Faculdade de Engenharia da Universidade do Porto

FEUP

Desenvolvimento de Interfaces de Administração para o Sistema de Gestão Documental iPortalDoc-Light

Bruno Filipe Lage Gonçalves

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major Telecomunicações

Orientador: Prof. Dr. José António Rodrigues Pereira de Faria
Proponente: iPortalMais Serviço de Internet e Redes Lda

28 de Julho de 2010

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES 2009/2010

A Dissertação intitulada

"DESENVOLVIMENTO DE INTERFACES DE ADMINISTRAÇÃO PARA O SISTEMA DE GESTÃO DOCUMENTAL IPORTALDOC-LIGHT"

foi aprovada em provas realizadas em 22/Julho/2010

o júri

Presidente Professor Doutor Américo Lopes de Azevedo
Professor Associado do Departamento de Engenharia Industrial e Gestão da Faculdade de Engenharia da Universidade do Porto

Professor Doutor Paulo Jorge Freitas Oliveira Novais
Professor Auxiliar do Departamento de Informática da Universiddade do Minho

Professor Doutor José António Rodrigues Pereira de Faria
Professor Auxiliar do Departamento de Engenharia Industrial e Gestão da Faculdade de Engenharia da Universidade do Porto (Orientador)

O autor declara que a presente dissertação (ou relatório de projecto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extractos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são correctamente citados.

Bruno Filipe Lage Gonçalves

Autor - **Bruno Filipe Lage Gonçalves**

Faculdade de Engenharia da Universidade do Porto

Resumo

Com o crescente aumento da quantidade de informação e documentação que uma empresa necessita tratar, torna-se essencial o uso de sistemas que possam assegurar uma melhoria na eficiência com que essa informação é gerida. O uso de sistemas de gestão documental e de gestão de *workflow* permitem automatizar processos e organizar a documentação de forma mais eficiente, tornando-se assim essenciais para assegurar um melhor funcionamento das empresas.

Este projecto encontra-se inserido numa proposta da empresa *iPortalMais* e tem por objectivo o desenvolvimento de módulos *Web* de administração para o *iPortalDoc-Ligh*, este consiste num módulo externo ao *iPortalDoc* (sistema de gestão documental e *workflow* que a empresa referida fornece) e permite fornecer a utilizadores externos, informação contida no *iPortalDoc*.

Foram desenvolvidos quatro módulos que depois de integrados no *iPortalDoc-Light* permitiram melhorar a quantidade e qualidade das funcionalidades que este oferece. Estes módulos são: um menu de pesquisa avançada que permite ao utilizador efectuar pesquisas segundo os parâmetros que desejar; uma interface de administração onde é possível criar *workflows* virtuais que serão posteriormente apresentados a utilizadores externos no *iPortalDoc-Light*; uma interface de administração onde é possível criar um documento interligando t*emplates* e posteriormente apresentar esse documento no *iPortalDoc-Light*, apresentando cada template a inserir numa *tab* diferente; integrar a possibilidade de assinar digitalmente documento, com o *iPortalDoc-Light*.

Foi feito um levantamento das principais linguagens de programação que poderiam ser utilizadas para o desenvolvimento deste trabalho, tendo sido escolhidas para a implementação destes módulos, as linguagens PHP e *JavaScript* em conjunto com *frameworks JavaScript* YUI e *Prototype*.

Com a elaboração deste projecto e a realização de testes e simulações permitiram determinar a qualidade das aplicações desenvolvidas, sendo também assegurada a funcionalidade e compatibilidade destas. O uso de *frameworks* e AJAX assegurou a escalabilidade e interactividade das ferramentas desenvolvidas.

Abstract

With the increasing amount of information and documentation, that a company needs to process it is essential the use of systems that can ensure and improve the efficiency with which this information is managed. The use of document management systems and workflow management systems allows the companies to automate and organize the documentation more effectively, making it essential to ensure a better functioning business.

This project is part of a proposal of the iPortalMais company and consists on the development of administration *Web* modules for the iPortalDoc-Light, module adjacent to iPortalDoc (Workflow and Document management system that iPortalMais provides) that allows external entities to have access to information contained at the iPortalDoc.

The goals achieved with the pursuit of this project relate to the development of four modules that will be integrated into iPortalDoc-Light improving the quality and quantity of the functionalities that it offers. These modules are:

• An advanced search menu enabling the user to search according to the parameters he wants;

• A management interface to enable the creation of virtual workflows, which will later be presented to external users in iPortalDoc-Light;

• A management interface that allows the user to create a document by connecting templates and after present that document in iPortalDoc-Light, where in each tab is presented a different template.

• Integrate the possibility of digitally sign documents on iPortalDoc-Light.

A survey was conducted on the major programming languages that could be used in the development of this work, and the ones chosen to implement these modules where PHP and JavaScript together with YUI Framework and AJAX.

The implementation of this project and the tests and simulations performed has helped to determine the quality of the applications developed. It was already ensured its functionality and compatibility. The use of frameworks and AJAX secured the scalability and interactivity of the tools developed.

Agradecimentos

Este espaço é dedicado a todas as pessoas que me ajudaram e tornaram possível o desenvolvimento deste projecto. Em especial agradeço:

Primeiramente, aos meus pais. Sem eles, não só este trabalho, mas toda a minha educação não teria sido possível.

De seguida gostava de agradecer aos meus orientadores, o Professor José António Rodrigues Pereira de Faria pela ajuda e conhecimentos passados ao longo de todo este projecto e à Engenheira Telma Cristina Marques Salgueiro pela ajuda e acompanhamento prestado durante a minha estadia na *iPortalMais*.

Gostaria de agradecer também aos Engenheiros Sérgio Lopes, Rui e José Queirós por todo o conhecimento partilhado, à Cláudia Monteiro pela ajuda disponibilizada no desenvolvimento deste documento e a todos aqueles que mesmo não estando aqui referidos me ajudaram e motivaram.

Por fim um agradecimento especial à empresa *iPortalMais* por me ter proporcionado a oportunidade de aprender e ambientar num meio empresarial e tornar possível a realização deste projecto.

Índice

Lista de figuras

Lista de tabelas

Abreviaturas e Símbolos

AJAX	Asynchronous JavaScript and XML
ASP	Active Server Pages
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
CVS	Concurrent Versions System
DOM	Document Object Model
FEUP	Faculdade de Engenharia da Universidade do Porto
FS	File System
HTML	HyperText Markup Language
IDS	Intrusion Detection System
IE	Internet Explorer
JSON	*JavaScript Object Notation*
JVM	Java Virtual Machine
KDE	K Desktop Environment
LPDA	Lightweight Directory Access Protocol
PBX	Private Branch Exchange
PBX	Private Branch Exchange
PDF	Portable Document Format
PERL	Practical Extraction and Reporting Language
PHP	Hypertext Preprocessor
PHP/FI	Personal Home Page Tools
PVP	Preço de Venda ao Público
SGD	Sistema de Gestão Documental
SGDW	Sistemas de Gestão Documental e Workflow
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
SO	Sistema Operativo
SQL	Structured Query Language
TI	Tecnologias da Informação
UI	User Interface
URL	Uniform Resourse Locator
UX	User Experience
VoIP	Voice over Internet Protocol
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network

XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations
YUI	Yahoo User Interface

Capítulo 1

Introdução

Este trabalho foi desenvolvido no âmbito da dissertação do Mestrado Integrado em Engenharia Electrotécnica e de Computadores e insere-se na proposta proporcionada pela empresa *IPortalMais – Serviço de Internet e Redes, Lda*.

Neste capítulo é apresentada, inicialmente, uma contextualização da dissertação, relacionando as funcionalidades fornecidas pelo sistema de gestão documental e *workflow* (*iPortalDoc)* e pelo módulo externo associado a este (*iPortalDoc-Light)* com o mundo empresarial. Em seguida, serão apresentados os objectivos que as aplicações a desenvolver devem cumprir e, por último, é detalhada a estrutura do documento.

1.1- Contextualização

No contexto actual de gestão empresarial verifica-se um acréscimo na informação e consequentemente documentação que necessita ser tratada. Como tal, é notória a necessidade de automatizar certos processos, de modo a melhorar a eficiência e a organização de uma empresa.

Uma forma de atingir este propósito é recorrendo à utilização de sistemas de gestão de *workflow* e sistemas electrónicos de gestão documental. Sistemas deste tipo permitem a uma empresa automatizar as trocas de documentação entre aplicações e pessoas sem a necessidade de uso de documentos impressos, diminuindo, assim, o tempo gasto nas trocas de informação e potenciando significativamente a eficiência. Outra melhoria a acrescentar é a possibilidade de não limitar a troca de informação ao interior da empresa, mas sim poder facultar essa informação, de uma forma segura e limitada, a entidades externas que interajam com essa empresa.

O *iPortalDoc* é um Sistema de Gestão Documental e *Workflow* (SGDW) que a empresa *iPortalMais* desenvolve e comercializa, o qual proporciona todos os serviços e funcionalidades que um serviço deste tipo abrange. Para além disso, fornece, incorporado com o *iPortalDoc*, um módulo *Web* auxiliar, o *iPortalDoc-Light*, permitindo este transmitir informação a entidades externas que possuam este serviço.

Esta dissertação surge na sequência da proposta da empresa *iPortalMais* para o desenvolvimento de interfaces para o módulo *iPortalDoc-Light*. Este consiste num módulo *Web* complementar ao *iPortalDoc* e permite fornecer, a entidades externas, o acesso a informação. As funcionalidades a adicionar a este módulo serão um menu de pesquisa avançada, a possibilidade de visualizar um *workflow* virtual baseado no apresentado no *iPortalDoc*, um menu de inserção de templates baseado em *tabs* e a possibilidade de assinar digitalmente documentos com recurso ao cartão do cidadão.

A adição destas funcionalidades permitirá melhorar a disponibilização de informação e filtrar a sua apresentação, optimizar a inserção da mesma e garantir os níveis de segurança com o recurso à assinatura digital.

1.2- Objectivos

Como foi referido anteriormente, o objectivo deste projecto consiste no desenvolvimento de funcionalidades que serão integradas no módulo *Web iPortalDoc-Light*.

As funcionalidades a desenvolver são as seguintes:

- Filtro de pesquisa avançada que permita a um utilizador do *iPortalDoc-Light* efectuar uma pesquisa de acordo com as características da procura que pretende efectuar. Os resultados da pesquisa serão filtrados consoante as permissões de cada utilizador;
- Gestão de *workflows* virtuais baseados nos *workflows* existentes no *iPortalDoc*. Para cada *workflow* existente no *iPortalDoc* será necessário configurar a informação (etapas, descrições, comentários, etc.) que será visível para utilizadores externos que acedam a esta funcionalidade a partir do *iPortalDoc-Light*. Deverá ser criado um *workflow* virtual que será idêntico ao *workflow* original criado no *iPortalDoc*, tendo, no entanto, a informação disponibilizada limitada;
- Gestão das interfaces de inserção de documento. Pretende dar-se a um utilizador externo a possibilidade de preencher *templates* de certos tipos de documentos (de modo semelhante ao *iPortalDoc*), sendo então os templates apresentados com recurso a *tabs*, de modo a tornar a sua associação mais intuitiva;
- Inserção, no módulo *iPortalDoc-Light*, da possibilidade de assinar documentos por assinatura digital recorrendo ao cartão do cidadão. Uma empresa necessitará de um leitor de cartões, da aplicação do cartão do cidadão e do titular do cartão. Ao inserir o cartão, os certificados serão registados automaticamente, possibilitando posteriormente a assinatura de documentos electrónicos.

A implementação destas funcionalidades no módulo *iPortalDoc-Light* conduzirá a uma maior autonomia e organização ao nível dos utilizadores e facilidade nas configurações por parte das organizações que possuam o *iPortalDoc* e o *iPortalDoc-Light*.

1.3- Estrutura do documento

Esta dissertação encontra-se organizada em seis capítulos.

O primeiro capítulo é composto pelo contexto e enquadramento desta dissertação, os objectivos a alcançar e a estrutura do documento.

No segundo capítulo, é efectuado um levantamento de conceitos e tecnologias que serão utilizadas neste projecto.

No capítulo três é efectuada uma análise e concepção de cada um dos módulos a desenvolver.

No capítulo quatro é demonstrada a implementação de cada um dos módulos.

No quinto capítulo são apresentadas as interfaces obtidos depois do desenvolvimento de cada módulo.

No capítulo seis são descritas as conclusões e apresentado o trabalho futuro.

Por último são apresentados os anexos e a bibliográfica consultada que serviu de suporte a este projecto.

4 Introdução

Capítulo 2

Revisão de conceitos e tecnologias

Como fase inicial no desenvolvimento deste projecto foi necessário efectuar um levantamento de conceitos e tecnologias direccionadas aos sistemas alvo deste trabalho. Uma apresentação precisa sobre esse levantamento será feita neste capítulo.

Numa primeira parte, será efectuada uma análise aos sistemas e tecnologias utilizadas pelo *iPortalDoc* e iPortalDoc-Light, nomeadamente sistemas de gestão documental e sistemas de gestão de *workflow*. De seguida será apresentada uma descrição das funcionalidades principais destas duas plataformas. Na terceira parte, serão descritas as linguagens de programação usadas no desenvolvimento do projecto. Por fim será apresentadas as conclusões obtidas deste levantamento.

2.1- Sistemas e tecnologias

Como referido acima, nesta secção são mostradas as tecnologias utilizadas pelo *iPortalDoc*, nomeadamente uma contextualização sobre sistemas de gestão documental e *Workflow*, dois sistemas sobre os quais assentam os princípios básicos de funcionamento dos módulos *iPortalDoc* e *iPortalDoc-Light* será também feita uma introdução à tecnologia de assinatura digita descrevendo o seu funcionamento e vantagens da integração com este tipo de sistemas.

2.1.1 - Sistemas de Gestão Documental

À medida que os computadores se tornam cada vez mais prevalentes no mundo do trabalho é espectável que estes irão eliminar a necessidade do papel, o que resultaria na diminuição dos gastos, na melhoria da eficiência e na eliminação de contactos cara-a-cara necessários num ambiente baseado em papel. No entanto, um ambiente empresarial sem papel foi impossível de atingir até aos nossos dias e existem estudos que comprovam que as pessoas conseguem reter mais 30% de informação se lhes for mostrada em papel, ao invés de num ecrã de computador (*New Zealand Management*, 2001; King,2001) [1].

Contudo, um factor crítico no desenvolvimento de qualquer ambiente empresarial actual é o aumento da agilidade operacional. Por esse facto, uma atempada e relevante organização e disponibilização de informação a todos os intervenientes impõe-se em toda e qualquer empresa. Cada vez mais, a utilização de documentos é essencial ao bom funcionamento de uma empresa. O uso extensivo e desorganizado de documentação leva a diversos problemas, tais como, documentos perdidos, diversas versões, autoria incerta/desconhecida dos documentos, custos com armazenamento e cópias, entre outros. Estes diversos factores aliados ao aumento exponencial do uso de informação por parte das empresas conduzem à necessidade de uma solução rápida e eficaz que possa captar, armazenar, disponibilizar, pesquisar e gerir toda a informação produzida por uma empresa.

Uma solução para o problema exposto acima é o uso de sistemas de gestão documental. Estes permitem diminuir os riscos, clarificando o ciclo de vida da documentação, tornando-se assim uma vantagem competitiva para qualquer empresa que empregue um sistema deste tipo. Embora existam empresas com a percepção de que um sistema deste tipo seria uma mais-valia no que toca à produtividade de uma empresa, factores como o "gosto pelo papel" dificultam a mudança. No entanto, encarar a situação com receio e não agir é como adiar a solução óbvia de um problema que cresce diariamente!

Recorrendo a uma breve contextualização sobre os Sistemas de Gestão Documental, para se perceber a sua importância e consequente relevância deste projecto realizado, poderemos afirmar que estes apareceram de modo a possibilitar uma melhor gestão da informação como, por exemplo, em documentos impressos, fotografias, entre outros e consequentemente melhorar a partilha da mesma [2].

Actualmente os sistemas de gestão documental permitem fornecer às empresas e utilizadores um serviço que permite uma maior rapidez, eficácia, organização, segurança e fiabilidade no acesso à informação produzida, recebia ou armazenada por estes. Um sistema de gestão documental (em que os documentos sejam guardados em formato electrónico) permite ter a informação ordenada, classificada e arrumada, fazendo com que a procura de informação seja executada com maior rapidez e a sua distribuição mais eficaz.

Poderemos ainda referir algumas características da gestão documental, como por exemplo, o conjunto de passos que poderá seguir: os documentos são recebidos em formato papel, de seguida digitalizados para se converterem em documento electrónico, sendo-lhes atribuída, posteriormente, uma classificação. Depois da entrada no sistema são definidos os vários estados do seu ciclo de vida, ou seja o seu percurso no sistema, e por fim, disponibilizado o acesso a esses documentos a qualquer utilizador previamente definido, por exemplo, através de um serviço como o *iPortalDoc*.

De acordo com [3] as principais funcionalidades/vantagens de um sistema de gestão documental são:

- **Desmaterialização** – Possibilidade de digitalização de documentos que poderão ser classificados e, posteriormente, disponibilizados de acordo com critérios predefinidos;
- **Normalização** – Possibilidade de uniformização de processos e normalização de todo o tipo de documentos e entidades;
- **Indexação** – Facilitação do processo de catalogação e classificação dos documentos electrónicos, em comparação com documentos em suporte físico;
- **Pesquisa** – Existência de motores de pesquisa que facilitam a procura de informação, de uma forma mais rápida, a partir de qualquer lugar;

- **Redução de custos** – A digitalização de documentos reduz os custos com papel, levando a um melhor aproveitamento do espaço.

De salientar que estas vantagens podem ainda ser optimizadas se a elas lhe juntarmos as características a nível de segurança que lhe estão inerentes. Encriptação de dados, certificação cronológica, assinatura digital, controlo de acesso de dados e definição de perfis de utilizador são algumas das possibilidades de melhoramento que podem ser incorporadas num SGD.

Estas funcionalidades, em conjunto com uma política de compromisso da empresa em racionalizar os seus processos, permitem que estas obtenham um retorno positivo devido aos visíveis ganhos de produtividade, uniformização e gestão automatizada de processos.

Todas as vantagens acima descritas são fornecidas pelo *iPortalDoc*, produto desenvolvido pela empresa *iPortalMais*. Um serviço de SGDW para *Internet*/*Intranets* desenvolvido com base em Linux que se encontra integrado no ambiente de trabalho dos utilizadores, sendo acessível através de aplicações comuns de processamento de texto, correio electrónico e browser. No ponto 2.2.2 são descritas as funcionalidades e arquitectura deste sistema.

2.1.2 - Sistemas de *Workflow*

Um *workflow* consiste numa sequência de passos que permite sistematizar a informação, documentação ou tarefas entre utilizadores, de acordo com uma série de regras, permitindo, assim, tornar estes processos mais simples e intuitivos aos utilizadores. A automatização de um processo identifica as diversas actividades, regras e dados usados para gerir o ciclo de vida de um *workflow* [4].

Um *workflow* é constituído por:

- **Caminhos** - caminhos entre elementos que podem ser lineares, circulares ou paralelos;
- **Regras** - condições que permitem a transição de estado;
- **Papéis** - funções de pessoas ou programas;
- **Tarefas** - acções que podem ser realizadas por utilizadores ou programas que, depois de completas, definem o caminho do *workflow*.

Um sistema de gestão de *workflow* cria, gere e executa *workflows* através de *softwares* que corram um motor de *workflow*. Este interpreta processos, cria e gere instâncias e interage com participantes e aplicações. O *workflow* gere as actividades resultantes de um processo e as suas relações do início ao fim deste. Durante o ciclo de vida de um processo, este é instanciado a um participante e são-lhe atribuídas tarefas que serão controladas pelo sistema de gestão de *workflow* para cada invocação do processo ou actividade, assegurando, desta forma, o seu correcto desenvolvimento [4].

O uso de um sistema de gestão de *workflow* por parte de uma empresa trará inúmeras vantagens, como por exemplo, a automatização dos seus processos de negócios e a interacção entre utilizadores e clientes. Isto conduzirá a uma maior eficiência a nível operacional que terá como resultado uma melhoria visível na qualidade dos serviços prestados.

Um sistema deste tipo permite minimizar os documentos duplicados e/ou perdidos e a empresa pode perceber melhor quanto tempo fica um documento na posse de uma pessoa,

bem como quanto tempo demora em média cada processo. A velocidade dos processos aumenta significativamente e, consequentemente, reflecte-se em melhor desempenho na empresa [2].

Na figura 2.1 podemos observar um exemplo simples de um *workflow*.

