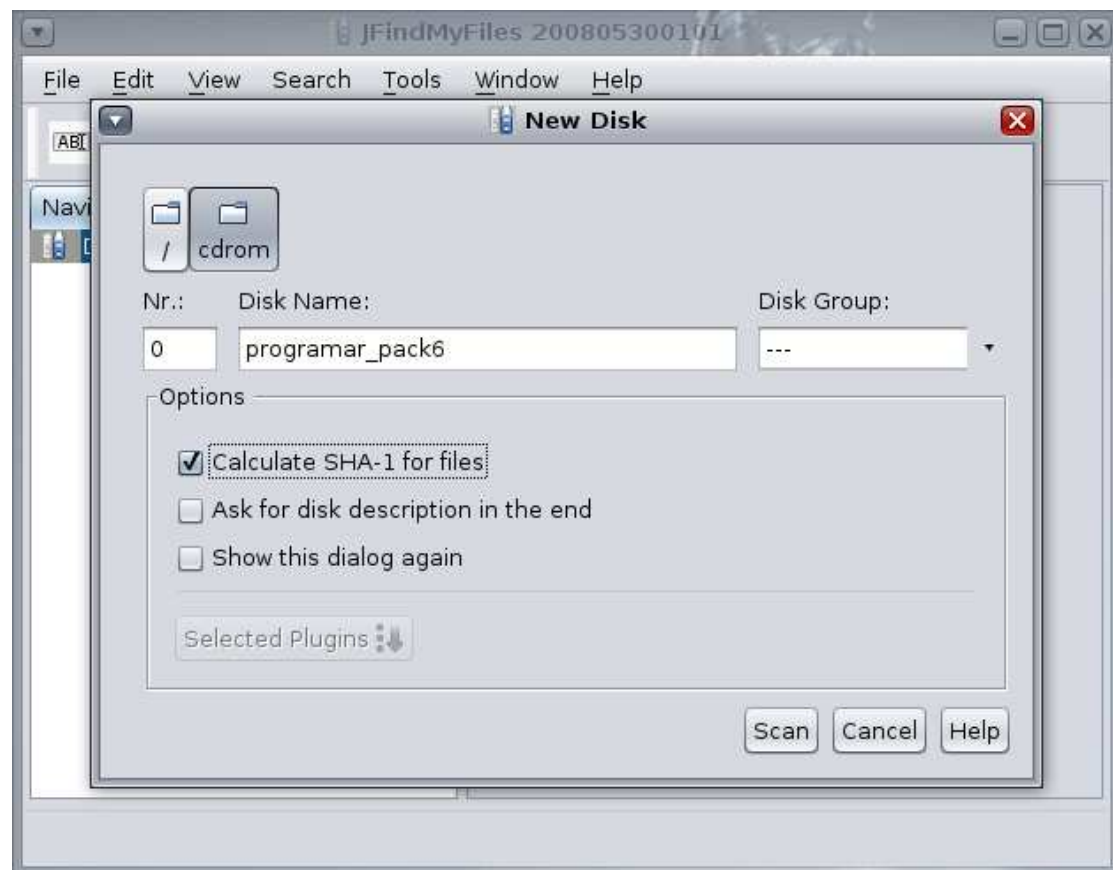


Figura 29. Janela que permite iniciar a leitura de um disco

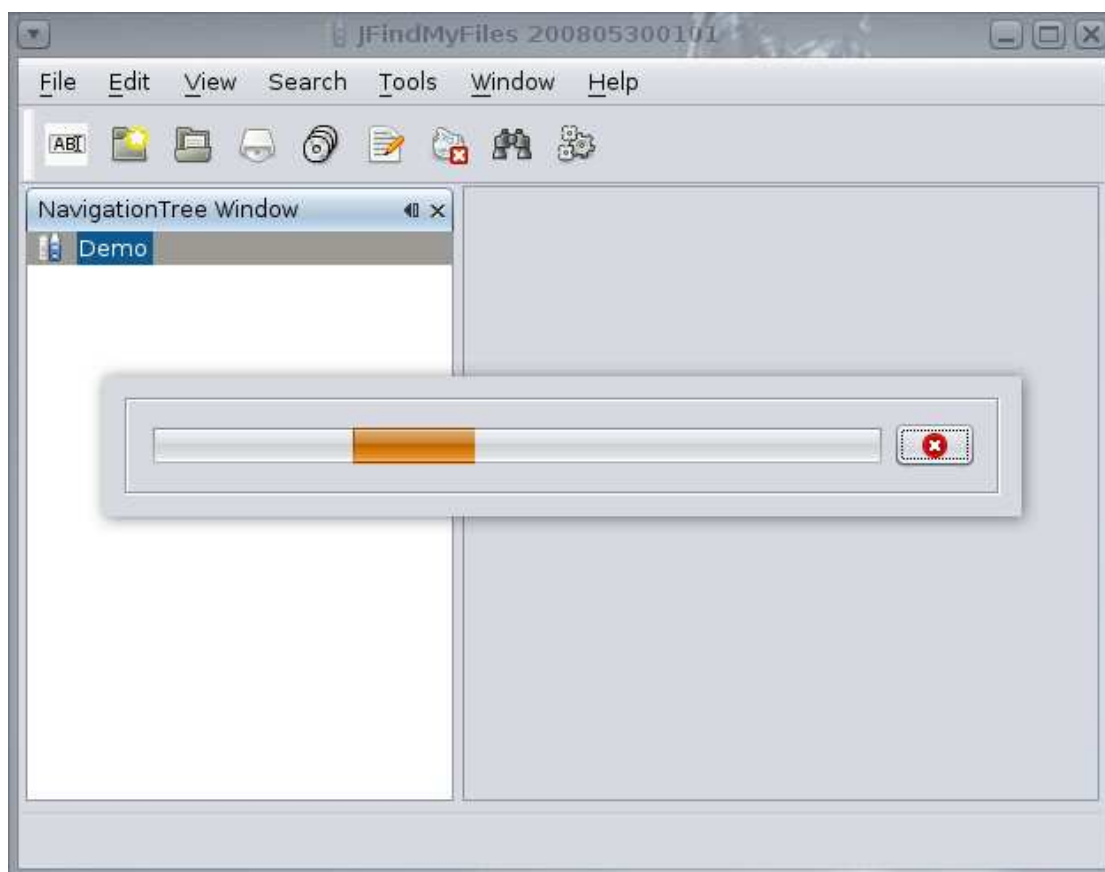


Figura 30. Processo de leitura de um disco