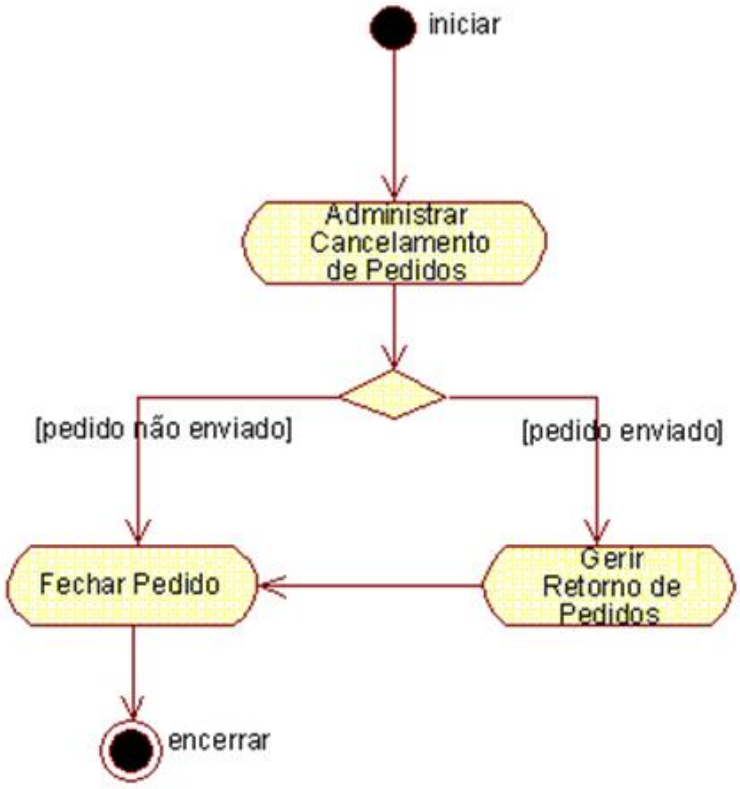

Figura 2.1 - Exemplo de workfow

Como pode ser visto, desde o inicio até ao fim do ciclo de vida do *workflow*, este passa um conjunto de etapas bem definidas:

- **Cancelamento de pedidos pelo administrador** - estado inicial do *workflow*;
- **Envio de pedido** - se o pedido não for enviado passa ao estado final e o pedido é fechado, caso o pedido seja enviado há um gerir retorno de pedidos e de seguida passa ao estado final e o pedido é fechado;
- **Fechar pedido** - estado final do *workflow*.

Poderemos, em jeito de conclusão de apresentação das características de um sistemas de gestão de workflow (como o disponibilizado pelo *iPortalDoc*), afirmar que este possibilita tratar e encaminhar a informação automaticamente, conduzindo assim a uma diminuição da intervenção humana e consequente melhoria da eficiência dos processos - tanto em serviços internos, como em serviços prestados a outros.

2.1.3 - Assinatura Digital

A assinatura digital consiste numa assinatura electrónica semelhante à assinatura física em papel, que permite a autenticação da identidade do autor de um documento electrónico.

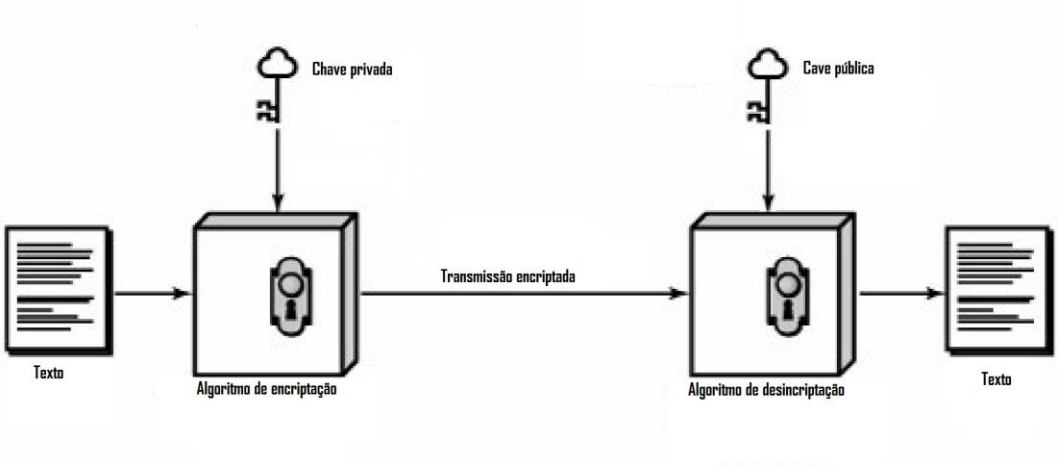

Figura 2.2 - Exemplo de assinatura

Como pode ser visto na figura 2.2, num sistema de assinatura digital cada utilizador tem uma chave privada (com que pode encriptar um documento e assim assinar digitalmente o mesmo) e uma chave pública que será partilhada, permitindo, assim, a um outro utilizador desencriptar o documento encriptado, autenticando a identidade do autor. Se a chave privada de cada utilizador se mantiver secreta, esta permite validar, de uma forma segura, a autenticidade do signatário de um documento electrónico.

O uso da assinatura digital deverá permitir a autenticação do utilizador, o não repúdio (o utilizador não pode negar que assinou um documento) e a integridade do documento (não deverá ser possível modificar o documento sem a permissão do utilizador) [5].

Cada cartão do cidadão possui uma chave privada, transformando-a numa ferramenta muito interessante no que diz respeito a assinar digitalmente documentos electrónicos.

Dado o *iPortalDoc* ter como um dos seus principais objectivos o aumento da produtividade, diminuindo a utilização de documentos em papel, a possibilidade de assinar instantaneamente e de um modo seguro um documento (recorrendo a assinatura digital) será uma solução bastante vantajosa quando comparada com a assinatura tradicional.

2.2- Plataformas

Nesta secção são apresentadas as plataformas sobre as quais assentam o desenvolvimento deste projecto, especialmente o *iPortalDoc* e *iPortalDoc-Light*, sendo também feita uma breve referencia ao *IPBrick* e às principais aplicações fornecidas por este sistema operativo que a empresa *iPortalMais* também desenvolve e comercializa, e sobre o qual assenta o funcionamento do *iPortalDoc*.

2.2.1 - IPBrick

A *IPBrick* é uma solução para servidores de comunicações e *intranet* que a empresa *iPortalMais* criou, e actualmente, desenvolve e comercializa.

Assenta num sistema operativo *Unix* baseado no *kernel* do *Linux* que devido ao seu potencial, robustez, usabilidade e preço fornece uma alternativa fiável à, por exemplo, *Microsoft*, disponibilizando às empresas a segurança no acesso à *Internet*, um sistema de comunicações unificadas e ainda e serviços de *groupware* e *e-mail*.

Poderemos apontar como características principais da *IPBrick*: software para servidores de *intranet* e comunicações, *gateway* de comunicações, *gateway* para telefonia VoIP e PBX, recuperação do sistema, instalação automática e interface *Web* funcional [6]. No anexo A.1 podem ser vistas as diversas aplicações da *IPBrick* [7].

O *iPortalDoc* é suportado pelo módulo *iPBrick.IC* integrando assim os serviços fornecidos pelo *iPBrick* com os sistemas de gestão documental e *workflow* fornecidos pelo *iPortalDoc*. Uma das principais vantagens da interligação prende-se com a gestão de utilizadores que podendo esta ser feita através do *IPBrick* recorrendo à ferramenta *IPContactos*. Isto pode ser visto neste capítulo, no ponto 2.2.2.3.2 onde é demonstrada a forma como os utilizadores são adicionados.

2.2.2 - iPortaldoc

Neste ponto, é efectuada uma descrição do *iPortalDoc*. Este revela-se essencial ao desenvolvimento deste projecto visto todas as configurações que são aplicadas ao *iPortalDoc-Light*, tanto a nível de permissões como de funcionalidades, serem realizadas neste módulo. Inicialmente, através de uma apresentação, serão focadas as principais vantagens que uma aplicação deste tipo permite; em seguida, serão expostas as principais funcionalidades fornecidas e que serão importantes para o desenvolvimento deste trabalho e, por fim, será feita uma descrição da arquitectura, salientando as aplicações mais relevantes para o desenvolvimento deste projecto.

2.2.2.1 - Apresentação

Em 2001, um ano após o nascimento da *iPortalMais*, surge a oportunidade da empresa dar os primeiros passos no i&DT com a ideia de desenvolver uma solução de Gestão Documental e *Workflow* (*networking*), como um serviço de rede baseada em *Linux* (sistema operativo que usufruía, durante esta época, de uma intensa actividade por parte dos seus apoiantes na cidade do Porto), dando assim origem ao *iPortalDoc*.

O *iPortalDoc* é um Sistema integrado de Gestão Documental e *Workflow* que a empresa *iPortalMais* disponibiliza de modo a melhorar a gestão documental e fluxo de trabalho de qualquer empresa que o decida adquirir [8].

De todas as vantagens que um serviço como o *iPortalDoc* pode proporcionar a uma empresa, são de salientar as seguintes [8]:

- Melhoria no acesso à informação dentro da organização;
- Informação sempre acessível e a vários utilizadores simultaneamente;
- Gestão do ciclo de vida dos documentos em toda a organização;
- Redução da circulação da informação em suporte papel e consequente rentabilização do espaço físico e redução de custos;
- Maior eficácia e eficiência da consulta;

- Maiores níveis de segurança dos documentos digitais;
- Facilidade de utilização, que implica uma formação reduzida;
- Integração no ambiente de trabalho dos utilizadores e na intranet;
- Integração com aplicação de fax.

Outra vantagem do *iPortalDoc* é a sua fácil utilização devido ao seu interface gráfico intuitivo e o facto de não ter necessidade de instalação de qualquer tipo de aplicação, ou seja, é apenas preciso um *browser* (Firefox, Internet Explorer, Crome, etc.), leitor de correio electrónico e gestor de ficheiros - aplicações estas que são fornecidas com o Sistema Operativo. Para além disso, visto tratar-se de um serviço de baixo custo, será uma alternativa vantajosa em relação a serviços idênticos de grandes empresas.

2.2.2.2 - Funcionalidades

Depois de devidamente autenticado o utilizador terá acesso à página principal do *iPortalDoc*, são listadas as acções que o utilizador tem pendentes sobre determinado documento. Estas são as tarefas, associadas aos *workflows*, que ainda não foram realizadas. A lista de acções pode ser acedida a qualquer momento, sendo para isso apenas necessário clicar no ícone que se encontra em cima do nome do utilizador.

Na figura 2.3, podemos ver a interface *Web* principal do *iPortalDoc* onde teremos acesso a várias funcionalidades. De salientar que o acesso aos menus é atribuído de acordo com as permissões de cada utilizador. Nas imagens apresentadas de seguida, poderemos visualizar uma interface de utilizadores com perfil de *Super User*.

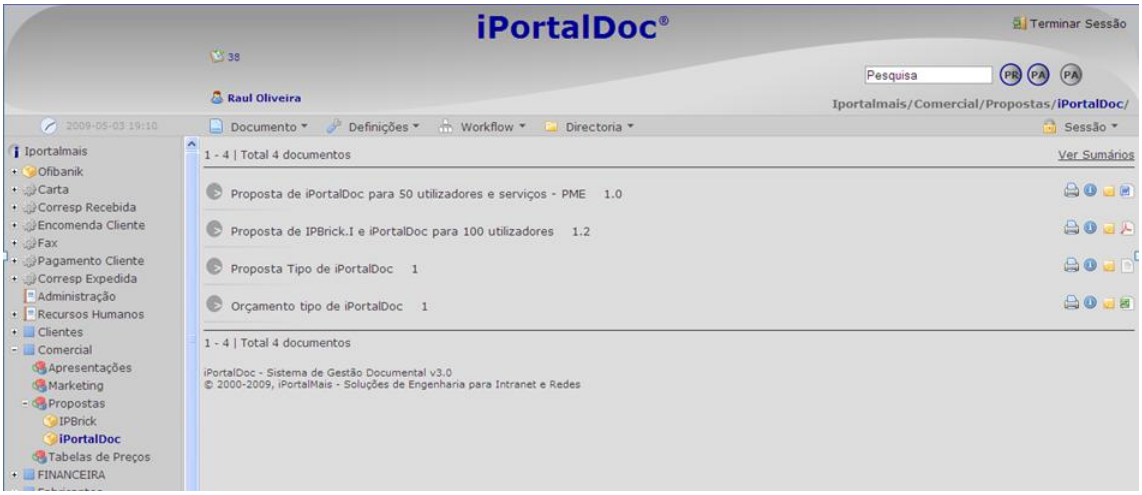

Figura 2.3 - Interface principal do iPortalDoc

Do lado esquerdo, podemos visualizar as directorias existentes (uma hierarquia de pastas e subpastas que dão acesso aos documentos presentes em cada uma delas). Este menu permite a um utilizador interagir com os documentos de cada pasta. Como podemos ver na figura 2.3, ao seleccionar uma pasta, serão visualizados os documentos que estão contidos na mesma e que serão apresentados no menu central. Na sequência de cada documento, são dadas quatro opções através de ícones representativos: imprimir, ver informação, abrir a localização e descarregar o documento.

Na barra de ferramentas temos acesso a quatro menus: “Documento”, “Definições”, “Workflow” e “Directoria”. Esta visualização dos quatro menus varia consoante as permissões atribuídas a cada utilizador, podendo ser possível só a visualização de um único menu: “Documento”.

Estes menus serão descritos de seguida.

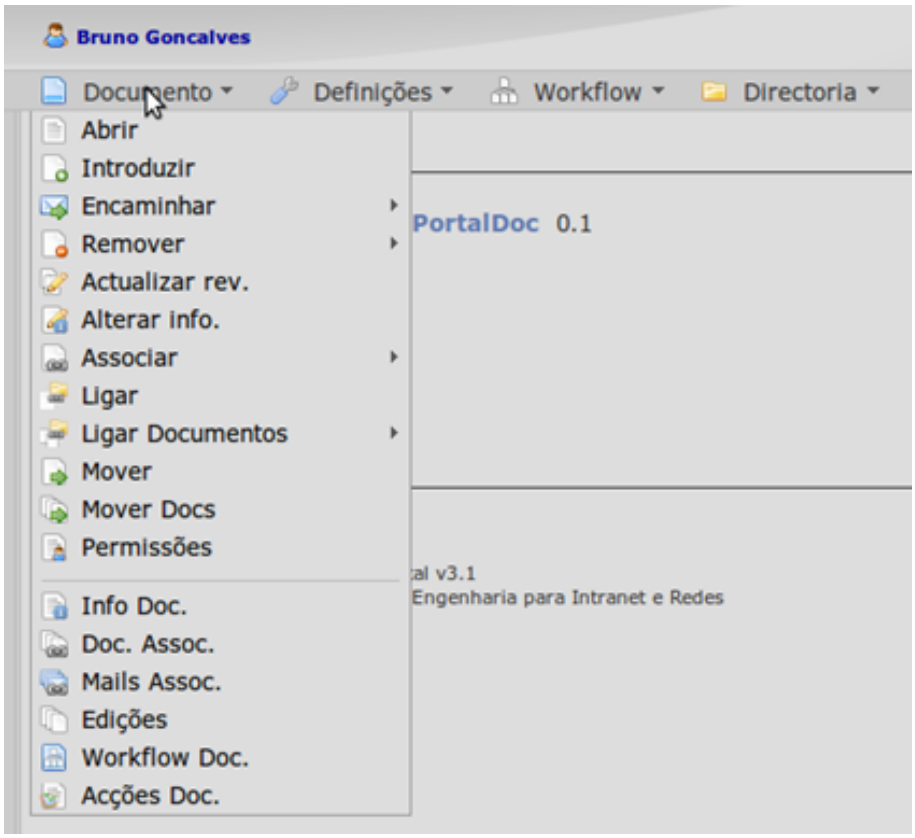

Figura 2.4 - Menu Documento

Na figura 2.4 podemos ver a listagem das opções a que um utilizador com permissões de *super user* tem acesso ao menu “Documento”, estas são:

- **Abrir** – Permite abrir o documento seleccionado (devendo este encontrar-se a azul quando seleccionado);
- **Introduzir** – Permite introduzir um novo documento. Ao seleccionar esta opção será também disponibilizado um formulário onde poderão ser introduzidas informações sobre o documento, associar um *workflow* ao documento, escolher entidades a que o documento está associado, entre outras. Existem duas formas de introdução de um documento: por upload de ficheiro ou por preenchimento de um template PDF;
- **Encaminhar** – Permite encaminhar o documento para outros utilizadores que serão devidamente notificados por *email*. Poderá escolher entre utilizadores internos, ou enviar para uma entidade externa inserindo, para isso, o *email* correspondente. O encaminhamento será registado no *workflow* respectivo;
- **Remover** – Neste menu são dadas duas opções, remover sem ou com pesquisa. Na primeira é removido o documento seleccionado da directoria actual. Na segunda, efectua-se uma pesquisa avançada onde será possível introduzir várias características

do documento que se pretende eliminar, escolhendo, posteriormente, o documento com as características resultantes dessa pesquisa;

- **Actualizar Ver.** – Permite actualizar o documento seleccionado. O documento mais antigo será guardado e associado à nova versão. A última versão será a apresentada;
- **Alterar Info.** – Permite alterar a informação inserida quando o documento foi adicionado, ou seja, a parte de classificação do documento;
- **Associar** – Permite associar um documento seleccionado a outro documento que se considere pertinente. Esta informação ficará presente no *+info* do documento respectivo;
- **Ligar** – Permite ligar o documento seleccionado para outra directoria;
- **Ligar documentos** – Permite ligar mais do que um documento;
- **Mover** – Permite mover o documento seleccionado para outra directoria;
- **Permissões** – Permite alterar as permissões do documento seleccionado;
- **Info. Doc.** – Permite visualizar a informação do documento – as especificações que definimos aquando da introdução do documento no sistema;
- **Doc. Assoc.** – Permite visualizar os documentos que foram associados ao documento seleccionado;
- **Mails Assoc.** – Permite visualizar os emails associados ao documento seleccionado.
- *Edições* – Permite ter acesso às diferentes edições do documento seleccionado;
- **Workflow Doc.** – Permite visualizar as etapas do *workflow* que já foram efectuadas;
- **Acções Doc.** – Permite visualizar acções que necessitam ser efectuadas sobre o documento para que este possa prosseguir o seu caminho no sistema.

Figura 2.5 - Menu Definições

Na figura 2.5 é possível visualizar as opções a que um utilizador tem acesso ao aceder ao menu "Definições":

- **Utilizadores** – Permite visualizar os dados de utilizadores associados ao *iPortalDoc* e associar utilizadores do iPBrick ao *iPortalDoc*;

- **Perfil** – Permite a um *super user* atribuir um perfil a um utilizador, configurando assim as suas permissões;
- **Assunto** – Permite editar o assunto que foi atribuído a um documento;
- **Ass. Utilizador** – Permite associar um utilizador a um tipo de documento;
- **Tipo Doc** - Permite criar, alterar ou remover tipos de documentos;
- **Templates PDF** – Permite criar, alterar ou remover modelos de documentos. Para além de poderem ser criados modelos, podem ser editados modelos já criados de modo a corresponderem aos requisitos necessários por cada instituição;
- **Constrói FS Global** – Permite reconstruir o sistema de ficheiros virtual de modo a ser acedido por um explorador de ficheiros de um qualquer sistema operativo;
- **Constrói mailFS Global** – Permite reconstruir o sistema de emails associado aos utilizadores;
- ***iPortalDoc-Light*** – Esta interface permite configurar os acessos ao *iPortalDoc-Light*.

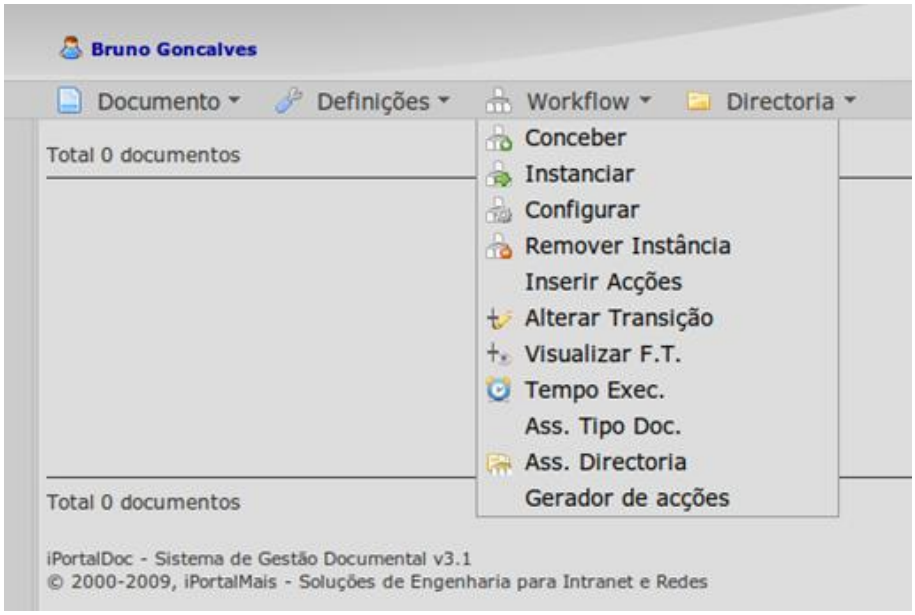

Figura 2.6 - Menu *Workflow*

De seguida são apresentadas as opções disponíveis no menu "Workflow", representado na figura 2.6. Este menu permite:

- **Conceber** – Esta opção permite criar *workflows* ou modificar/remover *workflows* já existentes;
- **Instanciar** – Permite instanciar um *workflow* uma entidade. O mesmo *workflow* pode ser instanciado para diferentes entidades;
- **Configurar** – depois de um *workflow* ter sido instanciado, esta opção permite atribuir acções do *workflow* a diferentes utilizadores;
- **Remover Instância** – Permite visualizar as instâncias atribuídas a um *workflow* e removê-las;
- **Inserir Acções** – Permite inserir uma acção num determinado estado de um determinado *workflow*;
- **Alterar Transição** – Permite alterar uma transição entre estados;
- **Visualizar F.T.** – Permite visualizar as funções de transição do *workflow* respectivo;

- **Tempo Exec.** – Permite configurar o tempo que cada utilizador tem para realizar uma determinada acção (por defeito encontra-se definido 30 dias);
- **Ass. Tipo Doc.** – Permite associar um *workflow* a um tipo de documento;
- **Ass. Directoria** – Permite associar um *workflow* a uma directoria de hierarquia;
- **Gerador de Acções** – Permite gerar acções que poderão ser inseridas num *workflow*.

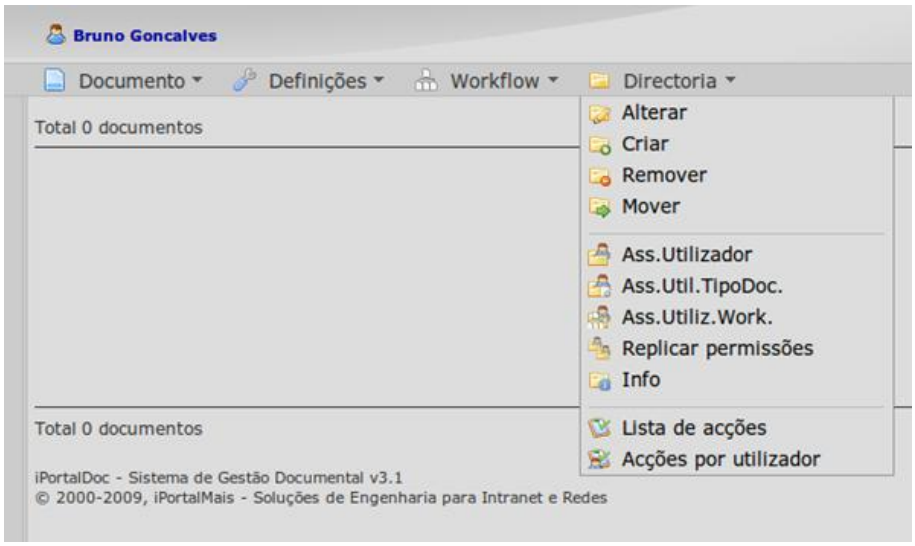

Figura 2.7 - Menu Directoria

Na figura 2.7 são mostradas as opções do menu "Directoria":

- **Alterar** – Permite alterar os dados de uma directoria previamente criada;
- **Criar** – Permite criar uma directoria ou uma subdirectoria, sendo para tal necessário seleccionar uma directoria raiz;
- **Remover** – Permite remover uma directoria sendo eliminado todo o conteúdo contido nesta;
- **Mover** – Permite mover directorias e subdirectorias;
- **Ass.Utilizador** – Permite associar um utilizador a uma secção;
- **Ass.Util.TipoDoc.** – Permite associar um utilizador a um tipo de documento;
- **Ass.Util.Work** – Permite associar um utilizador a um *workflow*;
- **Replicar Permissões** – Permite replicar as permissões atribuídas a um utilizador a outro utilizador;
- **Info** – Permite visualizar a informação de uma determinada directoria;
- **Lista de Acções** – Permite visualizar as acções que estão por realizar em cada secção;
- **Acções por Utilizador** – Permite visualizar todas as acções que estão por realizar por um determinado utilizador.

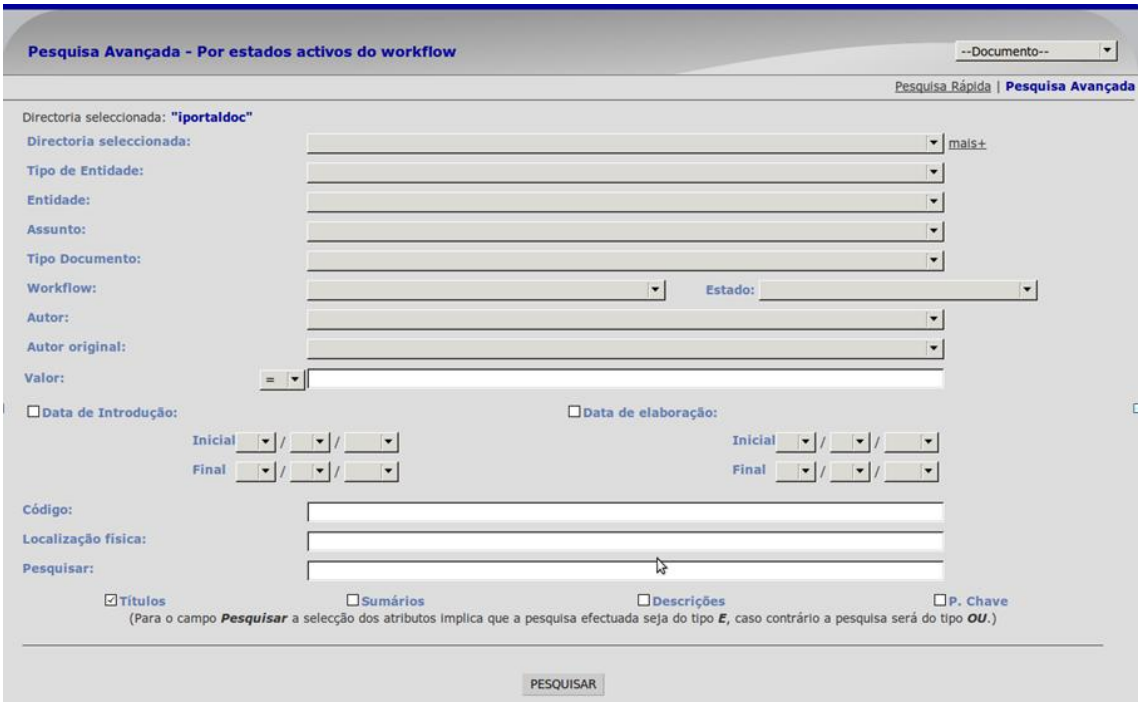

Figura 2.8 - Pesquisa avançada

No canto superior direito da interface principal do *iPortalDoc* encontram-se os menus de pesquisa. Estes permitem efectuar dois tipos de pesquisa:

- **Pesquisa Rápida** - O motor de pesquisa irá procurar palavras-chave nas directorias e subdirectorias
- **Pesquisa avançada** - Como se pode ver na figura 2.8, este tipo de pesquisa é muito mais detalhada e complexa permitindo, desta forma, ao utilizador escolher determinados parâmetros que caracterizam a sua pesquisa.

Uma funcionalidade relevante para o desenvolvimento deste projecto é a assinatura digital. O *iPortalDoc* facilita esta funcionalidade adicionando na criação de um *workflow* a acção "Assinar o documento com a assinatura electrónica do cartão do cidadão".

Quando, durante o ciclo de vida do *workflow*, o utilizador necessitar de realizar esta acção, ser-lhe-á apresentada a interface que pode ser vista em baixo.

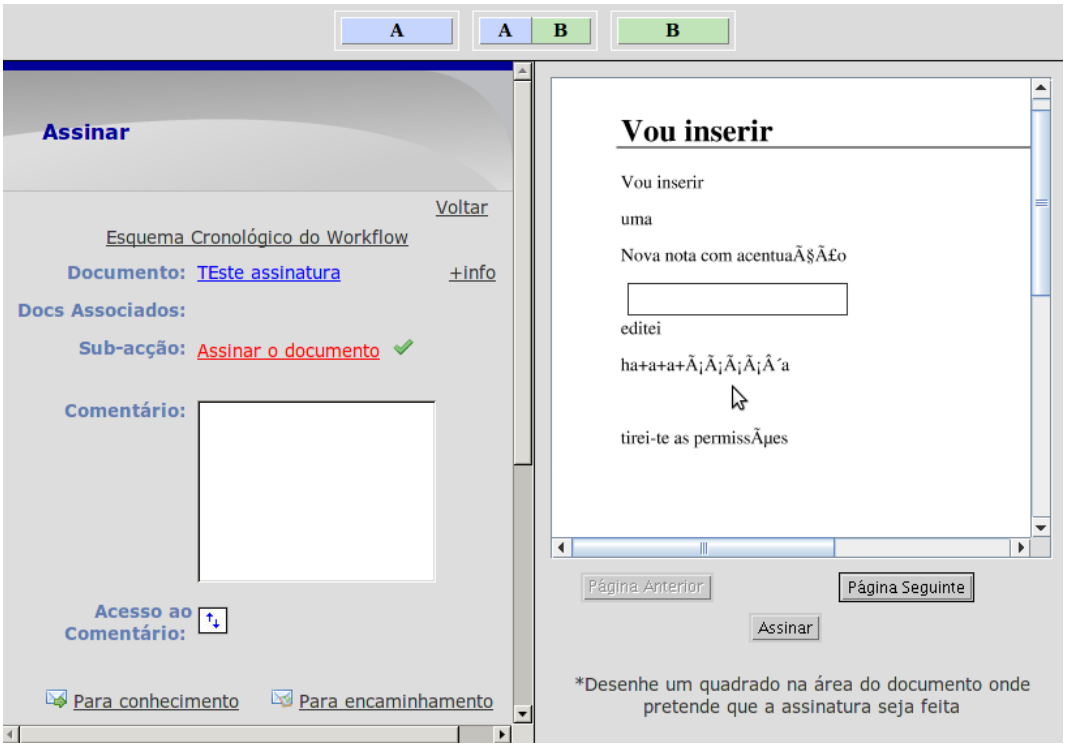

Figura 2.9 - Acção assinatura digital

Na figura 2.9 pode ser visto a interface onde o utilizador terá a possibilidade de assinar digitalmente um documento. Do lado esquerdo da interface é mostrado o menu correspondente à acção e do lado direito o documento no qual o utilizador irá inserir a assinatura digital.

2.2.2.3 - Análise de funcionamento

Neste ponto será feita uma análise do funcionamento do *iPortalDoc*. Inicialmente será feita uma descrição do funcionamento geral da aplicação, sendo de seguida feita uma apresentação geral ao modo de funcionamento das funcionalidades mais relevantes para o desenvolvimento deste projecto. Essas funcionalidades são o adicionar utilizador, criar template, criar *workflow* e o módulo de assinatura digital presente no *iPortalDoc*.

2.2.2.3.1 - Visão geral de funcionamento do iPortalDoc

De seguida é apresentado, de um modo geral, o funcionamento do *iPortalDoc*.

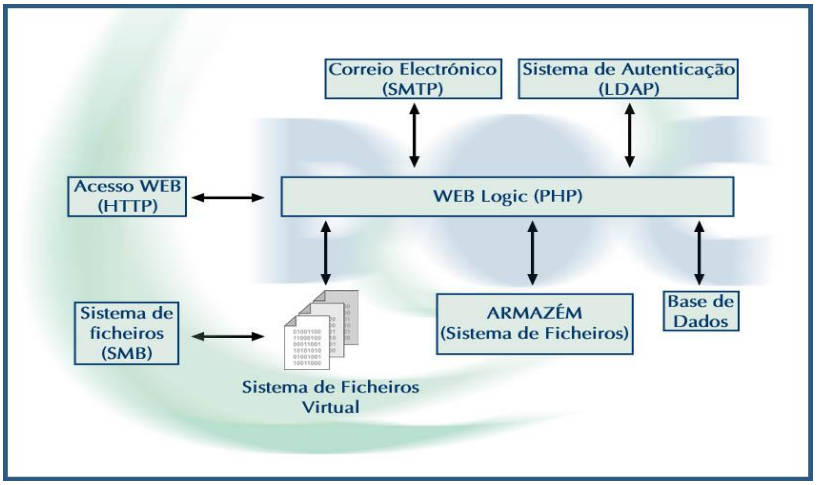

Figura 2.10 – Modelo de arquitectura do *iPortalDoc*

Como se pode ver na figura 2.10 o *iPortalDoc* encontra-se interligado com vários serviços *opensource*. Embora possa funcionar sobre qualquer sistema operativo, com vista a melhorar o seu desempenho e segurança, o *iPortalDoc* foi desenhado para funcionar em conjunto com o *iPbrick*. A base de dados utilizada está desenvolvida em *PostgreSQL*, o servidor *Web* utilizado é o Apache (servidor *Web* gratuito e mais utilizado no mundo), a gestão de ficheiros e arquivos é elaborada através de um servidor de ficheiros Samba, o servidor de correio electrónico usado é o SMTP e sistema de autenticação utilizado opera sobre o protocolo LDAP funcionando em parceria com o *iPBrick*.

2.2.2.3.2 - Adicionar utilizador

Para ser possível o acesso ao *iPortalDoc* um utilizador necessita de ser registado no *iPBrick* onde lhe serão atribuídos um conjunto l*ogin*/*password* de modo a poder aceder as diversas aplicações. Esta tarefa é desempenhada por um Administrador que procede à inserção e gestão dos utilizadores e respectivas entidades, através da aplicação IP Contactos.

Depois de registado no *IPBrick*, o utilizador deverá ser adicionado ao *iPortalDoc*. Para tal, um utilizador com permissões de *Super User* deverá aceder ao menu "Definições -> Utilizadores -> Gerir Utilizadores" onde terá acesso a uma interface que lhe permite atribuir/retirar acesso a utilizadores, como pode ser visto na figura 2.11. O *super user* terá também de atribuir um perfil de utilizador a cada utilizador ao qual concede determinadas permissões (*Super User*, Coordenador, Sub-Coordenador, Leitor Absoluto, Leitor, Editor e Navegador). A escolha do perfil de utilizador e respectivo nível de permissão, será o parâmetro que irá filtrar/limitar a quantidade de informação que será disponibilizada a cada utilizador no *iPortalDoc*.

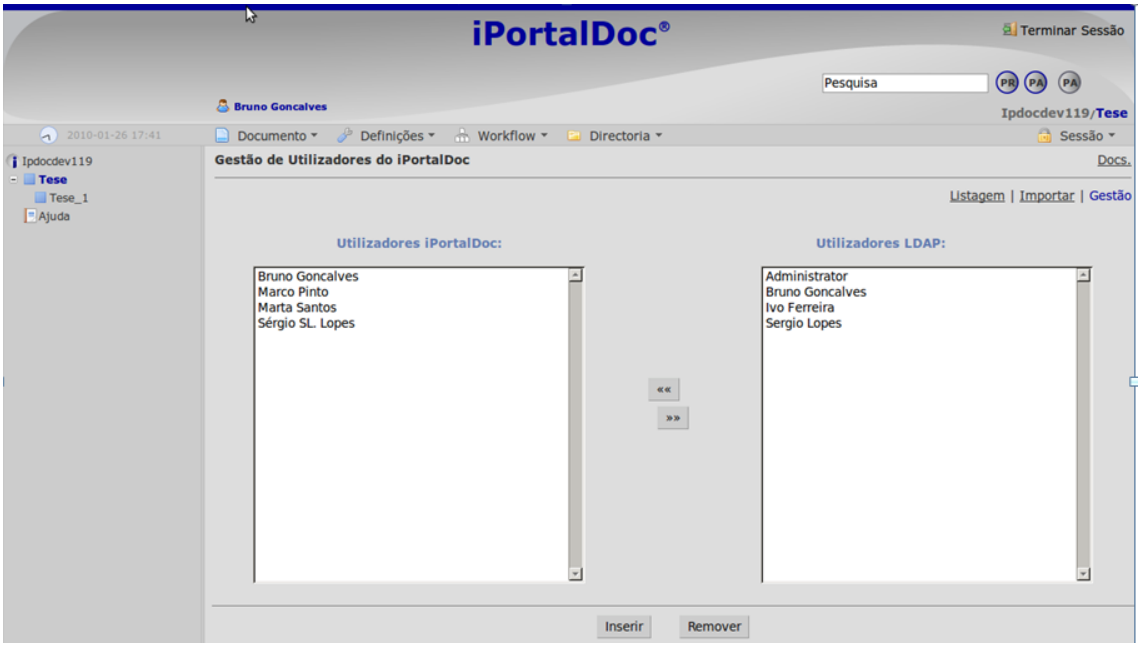

Figura 2.11 - Gestão de utilizadores no *iPortalDoc*

Depois de adicionado ao *iPortalDoc*, será necessário atribuir permissões ao utilizador de modo a que este possa ter acesso ao *iPortalDoc-Light*. Este passo terá que ser efectuado por um S*uper User* acedendo ao menu "Definições -> iPortalDoc-Light". Nesse menu será possível adicionar um utilizador ao *iPortalDoc-Light* e definir as permissões que este terá. As permissões a que um utilizador terá acesso no *iPortalDoc-Light* não são definidas por perfil, mas sim seleccionando o tipo e montante de informação que cada utilizador poderá ver.

2.2.2.3.3 - Criar template

De seguida, procede-se à explanação do desenvolvimento e utilização de *templates* de documentos. Na figura 2.12 é possível visualizarmos o diagrama de arquitectura correspondente à criação de um *template* no *iPortalDoc*.

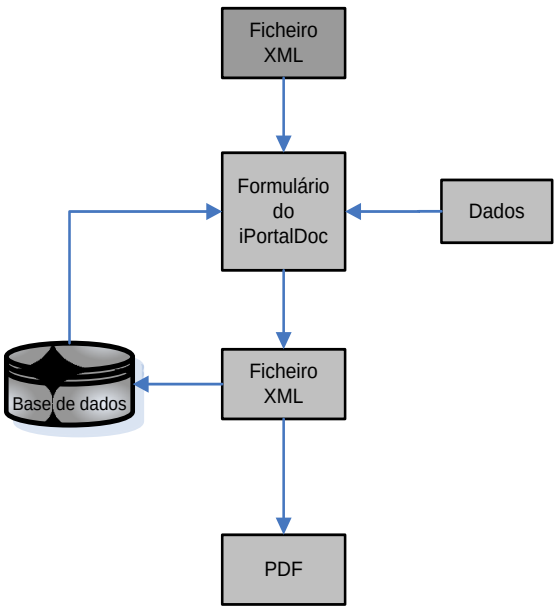

Figura 2.12 – Diagrama de desenvolvimento de um formulário

Um formulário consiste num ficheiro XML (*Extensible Markup Language*) que foi previamente criado, contendo um conjunto de campos que deverão ser preenchidos com os dados que cada utilizador pretenda inserir.

A simplicidade e usabilidade do XML no desenvolvimento de formulários facilitam a organização textual dos dados, permitindo, igualmente, que os formulários possam ser facilmente convertidos em outros formatos. Após o preenchimento do *template* XML com os dados inseridos pelo utilizador é criado e armazenado um outro ficheiro XML. Este pode depois ser transformado num ficheiro XSLT para ser convertido, posteriormente, para outro formato, como por exemplo PDF.

Na figura 2.13 é exemplificada a arquitectura da base de dados correspondente à inserção de formulários e correspondência a documentos.

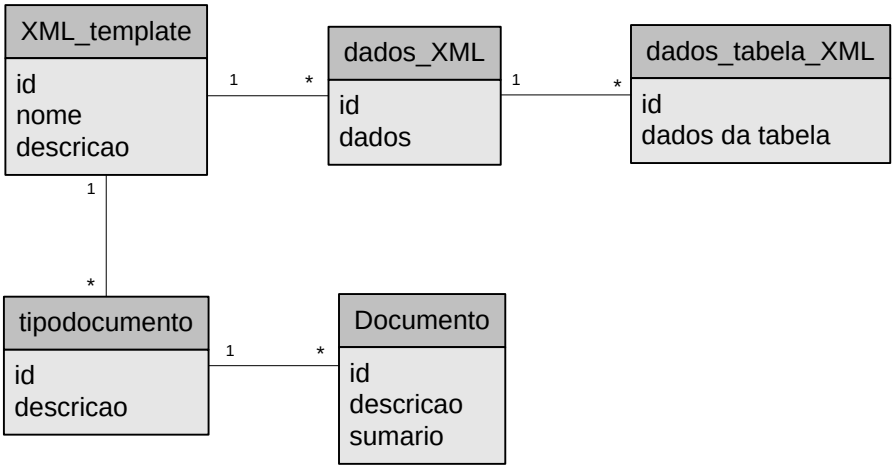

Figura 2.13 - Modelo relacional do formulário/documento

A tabela "XML_template" contém o nome e descrição referentes à elaboração de um *template*. Após o início do preenchimento do *template* de formulário, ser-lhe-ão adicionados vários dados que serão guardados na tabela "dados_XML". A cada conjunto de dados será associada uma tabela ("dados_tabela_XML") onde serão inseridos os dados correspondentes à organização do formulário criado. Cada *template* poderá ter associado a si vários tipos de documentos e cada tipo de documento poderá gerar vários outros documentos (isto pode ser visto nas associações entre as tabelas "XML_template"/"tipodocumento" e "tipodocumento"/"documento").

Os dados correspondentes a cada tipo de documento e ao documento que será criado estão contidos nas tabelas "tipodocumento" e "documento" respectivamente.

De seguida será abordada a arquitectura correspondente à aplicação *workflow*, focando principalmente a organização da base de dados que dá origem a este tipo de funcionalidade.

2.2.2.3.4 - Workflow

A gestão documental feita no *iPortalDoc* encontra-se directamente associada a *workflows*. Estes são elaborados em várias fases, seguindo uma série de etapas. Inicialmente, será

elaborado um *template* de *workflow*, de seguida é-lhe associado um utilizador e, por fim, um documento.

Posteriormente será descrita a arquitectura do desenvolvimento de um diagrama de *workflow*.