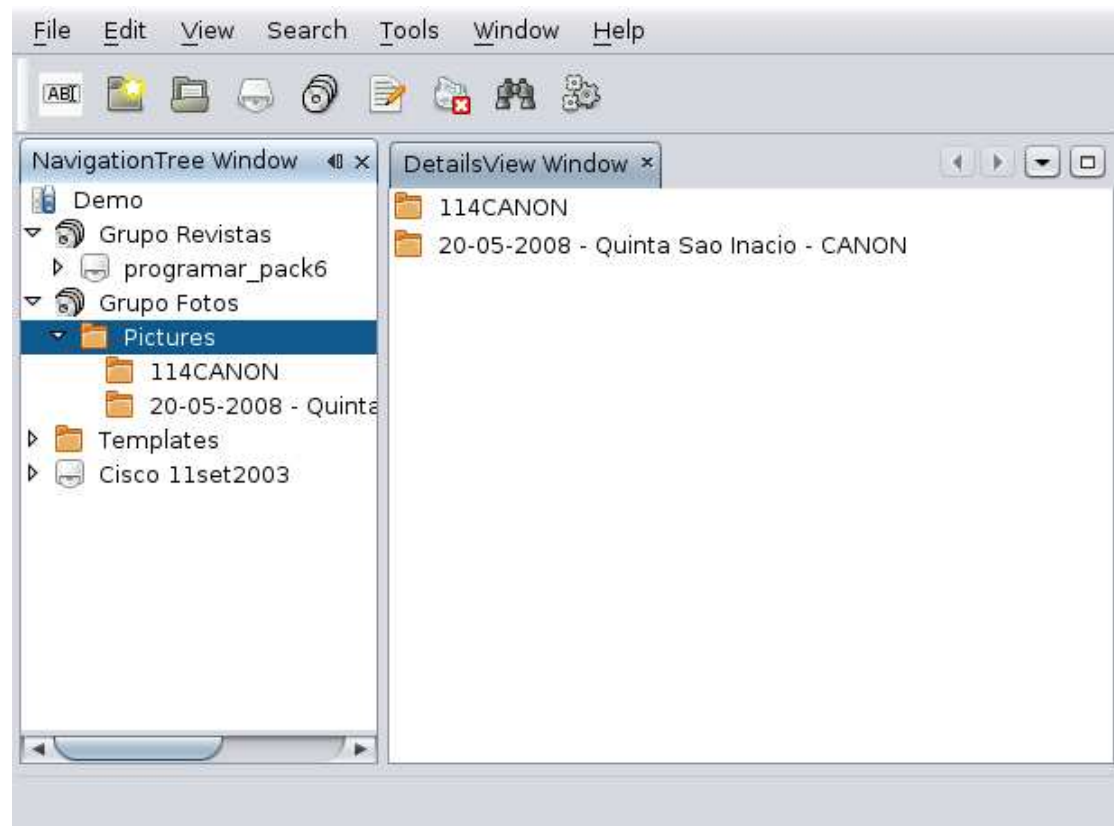


Figura 31. Janela principal do sistema com catálogo aberto

9 Bibliografia

- [1] Eclipse Bugs. *[OpenGL] MAC OS Leopard: OpenGL error with snippet 195*. 2008. 20-03-2008.

https://bugs.eclipse.org/bugs/show_bug.cgi?id=221483

- [2] Eclipse Bugs. *SWT_AWT not implemented for Mac*. 2006. 20-03-2008.

https://bugs.eclipse.org/bugs/show_bug.cgi?id=67384

- [3] Eclipse Bugs. *SWT_AWT.new_Shell() unimplemented on OS X*. 2008. 20-03-2008.

https://bugs.eclipse.org/bugs/show_bug.cgi?id=145890

- [4] Sun Microsystems. Project Swing (Java Foundation Classes). 2002 20-03-2008.

<http://java.sun.com/j2se/1.5.0/docs/guide/swing/> 10-07-2008.

- [5] Wikipedia, Wiki. 2008. 10-07-2008

<http://en.wikipedia.org/wiki/Wiki>

- [6] Tödter, Kai. *Eclipse und NetBeans*. Eclipse Magazin, Vol. 12 e 13

- [7] Well, Don. *Extreme Programming: A gentle introduction*. 2008 15-03-2008.

<http://www.extremeprogramming.org>

- [8] Nielsen, Jakob. *Usability Engineering*. Academic Press Limited, 1993

- [9] Martin, Robert C. *Agile Software Development, Principles, Patterns and Practice*. Prentice Hall, 2002

- [10] Rosenberg, Doug, Stephens, Matt, Collins-Cope, Mark. *Agile Development with ICONIX process, People, Process, and Pragmatism*. Apress, 2005

- [11] Shafranovich, Yakov. *Common Format and MIME Type for CSV Files*. 2005. 20-05-2008.

<http://www.rfc-editor.org/rfc/rfc4180.txt>

- [12] Microsoft Corporation. *Microsoft SQL Server 2008*. 2008. 30-03-2008

<http://www.microsoft.com/sqlserver/2008/en/us/default.aspx>

- [13] Sun Microsystems. *MySQL*. 2008. 30-03-2008

<http://dev.mysql.com>

- [14] PostgreSQL Global Development Group. *PostgreSQL*. 2008. 30-03-2008

<http://www.postgresql.org>

- [15] The hsqldb Development Group. *HSQldb*. 2008. 30-03-2008

<http://www.hsqldb.org>

- [16] WardsWiki. *Object Relational Tool Comparison*. 2007. 15-03-2008

<http://c2.com/cgi/wiki/ObjectRelationalToolComparison>

- [17] Sun Microsystems. *TopLink Essentials*. 2008. 15-05-2008

<https://glassfish.dev.java.net/javaee5/persistence>

- [18] Red Hat Middleware, LLC, *Hibernate*. 2006. 15-05-2008

<http://www.hibernate.org/>

- [19] Toedter, Kai. *JCalendar Java Bean, a Java Date Chooser*. 2006. 15-05-2008

<http://www.toedter.com/en/jcalendar/index.html>

- [20] Löfflmann, Johann N. *Jacksum, a java checksum utility*. 2008. 25-05-2008

<http://www.jonelo.de/java/jacksum/>

- [21] Noakes, Drew. *Metadata Extractor*. 2006. 25-05-2008

<http://www.drewnoakes.com/code/exif>

- [22] Sun Microsystems. *MySQL@Connector/J*. 2008. 25-05-2008

<http://www.mysql.com/products/connector/j>

- [23] The jTDS Project. *jTDS*. 2008. 25-05-2008

<http://jtds.sourceforge.net/>

- [24] The PostgreSQL Global Development Group. *PostgreSQL JDBC Driver*. 2005. 25-05-2008

<http://jdbc.postgresql.org/>

- [25] Firebird Project. *Jaybird JDBC/JCA Drivers*. 2008. 25-05-2008

<http://www.firebirdsql.org/index.php?op=devel&sub=jdbc>

- [26] Microsoft Corporation. *MICROSOFT SQL SERVER 2005 JDBC DRIVER REDISTRIBUTION LICENSE*. 2008. 15-07-2008

<http://msdn.microsoft.com/en-us/data/aa937726.aspx>

- [27] Gantt Project . *Gantt Project Home*. 2008. 15-03-2008

<http://ganttproject.biz/>

- [28] ProjectPier. *ProjectPier.org*. 2008. 14-03-2008

<http://www.projectpier.org/>

- [29] Sun Microsystems. *NetBeans*. 2008. 14-03-2008.