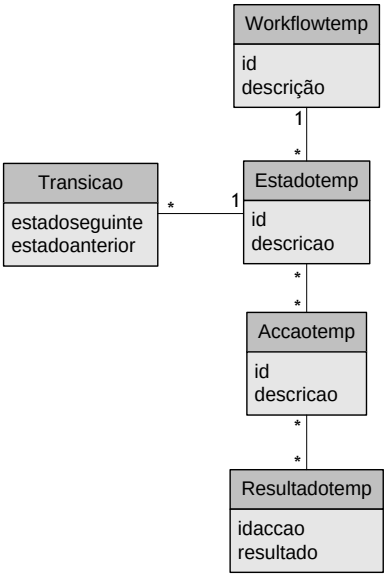

Figura 2.14 – Modelo relacional do módulo *workflow*

A figura 2.14 descreve o modelo relacional correspondente à elaboração de um *workflow*.

O primeiro passo na elaboração de um *workflow* corresponde à atribuição de um nome e uma descrição, informação que será guardada na tabela "Workflowtemp".

O *workflow* será baseado numa máquina de estados. Como tal, o próximo passo será a inserção de estados. Cada *workflow* poderá ter associado a si um ou mais estados, sendo cada um identificado por um *id* e uma descrição, dados estes que serão guardados na tabela "Estadotemp".

Depois de acrescentados os estados ao *template* serão adicionadas acções a estes. As acções a acrescentar consistem em tarefas que o estado terá que cumprir de modo a poder transitar para outro estado. Estas já se encontram pré-definidas, com tempos de execução e resultados já estipulados. Um estado poderá conter várias acções, sendo a informação relativa às acções guardada na tabela "Accaotemp". Por sua vez, cada acção poderá ter vários resultados que correspondem, cada um deles, à concretização da respectiva acção (com ou sem sucesso), responsáveis pelas transições entre estados. Os dados correspondentes aos resultados serão guardados na tabela "Resultadotemp".

A interligação entre estados é possível através do recurso a funções de transição. Caracterizam-se como ligações que descrevem as mudanças de estados e podem afectar estados consecutivos, intercalados ou antecedentes. Na tabela "transição" serão guardados os estados que cada transição trata.

2.2.2.3.5 - Assinatura digital

Para ser usada a funcionalidade de assinatura digital, recorrendo à assinatura electrónica do cartão do cidadão no *iPortalDoc* é necessário que o utilizador possua um cartão do cidadão, um leitor de cartões de cidadão e tenha instalado no seu computador uma *java*

virtual machine de modo a permitir que o seu computador consiga executar o código *Java* sobre o qual assenta esta funcionalidade.

Para ser adicionada uma assinatura digital a um documento, inicialmente, será necessário criar um *workflow* com uma acção "Assinar o documento com a assinatura electrónica do cartão do cidadão". Quando o utilizador tiver a necessidade de realizar a acção descrita em cima, ser-lhe-á apresentada a interface mostrada na figura 2.9 do capítulo anterior. Esta encontra-se dividida em duas partes distintas, do lado esquerdo é apresentado o menu correspondente à acção, do lado direito é mostrado o documento onde o utilizador poderá inserir a assinatura digital.

A interligação das funcionalidades do cartão do cidadão com o *iPortalDoc* foi realizada em linguagem *Java,* pois as funcionalidades inerentes ao cartão do cidadão assentam nesta linguagem. O uso de *Java* para a interligação com o cartão do cidadão permitiu também facilitar o desenvolvimento da interface de inserção da assinatura digital, recorrendo a applets *Java* para a interligação com a interface *Web*.

2.2.3 - iPortalDoc-Light

Embora os módulos a desenvolver estejam directamente relacionados com o *iPortalDoc*, pois as interfaces de configuração serão desenvolvidas para funcionar na interface de configuração do *iPortalDoc-Light* que se encontra no menu “Definições” do *iPortalDoc*, será sobre o módulo externo que todas as aplicações desenvolvidas terão impacto.

Em seguida, será realizada uma exposição do *iPortalDoc-Light*, sendo inicialmente efectuada uma apresentação deste módulo e, posteriormente, realizada uma descrição das funcionalidades já disponibilizadas por este.

2.2.3.1 - Apresentação

Tal como acontece com todas as aplicações de gestão documental existentes no mercado, o *iPortalDoc* restringe o acesso aos utilizadores da empresa, decorrente das necessidades de segurança e confidencialidade da informação que uma determinada empresa produz/ recebe/ armazena. Esta medida é fulcral no quotidiano de qualquer organização, permitindo uma maior fiabilidade e autenticidade da informação que é disponibilizada.

No entanto, existem excepções no que diz respeito à divulgação da informação, quer a nível interno, quer a nível externo e é importante existirem sistemas que contemplem estas crescentes e complexas necessidades das empresas.

No decorrer destas exigências do contexto empresarial actual, o *iPortalDoc* desenvolveu uma aplicação que cumprisse determinados requisitos no que diz respeito às trocas de informação que poderão não se circunscrever somente ao interior de determinada empresa. Esta preocupação tinha em mente a partilha de informação, muitas vezes necessária, entre empresa – cliente – parceiro, nunca menosprezando o nível de exigência e de segurança que uma aplicação destas requer. É criado, assim, o *iPortalDoc Light*.

O *iPortalDoc-Light* é uma interface *Web* que permite que entidades externas à empresa possam ter acesso a uma determinada quantidade informação contida no *iPortalDoc*. Para o

efeito, são estabelecidos critérios de acesso, definindo que entidade tem acesso a que informação. É possível ainda a introdução e o encaminhamento de documentos, conduzindo a uma maior autonomia no acesso à informação por parte de terceiros, desde que devidamente autorizados.

2.2.3.2 - Funcionalidades

Para aceder a este módulo, o utilizador terá de se autenticar com recurso a *Login/password*. As permissões que cada utilizador externo terá ao aceder ao *iPortalDoc-Light* são-lhe atribuídas por um administrador do *iPortalDoc*, aquando da sua configuração e são restringidas pelos seguintes campos: "Tipo de Entidade"; "Entidade"; "Tipo de documento"; "Assunto"; "Projecto". Esta atribuição irá filtrar, desde logo, a quantidade de informação que será disponibilizada a determinado utilizador externo.

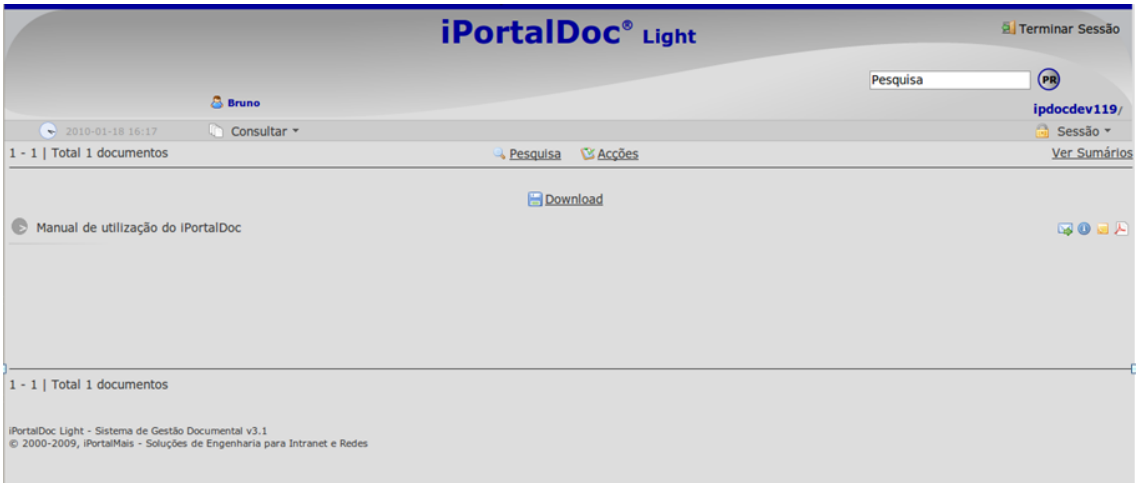

Figura 2.15 -Interface principal do *iPortalDoc-Light*

Depois de devidamente autenticado, o utilizador externo terá acesso à interface principal do *iPortalDoc-Light*, como se pode ver na figura 2.15. Nesta interface, o utilizador externo terá acesso a:

- **Hierarquia de documentos**: em tudo semelhante à apresentada no *iPortalDoc*, tendo, contudo, os documentos classificados consoante os campos que lhe foram atribuídos;
- **Listagem de acções pendentes**: é apresentada uma listagem das acções atribuladas aos documentos a que tem acesso;
- **Introdução de templates**: o utilizador terá, consoante as permissões que lhe forem atribuídas, a possibilidade de introduzir macros de documentos;
- **Menu de pesquisa rápida:** permite ao utilizador realizar uma pesquisa rápida no directório onde se encontra e segundo as palavras-chave que desejar;
- **Encaminhamento e download de ficheiros**: o utilizador poderá encaminhar documentos em modo *adhoc* para utilizadores internos do *iPortalDoc* e pode fazer o download dos documentos a que tiver acesso;
- **Consulta de informação de documentos**: será possível ao utilizador visualizar a informação de cada documento, bem como o *workflow* associado a esse documento.

Visto este módulo ter como principal vantagem o fornecimento de informação a entidades externas, a segurança desta aplicação é intensificada se esta for integrada com uma *IPBrick* de comunicação (*IPBrick.C*) dado que esta funciona como servidor de comunicação entre a empresa e o exterior.

No ponto 3.1 do capítulo 3 é feita uma descrição da arquitectura de funcionamento, das aplicações do iPortalDoc-Light, relevantes ao desenvolvimento deste projecto.

2.3- Linguagens de programação

Como foi referido no inicio do capítulo, neste ponto será feita um levantamento das linguagens de programação utilizadas no desenvolvimento do projecto. Foi também realizada uma comparação entre as possíveis soluções de modo a diferenciar possíveis alternativas e melhorias ao nível de processos e tecnologias utilizadas na implementação, tendo como objectivo final a melhoria do funcionamento e eficácia.

2.3.1 - PHP/Perl

O PHP (*Hypertext Preprocessor*) é uma linguagem de programação que deriva do PHP/FI (*Personal Home Page Tools*) por Rasmus Lerdof em 1995 que consistia num simples conjunto de script Perl. Em 1997, Andi Gutmans e Zeev Suraski lançaram o PHP3. Esta nova versão é a que mais se assemelha à utilizada actualmente. Das grandes vantagens do PHP3 salienta-se a sintaxe orientada a objectos, elevada expansibilidade e a integração com várias bases de dados, o que levou ao grande sucesso desta versão. Actualmente o PHP encontra-se na versão 5 e tem como principal objectivo o desenvolvimento de páginas *Web*, ultrapassando já o ASP.NET da Microsoft ao ser usado em mais de 20 milhões de páginas *Web,* em 2007. A sua rapidez, flexibilidade e simplicidade torna-a uma linguagem muito apreciada por programadores e designers [9].

Uma alternativa ao PHP é o PERL (Practical Extraction and Reporting Language). O PERL uma linguagem de programação criada em 1987 como linguagem de *script* para sistemas Unix que deriva principalmente da linguagem de programação C e da *shell* de *Unix*. As características do PERL tornam-na uma linguagem apropriada para tarefas que envolvam, entre outras, tarefas de gestão de sistemas, acesso a base de dados e programação na *Web*. Estas vantagens tornam-na uma linguagem muito popular entre administradores de sistema, autores de *scripts* CGI, etc. O PERL é uma linguagem que é distribuída gratuitamente, sendo assim suportada pelos seus utilizadores que ajudam assim a melhorar bibliotecas, módulos e documentação [10].

Ao contrário do PHP que é uma linguagem mais orientada ao desenvolvimento de páginas *Web*, o PERL é uma "verdadeira" linguagem de programação. No que toca ao desenvolvimento de páginas *Web* o Perl usa *scripts* CGI que criam páginas HTML, ao contrário do PHP em que o código é inserido no HTML e corrido quando a página é chamada. Duas das grandes vantagens do PERL no desenvolvimento de aplicações *Web* são o seu interpretador para Apache e a arquitectura baseada em componentes [10], estas vantagens aliadas à sua história e robustez, tornam o PERL uma opção vantajosa no desenvolvimento de páginas *Web*.

Foram realizados alguns estudos comparando a performance de linguagem *Web* como o PHP, PERL e JAVA, de onde se concluiu que o PHP se comporta muito bem para pedidos pequenos mas que tecnologias de servidores JAVA se comportam melhor que PHP ou PERL, no entanto estas duas linguagem são bastante robustas quando os conteúdos não são demasiado grandes [11].

Poderemos referir que as causas que levaram à escolha do PHP para desenvolvimento dos módulos prendem-se com o facto de esta ser uma linguagem relativamente simples e rápida [9], da possibilidade de integração com qualquer base de dados e o facto de ser vocacionada para o desenvolvimento de páginas *Web*.

2.3.2 - JavaScript

Ao longo dos últimos anos, a *World Wide Web* passou a ser amplamente utilizada como meio de apresentação para vários serviços e aplicações, o que levou à procura de um método que tornasse estes serviços e aplicações mais interactivos e dinâmicos. Como resposta a essa necessidade surgiu o *JavaScript*.

A *JavaScript* é uma linguagem *client-side* orientada a objectos que foi desenvolvida pela Netscape. A sintaxe básica é similar a Java ou C, no entanto esta é uma linguagem mais orientada a servidores e páginas *Web*. Ao contrário do HTML e CSS, o *JavaScript* é uma verdadeira linguagem de programação, o que a torna uma ferramenta essencial para criar páginas *Web* modernas e interactivas. Este permite que o *browser* responda automaticamente sem ter necessidade de voltar a carregar a página. Para que tal aconteça, é necessário que a linguagem seja capaz de correr no mesmo computador onde se encontra o browser.

Nos últimos tempos, porém, a *JavaScript* evoluiu para se tornar muito mais útil e funcional, com o conceito de AJAX, trazendo um novo nível de interactividade com a programação *Web-based*. AJAX significa *Asynchronous JavaScript* and XML, embora a referência ao XML não é mais válida como requisito *Ajax* pode retornar respostas em formatos diferentes de XML, por exemplo, JSON (*JavaScript Object Notation*). *Ajax* funciona permitindo que o *JavaScript* envie um pedido HTTP para o servidor, e processe a sua resposta sem actualizar ou abrir uma nova página. Em vez disso, o programador deve usar DOM. O DOM (*Document Object Model*) é uma interface de linguagem que permite a programas e *scripts* aceder dinamicamente e actualizar conteúdo, estrutura e estilo de documentos permitindo manipular e modificar o campo da página *Web* de modo a reflectir as alterações ou dados retornados pela resposta HTTP. Antes do AJAX, qualquer processamento *server-side* ou o acesso a uma base de dados exigiria que a página inteira fosse "actualizada", ou uma nova página aberta para o pedido ser processado pelo *browser*. Isto é não só lento e frustrante para o usuário, mas também um desperdício de recursos [12].

Com o aumento exponencial do uso do *JavaScript*, tornou-se essencial o desenvolvimento de um modo mais fácil e prático de desenvolver interfaces bem como de fornecer suporte aos vários *browsers* (tal como no HTML e CSS diferentes *browsers* têm diferentes implementações de *JavaScript*, como tal, torna-se por vezes complicado desenvolver uma aplicação que seja compatível com todos eles) e assim surgiram várias *JavaScript frameworks (frameworks* são um conjunto de funções pré-escritas que visam simplificar o desenvolvimento tratando das tarefas que habitualmente consomem mais tempo [13]. De entre as várias *frameworks* existentes são de salientar a jQuery, YUI, Prototype e mooTools).

Como se pode ver na tabela 2.1, todas as *frameworks* são compatíveis com os diferentes tipos de *browser*, no entanto em versões mais antigas de *browsers* não são suportadas, o que pode levar ao aparecimento de problemas no uso de *frameworks* nessas versões. Podemos também verificar que todas as *frameworks* se revelam bastante completas no que toca ao uso de AJAX, DOM, JSON sendo que, todas elas suportam este tipo de características. No que toca a melhoramentos a nível de utilizador, UX (*user experience*) e UI (*user interface*), apenas a YUI integra este tipo de características todas as outras necessitam de *add-ons* para as implementar, sendo que a MooTools se encontra bastante incompleta no que toca a este tipo de característica.

Tabela 2.1 - Comparação entre *frameworks* [12]

	Prototype	**jQuery**	**YUI**	**MooTools**
Ultima versão	1.6.1	1.4.1	3.00	1.2.4
Licença	MIT	MIT & GPL	BSD	MIT
Compatibilidade entre browsers				
IE	6.0+	6.0+	6.0+	6.0+
Firefox	1.5+	2.0+	3.0+	2.0+
Safari	2.0.4+	3.0+	4.0+	2.0+
Opera	9.25+	9.0+	10.0+	9.0+
Chrome	1.0+	1.0+	Não verificada	Não verificada
Características do núcleo				
Suporte para Ajax	Sim	Sim	Sim	Sim
Manipulação DOM	Sim	Sim	Sim	Sim
DOM Transversal	Sim	Sim	Sim	Sim
Manipulação de eventos	Sim	Sim	Sim	Sim
JSON	Sim	Sim	Sim	Sim
Selectores	Sim	Sim	Sim	Sim
Melhoramentos UX/UI				
Animações	scriptaculous	Sim	Sim	Sim
Auto Completion	scriptaculous	Não	Sim	Não
Histórico	scriptaculous	Não	Sim	Não
Calendário	Não	jQuery UI	Sim	Não
Drag and Drop	scriptaculous	jQuery UI	Sim	MooTools More
Grelhas	Não	Não	Sim	MooTools More
Barra de progresso	Não	jQuery UI	Sim	Não
Redimensionamento	Não	jQuery UI	Sim	Não
Editor de texto	Não	Não	Sim	Não
Slider	scriptaculous	jQuery UI	Sim	MooTools More
Tabs	Não	jQuery UI	Sim	Não
Temas	Não	jQuery UI	Sim	MooTools More
Vista em árvore	Não	Não	Sim	Não

Visto o *iPortalDoc* tratar-se de uma aplicação *Web* que tem como finalidade fornecer uma vasta gama de serviços a variados tipos clientes, faz com que o uso de *JavaScript*, *frameworks JavaScript* e AJAX seja indispensável, de modo a tornar o *iPortalDoc* uma ferramenta interactiva, actualizada e de fácil utilização.

2.3.3 - HTML

HTML (HyperText Markup Language) trata-se de uma linguagem usada para criar páginas *Web* e foi desenvolvido por Tim Berners-Lee, em 1990. O objectivo inicial era tornar possível a troca de informação e documentação de pesquisas, entre cientistas. Este possibilita apresentar informações através da internet. Actualmente, é utilizada para a produção de páginas na *Web*. Ou seja, o HTML fornece os meios para descrever a estrutura da informação baseada em texto de um documento e realiza isso dotando, por exemplo, texto como hiperligações, cabeçalhos, parágrafos, listas ou acrescentando a esse texto formatos interactivos, imagens ou outros objectos. Com efeito, os seus documentos são interpretados por browsers que entendem e transcrevem as formatações da página para a tela. Um documento HTML é escrito sobre a forma de *tags*. O HTML pode também, embora de forma limitada, descrever a aparência e a semântica de um documento além de poder embeber pequenos códigos escritos noutras linguagens, *scripts*, como por exemplo *JavaScript* [14].

Presentemente, encontra-se a ser desenvolvido o XHTML (Extensible Hypertext Markup Language), o qual usa uma sintaxe mais rigorosa e menos ambígua para que o código HTML se torne mais simples para ser processado.

Sendo simples arquivos de texto, podem ser criados e editados em qualquer editor de texto comum.

2.3.4 - Base de dados

A crescente necessidade de armazenamento de informação leva a um aumento exponencial da utilização de bases de dados. Um sistema de base de dados permite armazenar e organizar dados, possibilitando uma recuperação rápida e eficaz da informação, tornando-o, desta forma, uma ferramenta fundamental no desenvolvimento e funcionamento de inúmeras aplicações (e.g. um sistema de gestão documental).

De entre os vários tipos de bases de dados, salientam-se como as mais utilizadas o *PostgreSQL*, *MySQL* e *Oracle*. Destas, apenas a Oracle necessita de licença para a sua utilização. Quer o *MySQL*, quer o *PostreSQL* são gratuitos e *open source*, como a grande maioria dos sistemas de bases de dados. A linguagem mais utilizada para definição de dados e *queries* para efectuar pedidos à base de dados é a SQL (Structured Query Language).

Dos sistemas de base de dados referidos acima aquele que será de maior relevo para o desenvolvimento deste projecto é o *PostrgreSQL*. Este conta com mais de 15 anos de desenvolvimento e uma arquitectura que lhe valeu a reputação de confiável, fiável e correcto. O *PostgreSQL* é actualmente utilizado em todos os maiores sistemas operativos. O *PostgreSQL* fornece diversos recursos como consultas complexas, integridade de dados usando

chaves primárias e estrangeiras, *triggers*, vistas, suporte para *subqueries*, herança singular ou múltipla de tabelas (permite derivar tabelas de outra tabelas), entre outros [9].

As vantagens descritas, aliadas ao facto do *PostgreSQL* ser gratuito e possível de usar, distribuir e modificar de acordo com as intenções de cada utilizador fazem do *PostgreSQL* um sistema muito vantajoso para o desenvolvimento e gestão de uma base de dados.

Tal é o que acontece no *iPortalDoc* e *IPBrick*. Estes softwares utilizam o *PostgreSQL* como principal sistema de base de dados tendo, no entanto, suporte para outros sistemas como *MySql* e *Oracle*.

2.4- Conclusões

Actualmente, os sistemas de SGDW são uma ferramenta crucial para o bom funcionamento de uma empresa, podendo mesmo vir a tornar-se obrigatório num futuro próximo. Como tal, torna-se indispensável que o SGDW seja uma aplicação interactiva e funcional, dado que fará parte do quotidiano de um grande número de indivíduos e instituições.

Depois de apresentadas e analisadas algumas soluções na área do desenvolvimento *Web,* concluiu-se que o PHP é a linguagem de eleição, pois foi inicialmente projectada com esse intuito. Em conjunto com o *JavaScript,* é essencial ao desenvolvimento de uma interface *Web* moderna e interactiva, recorrendo ao uso de *frameworks* e assegurando o bom funcionamento nos diferentes *browsers*.

No que concerne a *frameworks* a utilizar, embora a *jQuery* se apresente como uma boa solução - principalmente pela quantidade e qualidade de documentação e exemplos disponíveis - a escolha recaiu sobre a *Prototype* e *YUI*. Como pode ser visto nas tabelas 2.1, são ambas *frameworks* escaláveis, robustas e com vasta gama de funcionalidades.

No entanto, a principal razão de escolha foi o facto de estas *frameworks* estarem já a ser utilizadas no desenvolvimento do código *iPortalDoc* e *iPortalDoc-Light*, sendo, portanto, desnecessário sobrecarregar o código com mais uma *framework*.

Como anteriormente referimos, uma base de dados é indispensável ao funcionamento de um *software* deste tipo e, tal como se procedeu em relação à escolha da *framework*, o factor que mais contribuiu para a escolha do *PostgreSQL* foi o facto de a empresa já trabalhar com este sistema de base de dados.

Capítulo 3

Análise e Concepção

Neste capítulo, será apresentada uma especificação dos requisitos necessários para a implementação de cada módulo.

Depois desta breve introdução, a secção 3.1 será dedicada a abordar arquitectura de cada sistema utilizado; primeiramente, será realizado um apanhado da interligação entre os diversos sistemas, sendo, em seguida, descrito o modo de funcionamento dos principais módulos já existentes, que serão necessários para o desenvolvimento do projecto.

Nas secções 3.2 a 3.5 é apresentada uma descrição do planeamento de cada módulo. Em cada secção será abordado um módulo distinto, onde serão especificados o modelo de dados, casos de uso, requisitos funcionais necessários e será delineada uma solução preliminar para o desenvolvimento de cada um dos módulos.

Este capítulo permitirá também concluir as modificações que serão necessárias efectuar a nível de interface e base de dados, para o desenvolvimento de cada módulo.

3.1- Módulos existentes

Neste ponto, será efectivada uma descrição da arquitectura geral do sistema, onde se encontram integrados o *iPortalDoc* e o *iPortalDoc-Light*.

Inicialmente, será feita uma descrição da arquitectura geral de sistema, focando a interligação entre aplicações, sendo, então, apresentada uma breve explicação do funcionamento, no *iPortalDoc-Light* dos módulos relevantes para a implementação deste projecto, o módulo de inserção de *templates* e o de *workflows*.

3.1.1 - Visão global do sistema

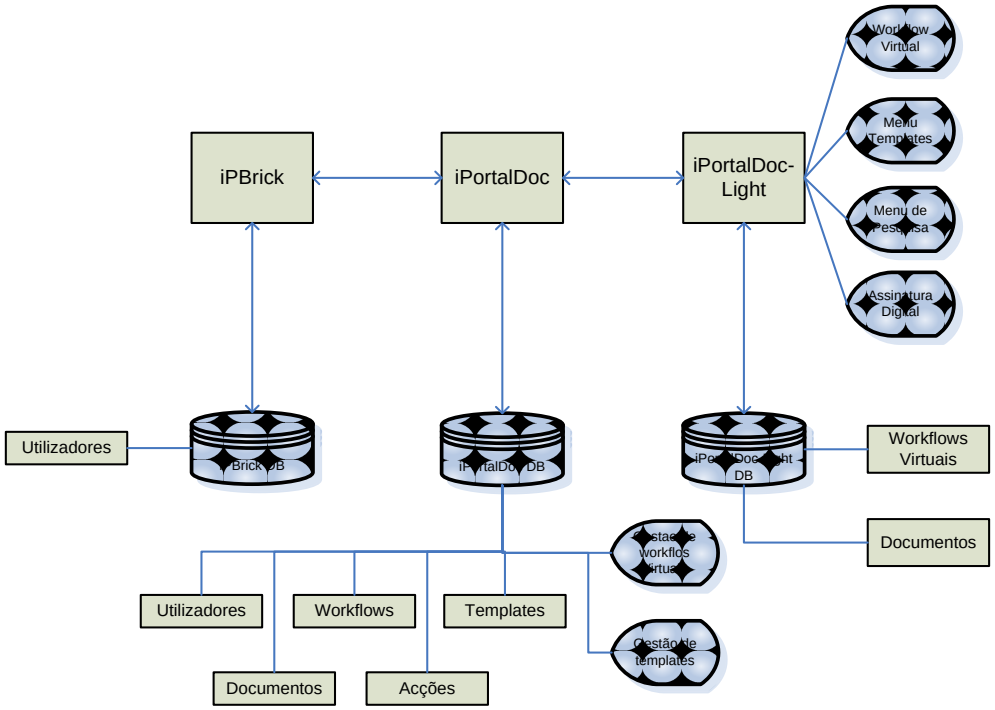

Figura 3.1 – Interligação entre módulos

Na figura 3.1, é exemplificada a interligação entre os diferentes módulos e as respectivas bases de dados.

Como foi descrito no ponto 2.2.2.3.2, a gestão de utilizadores do *iPortalDoc* é feita com recurso ao *IPBrick*. Através do *IPContactos* (aplicação que permite efectuar a gestão de utilizadores), um administrador poderá adicionar utilizadores, entidades, etc ao *iPortalDoc*. A gestão dos utilizadores que terão acesso ao *iPortalDoc-Light* é, por sua vez, efectuada no *iPortalDoc*, existindo uma interface de gestão desenhada especialmente para essa funcionalidade.

O *iPortalDoc-Light* funciona presentemente como um módulo integrado no *iPortalDoc*, sendo que a informação disponibilizada (*workflows*, documentos, *templates*, etc.) na interface deste módulo é obtida, maioritariamente, acedendo à base de dados do *iPortalDoc*.

Tendo como objectivo futuro transformar o *iPortalDoc-Light* numa ferramenta "independente" do *iPortalDoc*, o uso da base de dados, bem como o desenvolvimento do código, deverá ser realizado, o máximo possível, no *iPortalDoc-Light*.

Como pode ser observado no diagrama da figura 3.1, todos os módulos a desenvolver terão como objectivo final expandir funcionalidades que serão integradas com o *iPortalDoc-Light*, porém, visto toda a configuração ser executada no *iPortalDoc*, os menus de gestão dos módulos de *workflow* virtual e inserção de *templates* serão desenvolvidos no *iPortaldoc*.

3.1.2 - Inserção de templates

Quanto aos *templates* (*macros*) a que um utilizador tem acesso no *iPortalDoc-Light*, estes serão os mesmos que são disponibilizados no *iPortalDoc*. Contudo, um utilizador apenas poderá inserir *macros* a que tenha acesso. Acesso este que é atribuído ao utilizador no *iPortalDoc*, no menu de configuração do *iPortalDoc-Light*.

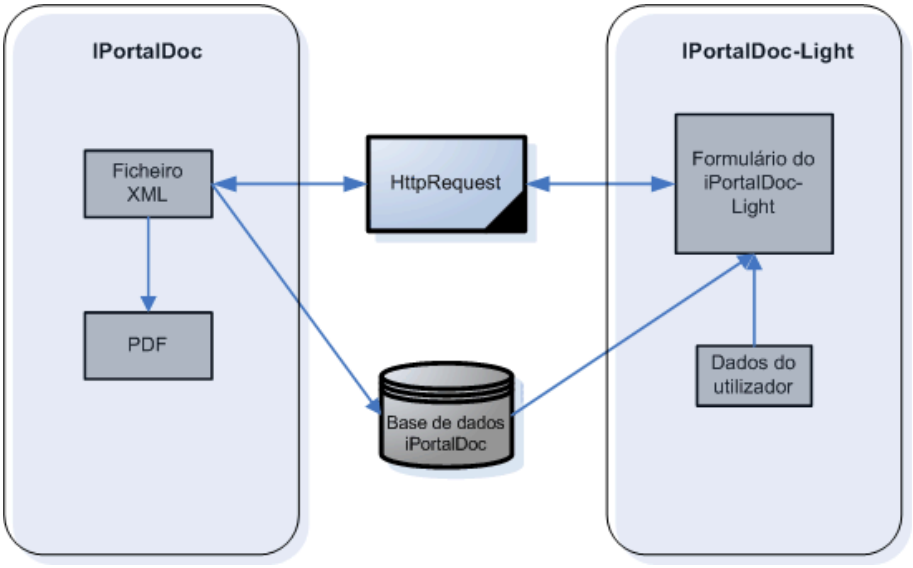

Figura 3.2 - Inserção de templates no *iPortalDoc-Light*

Na figura 3.2, podemos observar um diagrama que retrata a forma como a inserção de *templates* é realizada no *iPortalDoc-Light*. Os campos necessários à geração do formulário da macro (documento/*template*) são retirados da base de dados do *iPortalDoc*, criando o documento que o utilizador deverá preencher, depois de o utilizador inserir os dados que deseja, estes são submetidos por *HttpRequest* e processados no *iPortalDoc*, onde é criando o ficheiro XML correspondente. Por fim, é recebida no *iPortalDoc-Light*, a confirmação da inserção do documento.

3.1.3 - Visualização de *Workflows*

No que diz respeito aos *workflows*, é permitida, no *iPortalDoc-Light*, a consulta do *workflow* associado a um determinado documento. O *workflows* é em tudo idêntico ao fornecido pelo *iPortalDoc*, sendo que todas as etapas do seu ciclo de vida estão actualmente disponíveis para visualização por parte do utilizador. A informação que é disponibilizada é apenas filtrada pelo documento a que o utilizador tem acesso, ou seja, este poderá ver toda a informação contida no *workflow* de um documento que tenha permissões de visualizar.

Os dados relativos aos *workflows* que podem, actualmente, ser visualizados no *iPortalDoc-Light* encontram-se armazenados na base de dados do *iPortalDoc*.

3.2- Módulo de pesquisa

Nesta secção, será realizada a descrição da metodologia usada para a concepção do módulo de pesquisa avançada.

Inicialmente serão apresentados os requisitos funcionais que o módulo deverá cumprir, sendo de seguida apresentada uma descrição da arquitectura geral do sistema e será mostrada uma solução preliminar a ser seguida no desenvolvimento.

3.2.1 - Requisitos

De seguida serão apresentados os requisitos que o módulo deverá cumprir.

- O utilizador deverá ter a possibilidade de escolha entre um menu de pesquisa rápida ou pesquisa avançada, consoante as suas necessidades e especificidades;
- No menu de pesquisa avançada, deverá ser possível ao utilizador, efectuar uma procura pelos seguintes temas: assunto, entidade, tipo de documento, *workflow*, estado do *workflow*, data de introdução, data de elaboração, código, localização, título, sumário, descrição e/ou palavra-chave;
- Caso não seja obtido nenhum valor numa pesquisa deverá ser dada a possibilidade de efectuar uma nova pesquisa ou de sair.

3.2.2 - Arquitectura

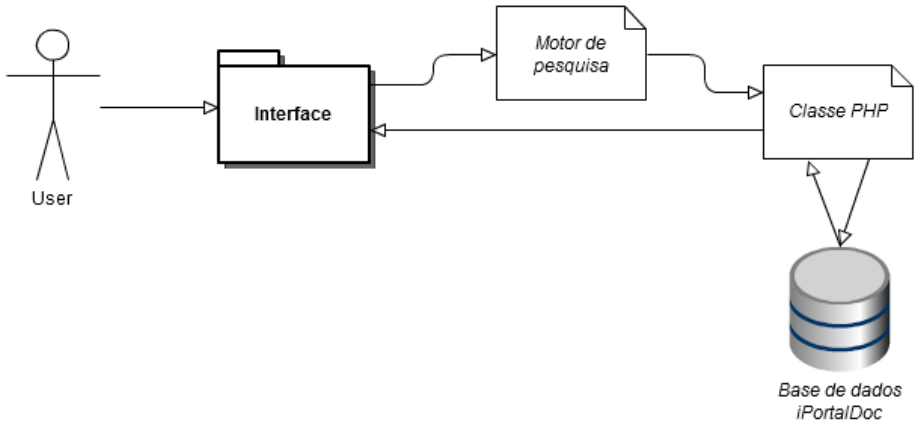

Figura 3.3 - Diagrama de arquitectura lógica do menu de pesquisa

Na figura 3.3, podemos observar uma descrição da arquitectura lógica do módulo de pesquisa. A primeira camada, de interface, é constituída pela interface com que o utilizador irá interagir, a segunda camada, camada lógica, contém os ficheiros que irão tratar os dados introduzidos pelo utilizador e, por último, na camada de dados irão ser encontradas as classes PHP responsáveis por realizar as *queries* à base de dados. Este tipo de modelo de arquitectura é aplicável, não só a este módulo, mas também aos módulos de *workflow* e *templates*.

Visto tratar-se de um módulo relativamente simples, os casos de uso ambicionados são relativamente simples. A aplicação deverá fornecer ao utilizador a possibilidade de escolher realizar uma pesquisa avançada. Depois de escolhida, a aplicação deverá fornecer ao utilizador um conjunto de campos que este possa preencher segundo os requisitos da pesquisa que pretenda efectuar. Os resultados da pesquisa deverão recair somente sobre o conteúdo que o utilizador tenha permissão de visualizar e deverá ter a possibilidade de realizar novas pesquisa depois de serem mostrados os resultados.

Na figura 3.4, é possível visualizar o diagrama de fluxo deste módulo, mostrando, de uma forma não muito detalhada, uma visão geral do sistema.

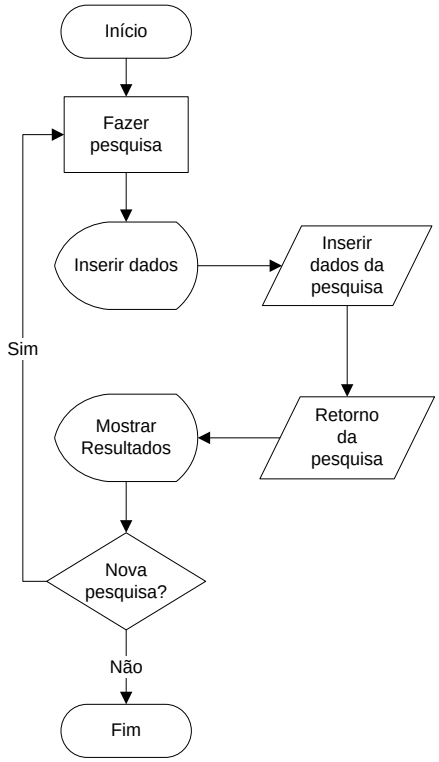

Figura 3.4 - Diagrama de Fluxo do módulo de pesquisa

Para efectuar uma pesquisa, o utilizador necessitará de introduzir os dados relativos à pesquisa que pretenda efectuar; em seguida, esses dados são processados e são-lhe retornados os resultados. Caso o utilizador queira efectuar uma nova pesquisa, é-lhe dada essa possibilidade. No capítulo 5, ponto 5.1, são mostradas as interfaces desenvolvidas para este módulo, bem como uma descrição mais detalhada do seu funcionamento.

Em relação à base de dados, não será necessária qualquer alteração em qualquer uma delas, visto ser apenas necessário, para a implementação deste módulo, efectuar *queries* a tabelas já existentes.

3.2.3 - Modelo de dados

Na figura 3.5 é possível ver o diagrama de dados correspondente ao menu de pesquisa.

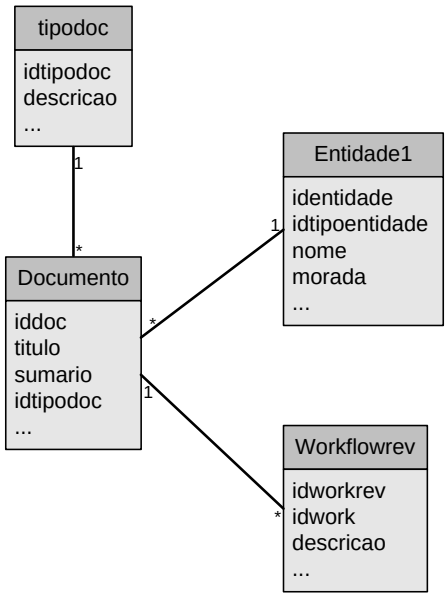

Figura 3.5 - Menu de dados do módulo de pesquisa

Embora para o desenvolvimento do módulo de pesquisa não seja necessário adicionar qualquer tipo de tabela à base de dados, este estará largamente interligado com a base de dados. No diagrama apresentado acima podemos ver as tabelas principais sobre as quais incidirão as pesquisas, consoante a pesquisa que o utilizador pretenda efectuar, serão realizadas *queries* à tabela pretendida. A tabela principal por onde todas as *queries* passaram é a "documento", dado que é nesta tabela que se encontram os dados dos documentos que serão apresentados como resultado da pesquisa.

3.2.4 - Solução preliminar

Os pontos fulcrais desta funcionalidade centram-se na interface que permitirá realizar a pesquisa e no *link* que a chamará. Este último será uma simples imagem com um *link* para a interface de pesquisa. Esta será desenvolvida em HTML, focando os requisitos de pesquisa descritos no ponto 3.2.1. Os dados inseridos para pesquisa serão posteriormente submetidos para um ficheiro que o tratará e efectuará as queries necessárias, mostrando de seguida os resultados numa outra interface.

3.3- Módulo *workflow* virtual

Esta secção contém a explicação do trabalho realizado para a concepção do módulo *workflow* virtual.

Nos próximos pontos serão apresentados os requisitos funcionais que este módulo deverá satisfazer, os casos de uso que a aplicação terá, o modelo relacional do sistema mostrando as tabelas que irão ser adicionadas à base de dados e por fim será feito o planeamento da solução.

3.3.1 - Requisitos

De seguida serão enumerados os requisitos funcionais que este módulo deverá cumprir.

Administrador (iPortalDoc):

- Para cada *workflow* existente no *iPortalDoc*, deverá ser dada a possibilidade a um administrador de criar um *workflow* virtual equivalente;
- Deverá ser possível a um administrador visualizar o *workflow* original enquanto é criado o *workflow* virtual. O *workflow* original deverá apresentar o título de cada estado;
- Deverá ser possível a um administrador editar o nome do *workflow*, ou mesmo remover o *workflow* virtual;
- Deverá ser dada a possibilidade a um administrador de adicionar estados e funções de transferência de forma normal (clicando em um *link*) e com recurso a *drag and drop;*
- Quando introduzido um estado, deverá será necessária a identificação do mesmo através de uma descrição, coordenadas da posição onde o estado deverá permanecer (no caso do *drag and drop,* estas coordenadas são atribuídas automaticamente no "*drop*");
- Não deverá ser possível introduzir um estado ou função de transferência sem ter os campos importantes/relevantes (nome e posição) preenchidos;
- Deverá ser dada a possibilidade de associar a cada estado virtual criado um ou vários estados do *workflow* original;
- Depois de criado um estado ou uma função de transferência, um administrador deverá ter a possibilidade de modificar todo o conteúdo ou remover, tanto os estados, como as funções de transferência;
- A qualquer momento, o utilizador deverá poder sair, permanecendo guardados os dados inseridos até ao momento.

Utilizador (iPortalDoc-Light):

- No *iPortalDoc-Light*, um utilizador externo deverá ter a possibilidade de visualizar o *workflow* original, caso não esteja criado nenhum *workflow* virtual, ou o *workflow* virtual correspondente ao *workflow* original associado ao documento em questão;
- Assegurar compatibilidade com os vários browsers.

3.3.2 - Casos de uso

O *iPortalDoc* possui a opção de gestão de utilizadores. Esta opção permite gerir os recursos a que cada utilizador está autorizado, ou não, a aceder. Dentro da gestão de utilizadores deverão ser atribuídos níveis de permissões que limitaram as opções a que cada utilizador terá acesso. Depois de definido o nível de permissão de administrador para um utilizador, este deverá ter acesso a este módulo, podendo de seguida criar *workflows* virtuais.

Na figura seguinte, é mostrado um diagrama simplificado dos casos de uso que esta aplicação deverá fornecer aos utilizadores.