<http://www.netbeans.org/>

- [30] H. Don. *Notepad++*. 2008. 15-03-2008

<http://notepad-plus.sourceforge.net/>

- [31] The Gnome Project. *Gedit*. 2007. 10-07-2008

<http://www.gnome.org/projects/gedit/>

- [32] Sun Microsystems, *XTest*. 2008. 15-03-2008.

<http://xtest.netbeans.org/>

- [33] Apache Software Foundation. *The Apache Ant Project*. 2008. 10-07-2008

<http://ant.apache.org/>

- [34] JUnit.org. *JUnit*. 2008. 15-03-2008

<http://www.junit.org/>

- [35] NetBeans user list. *Choosing Netbeans platform or Eclipse-RCP*. 2008.

<http://www.nabble.com/Choosing-Netbeans-platform-or-Eclipse-RCP-to16012394.html>

- [36] The Tortoise Team. *TortoiseSVN*. 2008. 15-03-2008

<http://tortoisesvn.tigris.org/>

- [37] Syntevo. *SmartSVN – Subversion/SVN Cliente*. 2008. 15-03-2008

<http://www.syntevo.com/smartsvn/index.html>

- [38] Sun Microsystems. *Javadoc Tool*. 2008. 15-07-2008.

<http://java.sun.com/j2se/javadoc/>

- [39] Sun Microsystems. *Code Conventions for the Java Programming Language*. 2008. 15-03-2008.

<http://java.sun.com/docs/codeconv/>

- [40] Tellico KDE-Apps.org. *Tellico*. 2008. 20-03-2008.

<http://www.kde-apps.org/content/show.php/Tellico?content=10030>

- [41] Fujiwata Software. *DiskCatalogMaker*. 2008. 20-03-2008

<http://hp.vector.co.jp/authors/VA008942/library/diskcatalogmaker/>

- [42] Deák, Péter. *CdCat*. 2008. 10-03-2008.

<http://cdcat.sourceforge.net/>

- [43] Galle, Robert. *Where Is It?*. 2008. 20-03-2008.

<http://www.whereisit-soft.com>

- [44] Argentum Corporation, *Argentum MyFiles*. 2008. 20-03-2008.

<http://www.argentuma.com/myfiles.html>

- [45] Netbeans. *File Type Integration Tutorial*. 2008. 18-07-2008.

<http://platform.netbeans.org/tutorials/60/nbm-filetype.html>

- [46] Wikipedia. *Digital asset management*. 2008. 18-07-2008.

http://en.wikipedia.org/wiki/Digital_Asset_Management

- [47] Sun Microsystems. NetBeans Platform. 2008. 15-03-2008.

<http://platform.netbeans.org/>

- [48] BerliOS. *BerliOS Developer*. 2008. 14-03-2008.

<http://developer.berlios.de/>

- [49] NetBeans Wiki. *What is a Node?*. 2008. 19-07-2008.

<http://wiki.netbeans.org/DevFaqWhatIsANode>

- [50] Doerner, Norbert M. *CDFinder*. 2008. 15-03-2008.

<http://www.cdfinder.de/Asdasda>

- [51] Eclipse Wiki. *Rich Client Platform*. 2008. 20-03-2008.

http://wiki.eclipse.org/index.php/Rich_Client_Platform

- [52] Wikipedia. *Microsoft Foundation Class Library*. 2008. 15-03-2008.

http://en.wikipedia.org/wiki/Microsoft_Foundation_Class_Library

- [53] Microsoft Corporation. *MFC Reference*. 2008. 20-03-2008.

[http://msdn.microsoft.com/en-us/library/d06h2x6e\(vs.71\).aspx](http://msdn.microsoft.com/en-us/library/d06h2x6e(vs.71).aspx)

- [54] The Spring Framework. *Spring richclient*. 2008. 20-03-2008.

<http://spring-rich-c.sourceforge.net/1.0.0/index.html>Adasda

- [55] Firebird Project. *Firebird - The RDBMS that's going where you're going*. 2008. 20-03-2008

<http://www.firebirdsql.org/>

- [56] Wikipedia. *Qt(Toolkit)*. 2008. 15-07-2008.

http://en.wikipedia.org/wiki/Qt_%28toolkit%29