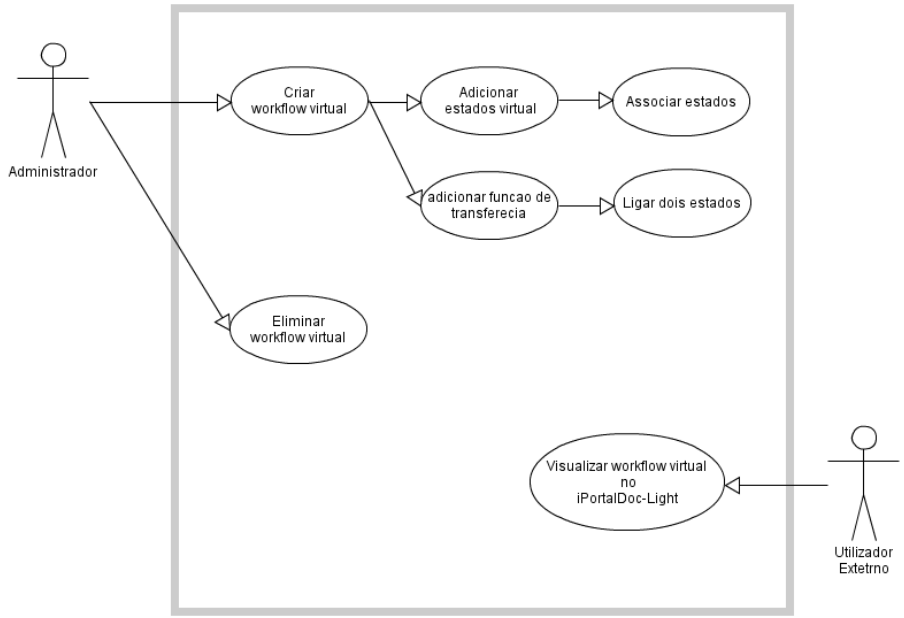

Figura 3.6 - Casos de uso módulo workflow

Os casos de uso deste módulo podem ser vistos na figura 3.5, em que um administrador terá como casos de uso:

- Criar *workflow* virtual;
- Adicionar estados virtuais;
- Adicionar função de transferência;
- Associar estados;
- Ligar dois estados;
- Eliminar *workflow* virtual.

Um utilizador do *iPortalDoc-Light*, depois de ter atribuídas a si as respectivas permissões, terá a possibilidade de visualizar o *workflow* virtual associado a um documento específico.

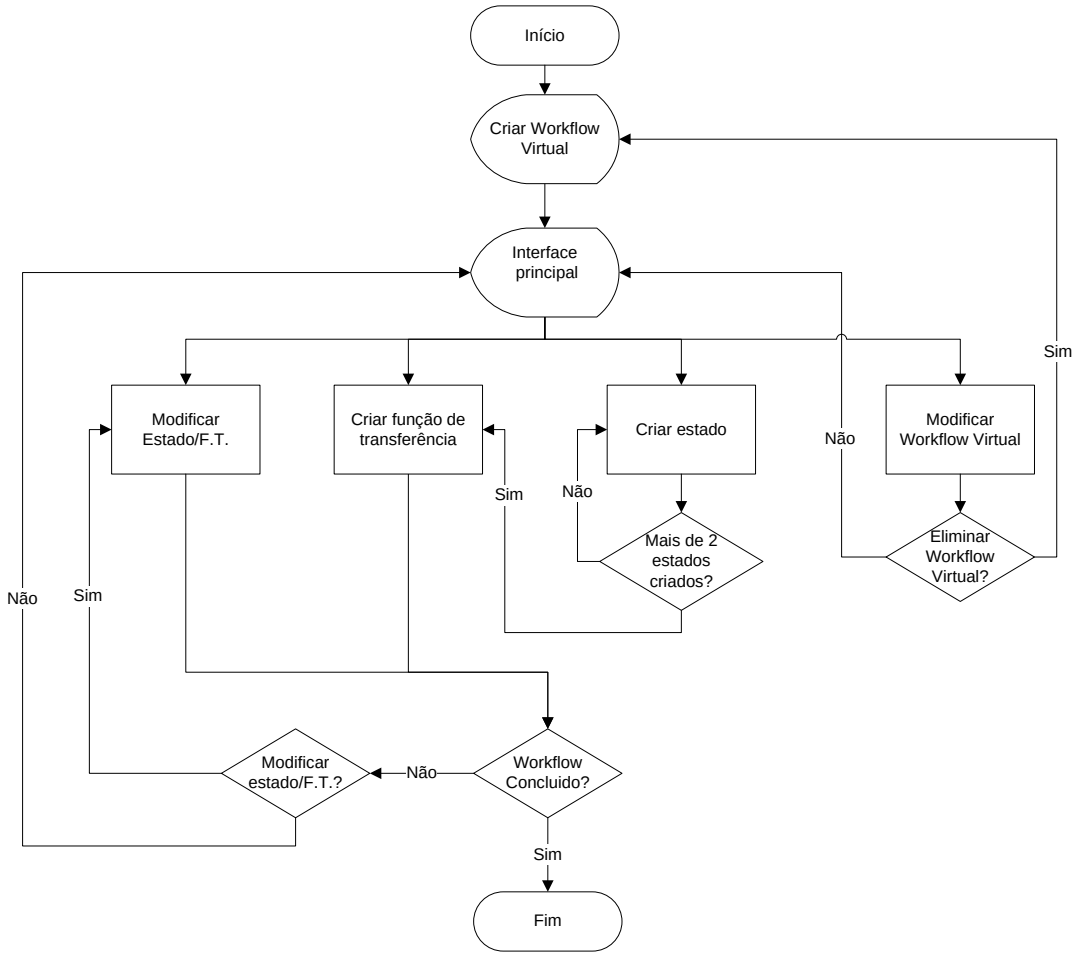

Figura 3.7 - Diagrama de Fluxo do módulo workflow

Na figura 3.6, é possível ver um diagrama de fluxo onde são detalhados, de forma simplificada, os fluxos de dados que este módulo deverá apresentar.

Depois da escolha inicial de criar *workflow* virtual, o utilizador terá acesso a um conjunto de menus onde poderá criar e modificar estados, criar e modificar funções de transferência, modificar o título e, no mesmo menu, eliminar o *workflow* virtual e eliminar estados previamente criados. Como o estado que dá início ao *workflow* é idêntico para todos os *workflows*, ao inserir o primeiro estado, este deverá ligar automaticamente ao estado inicial. Como pode ser visto no diagrama, só fará sentido a inserção de uma função de transferência depois de inseridos dois estados, assim como só fará sentido modificar um estado ou função de transferência depois de esta ser criada. Para tal, o acesso a estes menus será limitado por estas restrições. A qualquer momento o utilizador poderá sair da interface e dar o *worfklow* como concluído.

3.3.3 - Modelo de dados

Para o desenvolvimento do módulo *workflow* virtual será necessário adicionar à base de dados do *iPortalDoc-Light* um conjunto de tabelas que irão permitir armazenar os dados

correspondentes a este módulo. Na figura 3.7 pode ser visto o modelo de dados das tabelas criadas.

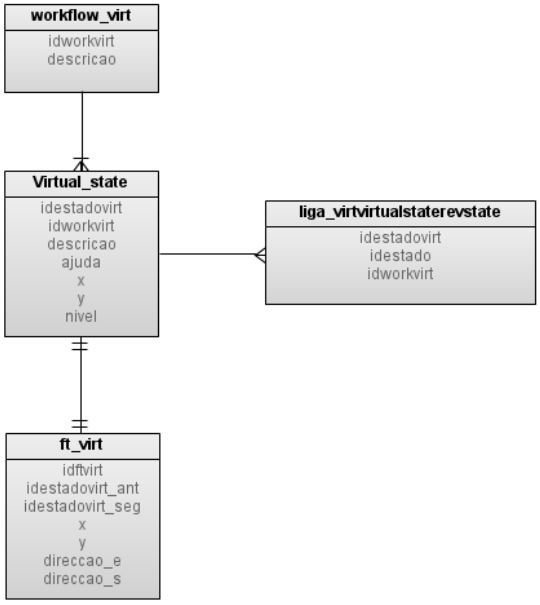

Figura 3.8 - Modelo de dados do módulo workflow virtual

Foram criadas quatro tabelas. Na tabela "workflow_virt" são alojados os dados do *workflow* virtual, uma chave primária ("idworkvirt") que o identifique e a correspondente descrição ("descricao"). De seguida será necessário criar tabelas que alojem os dados dos estados e as funções de transição que serão inseridas. Nas tabelas "Virtual_state" e "ft_virt" serão guardadas, respectivamente, os dados correspondentes aos estados e funções de transferência. A primeira compreenderá uma chave primária ("idestadovirt") uma chave estrangeira ("idworkvirt") da tabela "workflow_virt" uma descrição ("descricao"), ajuda ("ajuda"), as coordenadas x ("x") e y ("y") e o nível ("nível") do estado. A segunda terá uma chave primaria ("idftvirt") os identificadores dos estados anterior ("idestadovirt_ant") e seguinte ("idestadovit_seg") as coordenadas x ("x") e y ("y") da posição da função de transferência e as direcções de saída do estado ("direccao_e") e de saída da função de transferência ("direccao_s"). A tabela que efectua a ligação entre os estados do *workflow* original que estarão associados a um estado virtual chamar-se-á "liga_virtvirtualstaterevstate", esta conterá duas chaves estrangeiras, uma correspondente ao identificador do *workflow virtual* em causa ("idworkvirt") e outra para o identificador do estado virtual ("idestadovirt"), contém ainda um identificador para cada estado do *workflow* original que está associado ao estado virtual ("idestado"). Este identificador corresponde à chave primária da tabela estado, contida na base de dados do *iPortalDoc*.

3.3.4 - Solução preliminar

Este módulo deverá estar dividido em duas partes. A primeira consistirá na interface onde um administrador poderá criar e modificar o *workflow* virtual e que ficará alojada no menu de configuração do *iPortalDoc*. A segunda será a interface onde um utilizador poderá visualizar o *workflow* virtual. Esta será apresentada no menu de informação de documento do *iPortaDoc-Light*.

A interface de configuração deverá apresentar o *worklfow* original ao qual se pretende associar um *workflow* virtual e uma interface onde será efectuada a concepção do *workflow* virtual. No menu de configuração, deverão ser apresentados menus que permita adicionar e modificar estado, funções de transferência e dados do *workflow* virtual. Para cada uma das funções referidas será apresentada uma interface onde o utilizador possa adicionar e modificar os dados de cada elemento. Deverá também ser possível criar estado e funções de transferência com recurso à funcionalidade *drag and drop*, de modo a facilitar a inserção destes componentes.

3.4- Módulo de inserção de templates

Neste ponto será feita a descrição da concepção do módulo menu de templates. Este deverá permitir a um administrador, no *iPortalDoc*, especificar um tipo de documento que seja baseado em mais do que um template e, no *iPortalDoc-Light*, possibilitar criar esse documento mostrando num menu baseado em *tabs* os templates a inserir.

De seguida será focada arquitectura do modelo de dados de sistema, requisitos funcionais do módulo e etapas da implementação deste módulo.

3.4.1 - Requisitos

De seguida são apresentados os requisitos funcionais de cada um dos intervenientes no funcionamento deste módulo.

Administrador (iPortalDoc):

- Deverá ter a possibilidade de criar um menu de inserção de *templates*, definindo o nome do documento, o nome a apresentar em cada *tab* e respectiva macro a preencher;
- Deverá ser possível adicionar e remover tantos menus quantos desejar;
- Deverá ter a possibilidade de ver uma listagem de todos os documentos criados por si;
- Depois de criado um documento deverá ser dada a possibilidade de modificar ou remover o mesmo.

Utilizador (*iPortalDoc-Light*):

- No *iPortalDoc-Light*, caso existam documento criados, deverá ter acesso a um menu baseado em *tabs* onde em cada *tab* poderá inserir um documento;
- Depois de inseridos os templates, serão criados os documentos respectivos, sendo que estes estarão associados entre si.

3.4.2 - Casos de uso

Tal como no módulo anterior os conteúdos a que cada utilizador tem acesso são limitados pelo nível de permissão que é atribuído. Só um utilizador com permissões de administrador

pode ter acesso a este menu. Como se pode ver na figura 3.8, os casos de uso deste módulo são:

Administrador:

- Criar documento;
- Atribuir nome ao documento;
- Adicionar/remover menus;
- Atribuir macro ao menu.

Utilizador:

- Visualizar documento no *iPortalDoc*.

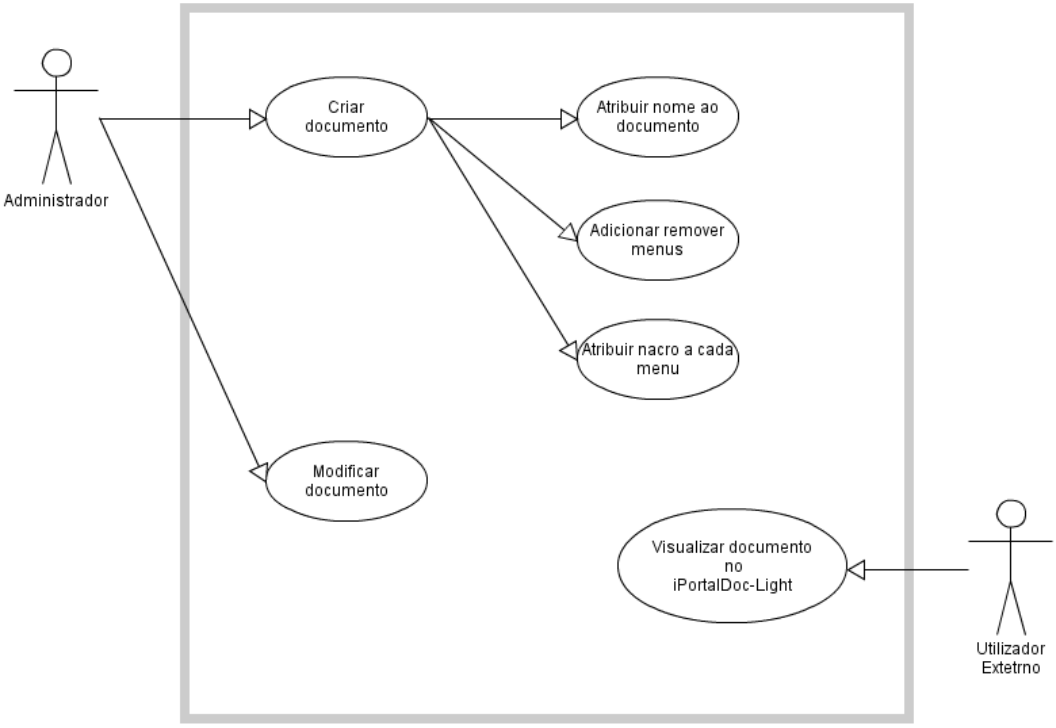

Figura 3.9 - Casos de uso do módulo templates

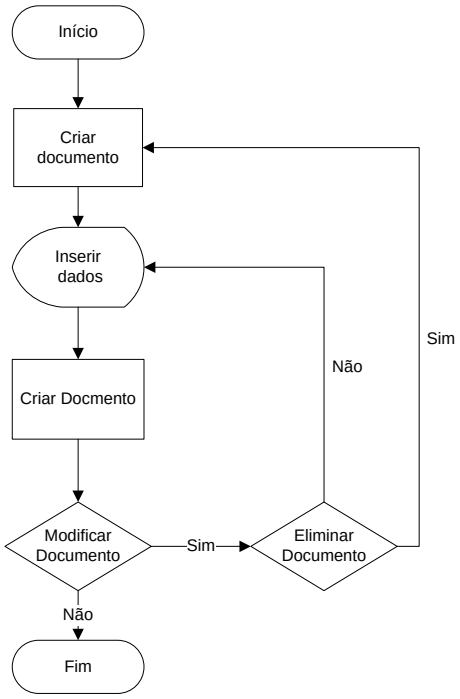

Figura 3.10- Diagrama de fluxo do menu de templates

Na figura 3.9 é possível ver um diagrama de fluxo que mostra a relação das funcionalidades que o módulo a desenvolver deverá suportar.

Um utilizador poderá criar um documento inserindo na interface principal deste módulo o nome do documento e definindo um nome e respectivo template a cada *tab* que pretenda criar. Depois de inseridos esses dados, o documento será criado e posteriormente o utilizador terá a possibilidade de modificar o documento.

3.4.3 - Modelo de dados

Tal como no módulo anterior, neste também foi necessário criar um conjunto de tabelas na base de dados do *iPortalDoc-Light*. Na figura 3.10 podem ser vistas as tabelas que será necessário adicionar à base de dados.

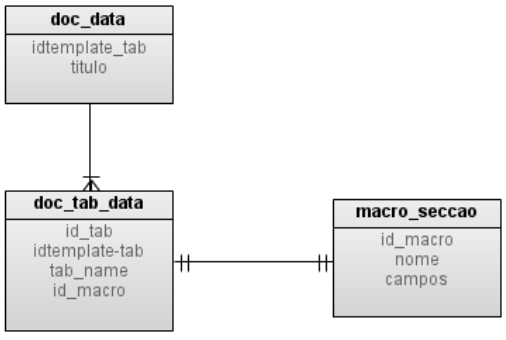

Figura 3.11 - Modelo de dados do módulo templates

Para alojar os dados relativos a este módulo será necessário criar três tabelas na base de dados do *iPortalDoc-Light*. Na tabela "doc_data" ficarão alojados os dados que identificam o documento a ser criado, estes consistem num identificador (idtemplate_tab) e no título do

documento (titulo). Na tabela seguinte, “doc_tab_data”, serão guardados os dados relativos a cada uma das *tabs* a ser criadas, contendo um identificador (id_tab) como chave primária, uma chave estrangeira (idtemplate_tab), o nome a atribuir a cada tab (tab_name) e o identificador da macro (id_macro) correspondente à macro que será inserida em cada uma das *tabs*. A tabela “macro_seccao” faz parte da base de dados do *iPortalDoc* e contém a informação relativa às macros que são criadas no *iPortalDoc*, é constituída por um identificado (id_macro), pelo nome (nome) e pelos campos que cada macro terá (campos). Os dados desta tabela serão necessário para especificar as macros que o utilizador quererá usar.

3.4.4 - Solução preliminar

Tal como o módulo de *workflow* virtual, este também será dividido em duas interfaces. Uma, situada no *iPortalDoc*, permitirá a um administrador criar os documentos a outra, situada no *iPortalDoc-Light* disponibilizará a interface de inserção do respectivo documento.

Na interface de concepção o administrador deverá ter a possibilidade atribuir um nome ao documento de escolher o número de *tabs* que pretende e o template estará associada a cada uma delas. Deverá também ser possível modificar um documento depois de o criar.

No *iPortalDoc-Light* o utilizador deverá visualizar o documento e em cada *tab* deverá ser mostrado o template que será necessário introduzir. A introdução de documentos deverá ser feita da mesma forma utilizada actualmente para inserir um template no *iPortalDoc-Light*, como pode ser visto no ponto 3.1.2 do presente capítulo.

3.5- Módulo Assinatura digital

Neste ponto será descrita a concepção do módulo de assinatura digital. Este deverá possibilitar a um utilizador, no *iPortlaDoc-Light*, assinar digitalmente um documento. Para tal será necessário que o *workflow* associado a um documento contenha a acção “Assinar o documento com a assinatura electrónica do cartão do cidadão”.

A concepção deste módulo não foi completada, tendo sido o seu desenvolvimento considerado trabalho futuro.

3.5.1 - Requisitos

Neste ponto serão descritos os requisitos funcionais que este módulo deverá cumprir

- Quando o utilizador tiver atribuída a si a acção “Assinar o documento com a assinatura electrónica do cartão do cidadão”, esta deverá aparecer na listagem de acções da interface principal do *iPortalDoc-Light*;
- Permitir inserir uma assinatura digital no documento;
- Permitir escolher a localização no ficheiro onde se pretende inserir a assinatura digital;

- Adicionar comentários à acção;
- Reencaminhar o documento para outros utilizadores.

3.5.2 - Casos de uso

Tal como nos dois módulos anteriores as permissões atribuídas a um utilizador que aceda ao *iPortalDoc-Light* são definidas pelo conteúdo a que este tem acesso. Como tal, o utilizador só terá acesso a acções de documentos que tenha permissão de visualizar.

Deste modo, os casos de uso resumem-se a:

- Visualizar acções;
- Inserir comentário na acção;
- Seleccionar localização da assinatura;
- Assinar digitalmente documento;
- Realizar acção.

De seguida será apresentado um diagrama funcional do módulo assinatura digital.

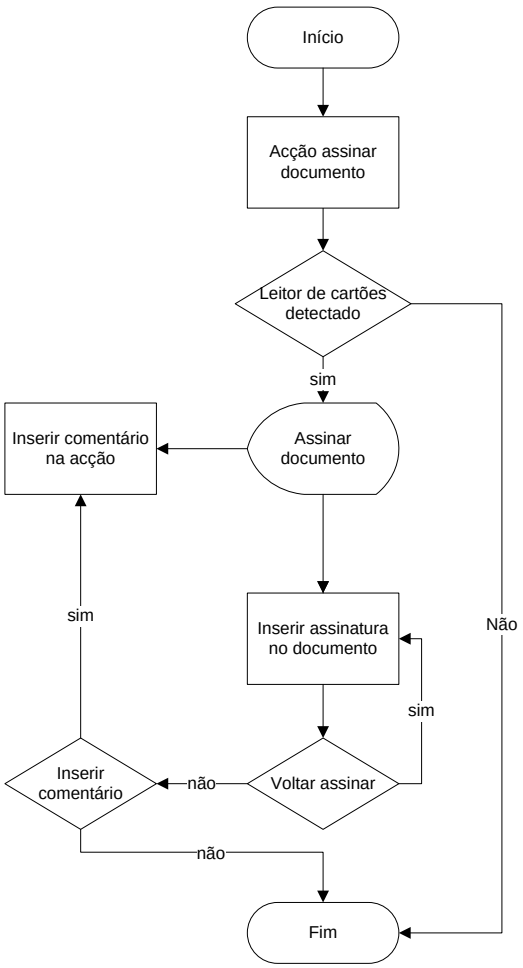

Figura 3.12 - Diagrama de fluxo da acção assinar digitalmente documento

Na figura 3.11 é possível ver o diagrama de fluxo que deverá ser cumprido pelo módulo. Inicialmente, e para permitir o acesso do utilizador à interface onde poderá realizar a acção, será necessário detectar se existe algum leitor de cartões instalado. Caso o leitor de cartões seja detectado será permitido o acesso do utilizador à interface que permite introduzir uma assinatura digital no documento. Nessa interface o utilizador poderá inserir um comentário à acção que vai realizar, e escolher o local no documento onde pretende inserir a assinatura digital. Depois de concluída a acção esta será submetida.

3.5.3 - Modelo de dados

Para o desenvolvimento deste módulo não será necessária a inserção de novas tabelas na base de dados do *iPortalDoc*, pois esta funcionalidade basear-se-á apenas na realização de uma acção.

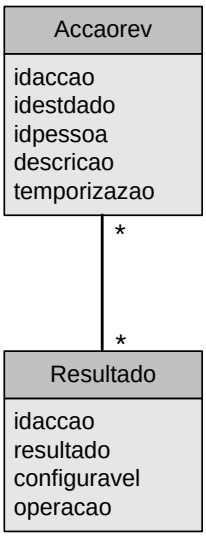

Figura 3.13 - Modelo de dados

Na figura 3.12 podemos ver as tabelas responsáveis pelo armazenamento dos dados relativos à realização de uma acção. Cada acção contém uma serie de atributos que a identifica, atributos estes que são guardados na tabela "Accaorev". Depois de realizada a acção a tabela "Accaorev" será actualizada e na tabela "Resultado" serão guardados os dados indicando que a acção foi completada.

3.5.4 - Solução preliminar

As funcionalidades que este módulo fornecerá estarão relacionadas com o menu de acções que é apresentado ao utilizador do *iPortalDoc-Light* que este faz login na aplicação.

Se a acção "Assinar o documento com a assinatura electrónica do cartão do cidadão" estiver associada ao *workflow* de um documento que o utilizador tenha permissão de ver, deverá ser apresentada a este a acção de assinar digitalmente o documento.

A interface onde o utilizador poderá assinar digitalmente o documento deverá fornecer ao utilizado uma visão geral do documento que irá assinar, um campo onde o utilizador possa

inserir um comentário à acção que está a realizar e um menu onde o utilizador escolha quais os destinatários do documento depois de completa a acção.

Devido à escassez de tempo para a realização por completo deste projecto e à importância no desenvolvimento de cada módulo no âmbito da empresa, o módulo de assinatura digital não foi terminado

Capítulo 4

Implementação

Como foi explanado no capítulo 2, depois de efectuada uma pesquisa e comparação entre várias as linguagens de programação que poderiam ser usadas para o desenvolvimento deste projecto, a escolha recaiu sobre o HTML, PHP, *JavaScript* e *frameworks YIU* e *Prototype*. O sistema de base de dados utilizado foi o *PostgreSQL*. Como foi referido, a escolha do PHP para linguagem principal de desenvolvimento e do *PostgresSQL* para sistema de base de dados, prendeu-se maioritariamente com o facto de a empresa já os utilizar. Quanto ao uso das *frameworks*, o factor referido anteriormente foi também preponderante, no entanto depois de ter sido feita uma comparação entre as principais *frameworks*, esta sobressaíram como escolhas acertadas.

Na secção inicial é feita um apanhado das plataformas de desenvolvimento utilizadas para a implementação deste projecto sendo nas secções seguintes descritas as várias etapas do desenvolvimento de cada um dos módulos.

4.1- Plataformas de desenvolvimento

Nesta secção será feita uma síntese das ferramentas utilizadas para o desenvolvimento de todo o trabalho realizado.

Dado que o desenvolvimento deste projecto foi realizado na empresa *iPortalMais*, todo o hardware necessário à implementação deste foi facilitado pela empresa.

Em relação ao software utilizado e visto a empresa *iPortalMais* funcionar, na sua maioria, com *software open source*, todo o desenvolvimento do projecto foi feito com recurso a este tipo de *software*:

- **Sistema operativo:** Ubuntu - sistema operativo (S.O.) baseado em Debian GNU/Linux. É um S.O. estável e actualizado constituído por vários pacotes de software que tal como o S.O. em si são na sua maioria gratuitos e *open source* [15];
- **Editor de desenvolvimento:** Quanta Plus - ambiente de desenvolvimento baseado no KDE, ou seja, integrado com o sistema operativo Linux para linguagens como HTML, PHP, XML, etc. [16];

- **Browsers:** Firefox – *Web browser* gratuito e *open source* desenvolvido pela coorporação Mozilla [17]. É o segundo *browser* mais utilizado no mundo e, de acordo com a W3C [18], o *browser* mais utilizado por programadores *Web*;
- **Ferramenta de acesso à base de dados:** PostgeSQL – Sistema de base de dados gratuito e *open source* baseado em linguagem de *query* SQL [19];
- **Ferramenta de gestão de versões (CVS):** Cervisia – Interface gráfico baseado no KDE para o sistema CVS (Concurrent Versions System) facilitando, deste modo, as funções de inserir, remover e submeter ficheiros presentes num sistema de CVS [20].

4.2- Módulo de pesquisa

Nos pontos seguintes serão detalhadas as etapas do desenvolvimento de cada componente do módulo. De modo a tornar mais perceptível os passos seguidos durante a implementação, cada funcionalidade desenvolvida será abordada num ponto diferente.

4.2.1 - Ícone de pesquisa avançada

Tal como é apresentado na interface principal do *iPortalDoc*, o menu de pesquisa avançada será invocado ao clicar num ícone com as letras PA (pesquisa avançada) situado no canto superior direito da interface principal do *iPortalDoc-Light*. Este ícone foi criado com recurso a CSS e ao clicar nesse ícone é chamada uma função *JavaScript* que irá abrir uma nova janela onde será apresentado o menu de pesquisa. Este menu será descrito no ponto seguinte.

4.2.2 - Formulário de pesquisa

O ponto inicial da implementação deste módulo consistiu no desenvolvimento do ficheiro PHP (“pesquisa_avancada.php”), ficheiro este que contém o código desta funcionalidade.

Os campos que o utilizador poderá usar para efectuar uma pesquisa são: assunto, entidade, tipo de documento, *workflow*, estado do *workflow*, data de introdução, data de elaboração, código, localização, título, sumário, descrição e/ou palavra-chave. Cada um destes menus será um elemento criado no interior de um “form” HTML. Os campos assunto, entidade, tipo de documento, *workflow* e estado foram desenvolvidos com recurso ao elemento “select” de HTML e disponibilizam uma listagem dos respectivos conteúdos de cada campo. A data de introdução e data de elaboração são “checkboxes”, que ao serem seleccionadas fornecem, ao utilizador, a possibilidade de seleccionar uma data para o campo escolhido. Os campos código, localização física e pesquisar são campos de texto criados com recurso ao elemento “text”. O campo pesquisar permite distinguir sobre que característica a pesquisa irá incidir e pode ser do tipo título, sumário, descrição e/ou palavra-chave. Estes foram implementados com recurso a “checkboxes” permitindo desta forma escolher um ou mais que um tópico de pesquisa.

Ao submeter a pesquisa, é invocada uma função *JavaScript* `Form_Validator()` que irá validar a pesquisa e de seguida submete-la para o ficheiro onde irá ser realizada a acção

(pesquisa_chave.php). Este ficheiro irá invocar as *queries* necessárias consoante o tipo de pesquisa que o utilizador pretenda efectuar.

De modo a organizar as funções que realizam as *queries*, cada campo do menu de pesquisa chama uma função contida numa classe PHP associada a esse campo. Por exemplo, no fragmento de código mostrado abaixo, pode ser vista uma função, invocada quando o utilizar efectua uma pesquisa por assunto, e que ficará alojada no ficheiro "Assunto.phpclass".

```
function getAssuntoByIdassunto($idassunto)
{
…
$query = "select iddoc from assunto where idassunto='".$idassunto."'";
$resultado = $GLOBALS["libdatabase"]->queryDB($this->conn,$query);
…
$vect=($GLOBALS["libdatabase"]->getQueryResultsDB($resultado));
return $vect;
}
```

Depois de recebidos os dados relativos às *queries*, são mostrados os resultados correspondentes aos paramentos de pesquisa introduzidos.

Na interface onde são fornecidos os resultados da pesquisa foi criado um *link* que permite ao utilizador efectuar uma nova pesquisa. Esta nova pesquisa irá chamar a página "pesquisa_avancada.php" e serão passados por URL os parâmetros que foram introduzidos na pesquisa anterior, o que irá mostrar, nesta nova pesquisa, uma interface com esses campos já preenchidos.

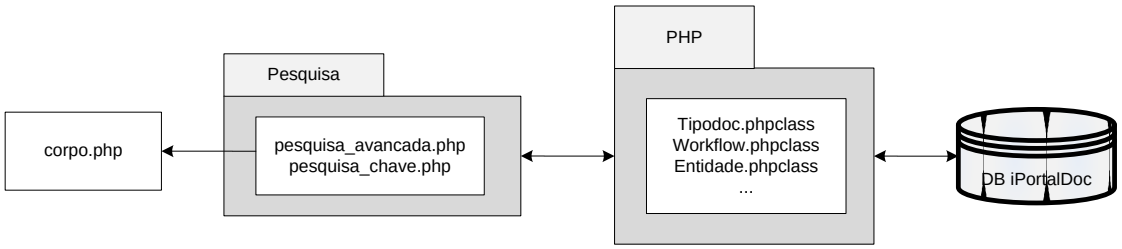

Figura 4.1 - Organização dos ficheiros do menu pesquisa

Como pode ser visto na figura 4.1, os ficheiros são organizados consoante o seu tipo ou função. O ficheiro "corpo.php" é responsável por fazer a inclusão de novos ficheiros quando estes são chamados. No directório "Pesquisa" serão alojados todos os ficheiros correspondentes às interfaces do menu de pesquisa. O directório onde são guardadas as classes PHP responsáveis pelas *queries* à base de dados do *iPortalDoc* tem o nome de PHP. Neste directório podemos ver as classes PHP que irão realizar as *queries* consoante a pesquisa que o utilizador pretenda realizar. Como se pode ver, o nome das classes é em tudo semelhante aos campos sobre os quais uma pesquisa pode incidir, facilitando assim a sua organização.

4.3- Módulo de *Workflow* virtual

Nas secções seguintes será detalhada a metodologia implementada no desenvolvimento de cada componente do módulo. De modo a tornar mais perceptível os passos seguidos durante a implementação, cada funcionalidade desenvolvida será abordada numa secção diferente.

4.3.1 - Escolha de *workflow*

O caso de uso a ser implementado inicialmente foi o que permite ao utilizador seleccionar o *workflow* a que pretende associar um *workflow* virtual. Para tal foi desenvolvida uma interface que será adicionada ao menu de configuração do *iPortalDoc-Light*.

Foi criado um ficheiro "lista_workflow.php" que ira alojar o código dessa interface. Nesse ficheiro foi criado um elemento "select" de HTML onde serão apresentados os *workflow* que se encontram actualmente associados. Cada opção do elemento "select" é preenchida com o resultado de uma *query* à tabela de base de dados que contém a listagem dos *workflows* associados. De modo a não replicar código, a *query* foi realizada com recurso a uma classe PHP ("Workflow.phpclass") que já desempenhava essa funcionalidade. Ao seleccionar uma opção será chamada uma função *JavaScript* ("`reloadworkflowed()`") que irá chamar uma nova janela, onde será apresentada a interface principal.

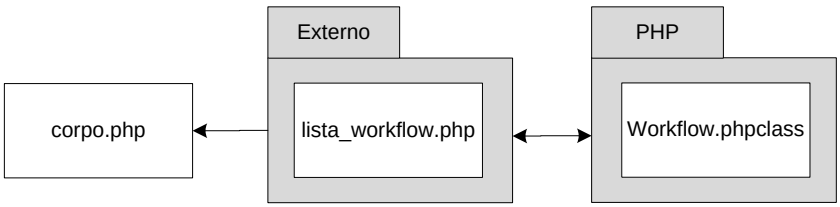

Figura 4.2 - Organização de ficheiros da interface de selecção de workflow

Na figura 4.2 pode ser vista a organização dos ficheiros correspondentes a esta interface. Quando a interface é chamada, o ficheiro "corpo.php" faz a inclusão do ficheiro "lista_workflow.php" que contém a interface correspondente a este caso de uso. O directório PHP contém o ficheiro "workflow.phpclass" onde estão inseridas as funções que realizam as *queries* pretendias.

4.3.2 - Interface principal de geração do *workflow*

De seguida, o problema consistiu na escolha da melhor forma de desenvolver a interface principal, onde será possível visualizar a menu de concepção do *workflow* virtual e, simultaneamente, permitir visualizar o *workflow* ao qual se pretende associar o *workflow* virtual.

Inicialmente foi criado o ficheiro "visualiza_workflow_light.php" que contém o código relativo a esta interface.

Como solução para este problema exposto acima, foi usada a componente *tabview* da *framework YUI*. O uso desta componente permitiu resolver o problema facilmente,

apresentando uma interface com duas *tabs* - onde na primeira seria apresentado o *workflow* original e na segunda o menu que permitisse criar o *workflow* virtual. Como se pode ser no fragmento de código apresentado abaixo o código necessário à implementação das *tabs* é relativamente intuitivo, sendo apenas necessária a inserção dos ficheiros CSS e dependências do componente *tabview*.

```
<div id="demo" class="yui-navset">
   <ul class="yui-nav">
      <li class="selected"><a href="#tab1"><em> Workflow iPortalDoc </em></a></li>
      <li><a href="#tab2"><em> Workflow Virtual </em></a></li>
   </ul>
   <div class="yui-content">
      <div><p>
            … //conteúdo da primeira tab – workflow original
     </p></div>

      <div><p>
            … //conteúdo da segunda tab – interface de concepção do workflow virtual
     </p></div>
   </div>
</div>
```

Esta interface necessitará de utilizar tanto a base de dados do *iPortalDoc*, como a do *iPortalDoc-Light*. Isso deve-se ao facto de ser mostrado o *workflow* original que se encontra inserido na base de dados do *iPortalDoc* e será nesta interface que será criado todo o *workflow* virtual que ficará inserido na base de dados *iPortalDoc-Light*. No ponto seguinte será demonstrado como foi implementado o desenho dos *workflows*.

De salientar que esta interface servirá de interface de concepção e de modificação, como tal foi necessário distinguir entre as duas, para tal foi criado um simples ciclo em PHP em que, caso o identificador do *workflow* virtual (`idworkvirt`) não exista mostra uma interface onde o utilizador poderá criar o *workflow* virtual, caso contrário mostra a interface de desenvolvimento.

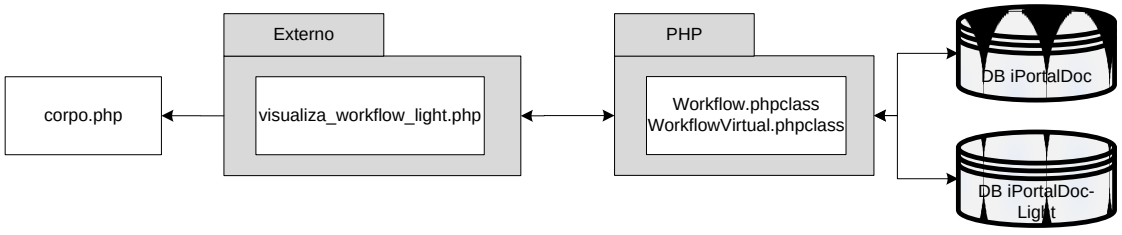

Figura 4.3 - Organização de ficheiros da interface principal de concepção

Na figura 4.3 pode ser vista a organização de ficheiros desta interface. Tal coma na interface anterior, quando a interface é chamada é feita a inclusão desta através do ficheiro corpo.php. As funções responsáveis pelas *queries* que esta interface necessitará fica incluídas

nos ficheiros workflow.phpclass e WorkflowVirtual.phpclass que efectua *queries* à base de dado do *iPortalDoc* e *iPortalDoc-Light* respectivamente.

4.3.3 - Desenho da grelha, estados e funções de transferência

O passo seguinte consistia na criação de uma grelha que serviria de base ao desenho do *workflow*.

Para tal foi criado um ficheiro chamado "diagrama_virtual.php" que permitirá gerar imagens dinamicamente usando um script CGI. Este tipo de scripts permite passar paramentos a este ficheiro e gerar, neste caso uma imagem. Para tal basta definir o tipo de conteúdo. Viste tratar-se de uma imagem, foi definido da seguinte forma - `Header("Content-type: image/jpeg");`.

Depois de se ter conhecimento sobre o tamanho da grelha pretendida e a suas posições na interface, esta foi criada com recurso à função "`imageline()`" para desenhar a grelha, bem como à função "`ImageChar()`" para desenhar, ao redor da grelha, as coordenadas. Esta imagem servirá de base ao menu *workflow*, que é apresentado para criar o *workflow* virtual e para o menu que apresenta o *workflow* já existente no *iPortalDoc*.

O desenho dos estados é conseguido com recurso função "`imagefilledellipse`", esta função desenha uma elipse centrada nas coordenadas passadas pelas variáveis $c_x e $c_y. Como se pode ver no fragmento de código apresentado abaixo, a função tem como parâmetros a imagem a inserir (`$imagem`) as coordenadas x e y (`$c_x` e `$c_y`) a largura e altura (`$largura` e `$altura`) e a cor (`$cor`).

```
…
imagefilledellipse($imagem, $c_x, $c_y, $largura, $altura, $cor)
…
```

As variáveis correspondentes às coordenadas são preenchidas com queries à base de dados efectuadas por funções contidas na classe PHP "VirtualState.phpclass". Como se pode ver na figura 3.5, a tabela "virtualstate" contém os campos x e y que correspondem às coordenadas onde o estado foi criado. As outras variáveis são predefinidas dentro do próprio ficheiro.

Visto as funções de transferência serem apenas linhas, estas foram implementadas, tal como a grelha em si, com recurso à função PHP "imageLine()", no entanto as coordenadas necessárias são obtidas dá base de dados. Como se pode ver na figura 3.6, a tabela "virtua_ft" contém os campos x e y que corresponde às coordenadas da função de transferência.

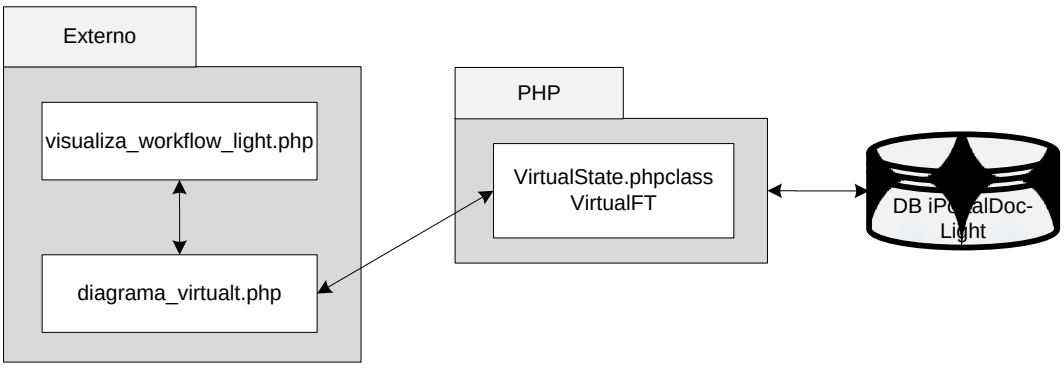

Figura 4.4 - Organização de ficheiros do desenho da grelha, estados e F.T.

Na figura 4.4 é possível ver a interligação e disposição dos ficheiros. Sempre que a imagem gerada pelo "diagrama_virtual.php" for chamada pelo ficheiro "visualiza_workflow_light.php" são realizadas *queries* à base de dados do *iPortalDoc-Light*, permitindo desta forma gerar uma imagem com os campos devidamente preenchidos.
A implementação da imagem onde é mostrado o *workflow* original é implementada de forma semelha, sendo que nesse caso as *queries* são realizadas à base de dados do *iPortalDoc*.

4.3.4 - Menus de configuração

O ponto seguinte consistiu no desenvolvimento dos menus onde o utilizador poderá inserir e editar os dados de cada um dos componentes do *workflow*. De modo a tornar a interface mais funcional e interactiva, o recurso ao *JavaScript* torna-se, mais uma vez, essencial. Para tal, ao invés de serem usadas novas janelas para os diferentes menus, o que levaria a um decréscimo da performance do *browser* e da aplicação em si, foi usado a componente *panel* da família *container* da *framework YUI*. Este imita o funcionamento de uma janela, consistindo num conteúdo modular que pode ser posicionado dentro dos limites de um campo e por cima da camada principal da página sem afectar o funcionamento desta. Com recurso a este componente foram implementados painéis para: adicionar estados; modificar estados; remover estados; adicionar *Workflow*; adicionar f.t.; modificar f.t.; configurar *workflow*. Os elementos que o utilizador terá de editar em cada painel, foram implementados com recurso à ferramenta "form" em HTML. É de salientar que dada a semelhança entre os painéis "Adicionar estado" e "Modificar estado" (como se pode ver na figura 4.5), estes, ao invés de serem implementados separadamente, partilham o mesmo código, ou seja, foram concebidos com recurso ao código necessário para criar um só painel.

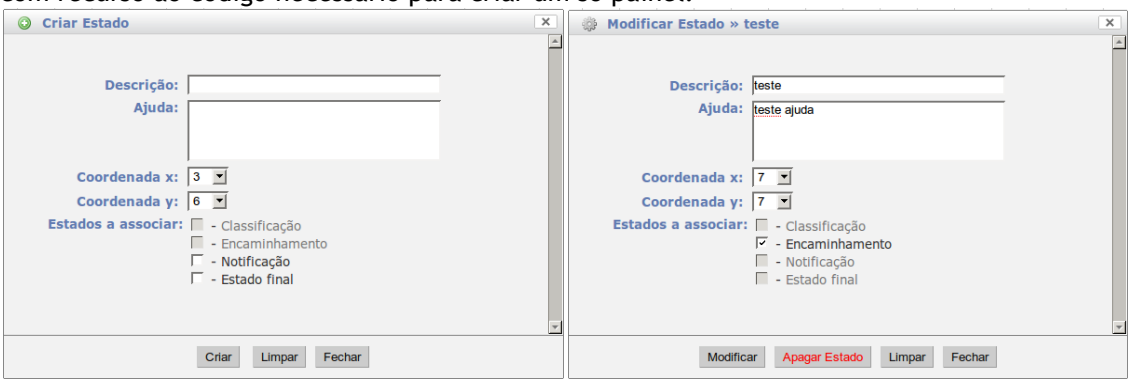

Figura 4.5 - Painéis de criar estado e modificar estado

Isto foi conseguido através do uso de DOM e da manipulação de elementos que o este permite obter. Como se pode ver no exemplo de código mostrado abaixo, o título do painel e o nome do botão são modificados com recurso à propriedade "innerHTML".

```
function showpanel_add_state(non_assoc, head, header, help, id, st_x, st_y, tipe)
{
…
ar sas_head =document.getElementById('state_header');
```

```
var sas_but_ins = document.getElementById('cvs_but_inserir');
…
    if (tipe == "0")
    {
sas_but_ins.value="Criar";
sas_head.innerHTML = "<b>" + “Criar Estado" + "</b>";
…
    else if (tipe == "1")
    {
            sas_but_ins.value="Modificar";
sas_head.innerHTML ="<b>"+"Modificar Estado"+"»"+header+"</b>";
…
}
```

O acesso aos menus de modificação de estados e funções de transferência foi implementado de forma a um menu aparecer quando se clica num estado ou função de transferência previamente criado. Para tal, foi criado um mapa da imagem da grelha, usando a *tag* HTML <map>, em que são efectuadas *queries* de modo a obter a posição dos estado e funções de transferência, inserindo nessa posição um *link* para o respectivo menu de modificação.

Na eventualidade de um campo de preenchimento obrigatório ser deixado em branco, será visualizado um *popup* de alerta que obrigará ao preenchimento dos campos para ser possível a continuidade da acção. É também apresentado um *popup* de confirmação quando se pretende eliminar algum dos componentes. Existem 3 tipo de caixas de *popup*, *alert* (usadas quando se pretende fazer chegar alguma informação ao utilizador [21]), *confirm* (quando se pretende que o utilizador verifique ou aceite algo [21]) e *prompt* (quando se pretende que o utilizador insira um valor [21]). Nesta implementação serão usadas caixas *popup* de *alert* e *confirm*.

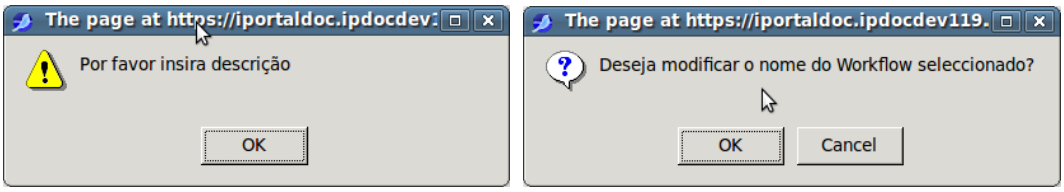

Figura 4.6 - Popup de alerta e confirmação

Na figura 4.6 podem ser visto dois *popups*, um de alerta e outro de confirmação, estes dois menus de confirmação permitem evitar erros na inserção de dados, quando os dados são submetidos, o que poderá evitar grandes perdas de informação.

4.3.5 - Interface iPortalDoc-Light

No *iPortalDoc-Light* foi adicionada à interface que permite ao utilizador visualizar as informações um campo que lhe permitirá visualizar o *worklow* virtual correspondente ao documento em questão.

O problema inicial consistiu em desenvolver a interface onde iria ser apresentado o *workflow* virtual, para tal foi criada uma imagem com o formato de uma grelha onde são

apresentados os estados e funções de transferência, de seguida foi criado, com o auxílio do componente *panel* da *Framework YUI*, um painel onde é mostrada a ajuda e descrição do estado. O acesso a este painel é feito clicando no estado correspondente, para tal foi feito um mapa da imagem atribuindo a cada estado da mesma um *link*. O procedimento de desenvolvimento dos componentes desta interface é em tudo semelhante ao descrito anteriormente para a elaboração do menu de criação de *workflow* virtual, como tal será omissa a descrição de todos os passos utilizados no seu desenvolvimento.

O *workflow* é visualizado clicando no ícone "*workflow*" que é apresentado na interface de informação do documento. Este link foi modificado de modo a apresentar a opção "*workflow* virtual", caso uma exista um *workflow* virtual, caso contrário será apresentada a opção original de mostrar o workflow "normal".

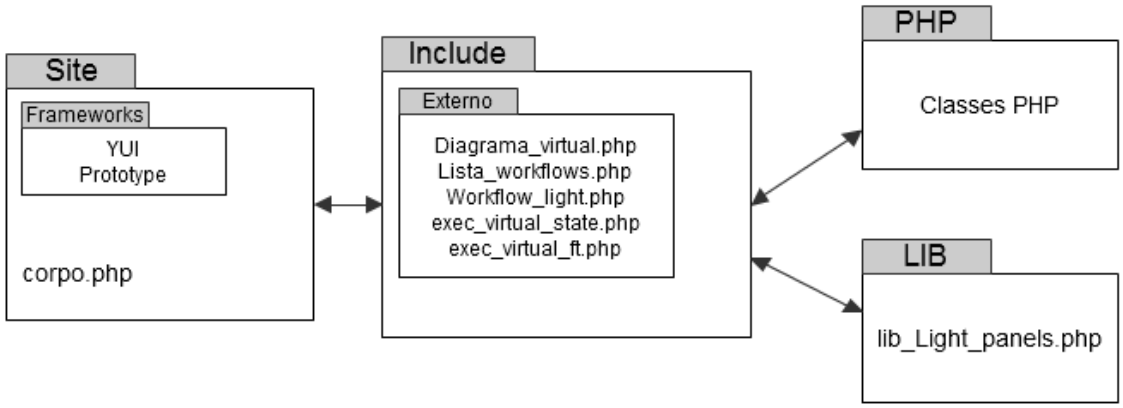

Figura 4.7 - Organização dos ficheiros do menu *workflow*

Na figura 4.7 pode ser vista a disposição dos directórios e ficheiros essenciais ao funcionamento deste módulo. O directório "Site" contém o ficheiro "corpo.php" que faz a gestão da inclusão de ficheiros, e também o directório *frameworks*, onde estão inseridos os ficheiros *JavaScript* e CSS dos vários componentes das *frameworks*. Tal como no módulo anterior, os ficheiros relativos às interfaces são armazenados no directório "Include" sendo que os correspondentes às interfaces deste módulo ficarão inseridos no directório "Externo". O ficheiro "Diagrama_virtual.php" contém o código necessário para o desenho da grelha, o "Lista_workflows.php" contém o código da interface que será apresentada no menu de configuração do *iPortalDoc-Light* e dará uma listagem dos *workflows* do *iPortalDoc*, o "Workflow_Light.php" contém a interface principal onde é possível criar o *workflow* virtual. Os ficheiros "exec_virtual_state" e "exec_virtual_ft.php" são os ficheiros onde são tratadas as acções provenientes dos formulários. As funções que invocam os painéis são guardadas num ficheiro chamado "lib_light_panels.php", ficheiro este que será armazenado directório à parte do código principal, chamado "LIB". Tal como no módulo anterior, as classes PHP que realizam as *queries* à base de dados são guardadas na pasta "PHP".

4.4- Módulo de inserção de templates

De seguida serão apresentados, divididas por secções, as etapas correspondentes à implementação deste módulo.

4.4.1 - Interface inicial

Inicialmente foi criado o ficheiro "lista_templates.php" onde ficaria inserido o código correspondente a esta interface.

O passo seguinte consistiu em criar, na interface de configuração do *iPortalDoc-Light*, um menu onde o administrador tenha acesso á interface onde poderá criar e modificar documento baseados em macros que será apresentado no *iPortalDoc-Light*. Para tal foi empregada a mesma solução utilizada na interface do módulo *workflow* e foram criadas duas *tabs*, com recurso ao componente *tabview* da *framework YUI*. Na primeira *tab* será apresentado o menu onde o utilizador pode criar o documento e na segunda o meu onde o utilizador poderá modificar um documento previamente criado.

4.4.2 - Interface de concepção de documento

Esta interface corresponde à apresentada na primeira *tab* em que será implementado o caso de uso criar documento.

O problema inicial consistia em criar um menu em que o administrador possa adicionar e remover menus de uma forma dinâmica. Embora o problema pudesse ser resolvido com recurso a PHP, depois de alguma pesquisa, foi escolhido o *JavaScript* para o solucionar.

Foi criada uma função que permite de modo dinâmico adicionar e remover elementos de uma tabela. Como pode ser visto no exemplo de código mostrado a baixo que exemplifica a criação de um botão que permite remover uma linha, são adicionadas linhas à tabela com o identificador `table_cr`, usando para tal os métodos DOM `insertRow()` e `appendChild()`. Os atributos ("type", "value", etc.) foram conferidos a cada elemento criado, através da função DOM, `setAttribute()`.

```
…
var tbl = document.getElementById('table_cr');
var lastRow = tbl.rows.length;
var row = tbl.insertRow(lastRow);
…
var cellRightBut = row.insertCell(3);
var but = document.createElement('input');
but.setAttribute ('type','button');
but.setAttribute ('value','Remover Menu');
but.setAttribute ('class','dados_red');
but.setAttribute ('className','dados_red'); // para o IE
but.onclick = function () {deleteRow(this,doo);};
cellRightBut.align = 'center';
cellRightBut.appendChild(but);
…
```

As macros de templates que o utilizador poderá inserir são apresentadas com recurso a um elemento "select" de HTML. Cada opção é preenchida com o nome e identificador de cada macro, que foram obtidos através de uma *query* à base de dados do *iPortalDoc*, à tabela "MacroSeccoes" onde estão guardados os dados das macros. O elemento referido anteriormente bem como restantes elementos da interface criar documento encontram inseridos num "form" HTML.

De modo a validar os dados inseridos, foi criada a função "validateForm()" que verifica se todos os campos estão preenchidos correctamente. Caso isso não se verifique é mostrado um menu de "alert" avisando o utilizador desse facto. Depois de validado, os dados do formulário são submetidos para o ficheiro PHP "exec_templates_ligth.php" que irá desempenhar as acções pretendidas. Neste ficheiro os dados são inseridos nas tabelas "doc_data" e "doc_tab_data" (como pode ser visto na figura 3.8 do capítulo anterior), tabelas estas que guardam os dados correspondentes ao documento criado.

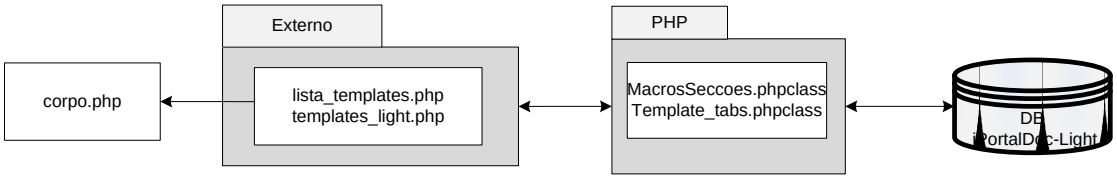

Figura 4.8 - Organização de ficheiros da interface de concepção

Na figura 4.8 pode ser vista a organização dos ficheiros utilizados para o desenvolvimento desta interface. Quando a interface inicial é camada, é feito a inclusão do ficheiro "lista_templates.php" com o auxílio do ficheiro "corpo.php". Na directoria PHP podemos ver a classe PHP "MacrosSeccoes.phpclass" que contém a função que realiza a *query* à tabela "MacroSeccao" de modo a obter uma listagem das macros disponíveis para inserção e a Template_tabs.phpclass responsável pelas funções que permitem a administração do módulo de templates.

4.4.3 - Interface de modificação de documento

Neste ponto é descrita a metodologia utilizada na interface que permite a modificação de documentos.

O código relativo a esta funcionalidade estará contido no ficheiro "lista_templates.php", este menu será apresentado na segunda *tab* da interface principal.

Para efectuar a selecção do documento que se pretende modificar foi criado um simples menu *drop down*, recorrendo a um elemento "select" de HTML onde cada opção corresponde a um documento.

De modo a tornar este menu mais interactivo, foi usado AJAX para chamar o menu de configuração escolhido no menu *drop down*. O uso de AJAX ao invés de PHP permitiu desenvolver uma interface mais rápida e funcional, não tendo a necessidade de actualizar a página. De modo a evitar problemas de compatibilidade com os diferentes browsers foi usada uma Framework para implementar o pedido AJAX. No excerto de código apresentado abaixo pode ser vista a função que implementa o pedido AJAX.

Quando é seleccionado um documento é chamada a função `editDocTemp()` que implementa o pedido AJAX, as variáveis correspondestes ao pedido que é efectuado são passadas para a função "DrawEditDocTable()" que guarda numa variável o código HTML gerado (tabela que contém um formulário idêntico ao usado para criar o documento mas com os dados correspondentes ao documento escolhido) e de seguida retorna essa variável. Durante o processamento do pedido, é passado por "innerHTML" (método DOM), uma `waiting_message` para dentro da "div" identificada por `add_doc_menu`. Assim que é recebido a resposta do pedido AJAX, o conteúdo da "div" é actualizado com a resposta.

```
function editDocTemp(iddoc_temp)
{
…
document.getElementById("add_doc_menu").innerHTML = waiting_message;
new Ajax.Request("ajax/corpoajax.php", {
method: 'post', parameters: postsvars,
    onComplete: function(response)
        {
            var edit_table = response.responseText;
            document.getElementById("add_doc_menu").innerHTML = edit_table;
        }
    ,onFailure: errFunc } );
}
```

Depois do conteúdo da "div" ser actualizado o utilizador terá ao seu dispor uma tabela idêntica à fornecida pelo menu de concepção onde poderá efectuar as alterações pretendidas.

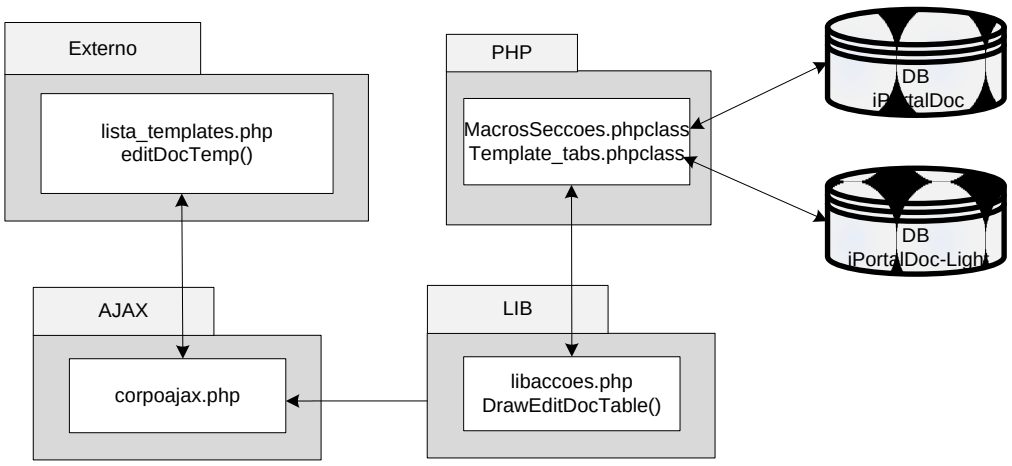

Figura 4.9 - Organização de ficheiros da interface de modificação

Na figura 4.9 pode ser vista a organização de pastas e ficheiros que foi seguida para o desenvolvimento desta funcionalidade. Como pode ser visto, quando a função `editDocTemp()` é chamada, esta inicializa o pedido AJAX. De seguida este comunica com o ficheiro "corpo.php" de forma a incluir o ficheiro "libaccoes.php" que contém a função `DrawEditDocTable()` onde é gerado a resposta ao pedido AJAX. A função

`DrawEditDocTable()`, de modo a preencher o código HTML que será gerado, necessita efectuar queries à base de dados do *iPortalDoc* e do *iPortalDoc-Light* Para tal usará os ficheiros "MacrosSeccoes.phpclass" e "Template_tabs.phpclass" contidos nas pasta PHP, que realizaram as *queries* necessárias.

4.4.4 - Interface iPortalDoc-Light

No *iPortalDoc-Light*, foi desenvolvido um menu onde é possível visualizar os vários documentos que se pretendem inserir, simultaneamente. Para tal recorreu-se mais uma vez ao uso da *Framework YUI* e do seu componente *tabview*. O uso deste componente permite que seja mostrado em cada *tab* do documento o template/macro que se pretende inserir.

A permuta entre *tabs* e consequente template/macro apresentar, foi realizada com recurso, mais uma vez, a AJAX. A implementação de comunicação AJAX, foi tudo semelhante à descrita no ponto anterior, tendo sido usado o pedido AJAX disponibilizado pela *Framework Prototype*, o que permite garantir a compatibilidade entre browsers.

Depois de preenchido o formulário este é apresentado dentro de uma "iframe" que se encontra no interior de cada *tab*.

A inserção do template e respectiva confirmação é feita recorrendo a um "HttpRequest", como pode ser visto no diagrama da figura 3.2 do ponto 3.1.2 do capítulo anterior.

No que toca à disposição dos ficheiros deste módulo no *iPortalDoc*, esta é idêntica à descrita nos pontos anteriores. Sendo que o ficheiro que contém a interface de criação e modificação de templates tem o nome de "templates_light.php".

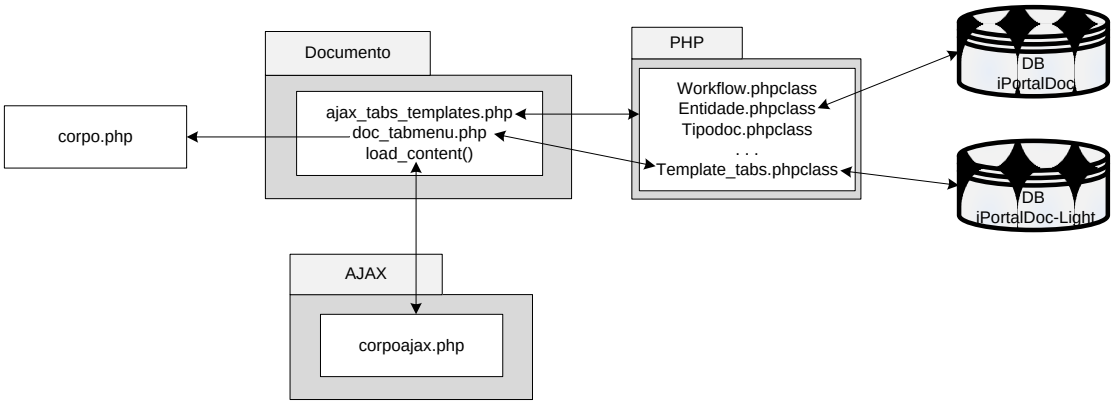

Figura 4.10 - Organização dos ficheiros

Na figura 4.10 é mostrada a organização dos ficheiros correspondentes à interface desenvolvida no *iPortalDoc-Light*.

Quando a interface é chamada, o ficheiro "corpo.php" faz a inclusão do ficheiro "doc_tabmenu.php" onde esta inserida a interface principal deste módulo. Este ficheiro irá gerar um determinado numero de *tabs*, consoante os dados inseridos pelo utilizador no menu de concepção definido no ponto 4.4.2. Sempre que há uma alternância entre *tabs* é chamada a função `load_content()` que realiza o pedido AJAX. O ficheiro "corpoajax.php" faz a inclusão do ficheiro "ajax_tabs_templates.php" que contém o código HTML que será a

resposta ao pedido AJAX. No ficheiro “ajax_tabs_templates.php”, consoante o identificador que lhe é passado, o ficheiro ira gerar o campo de edição da macro.

4.5- Conclusões

Depois de relatados os passos seguidos para o desenvolvimento dos diferentes módulos, verifica-se que todos os requisitos funcionais foram cumpridos e também que todos os casos de uso foram satisfeitos.

O uso de AJAX no desenvolvimento do módulo de inserção de templates contribuiu para desenvolver uma interface mais interactiva e funcional.

É de salientar que foram encontradas algumas dificuldades em assegurar a compatibilidade entre os diferentes *browsers*. A principal incompatibilidade verificou-se com browser Internet Explorer, e com os módulos de *workflow* virtual e inserção de templates, problema este que surgiu do elevado uso de *JavaScript* para o desenvolvimento destes, no entanto este problema foi ultrapassado com a utilização de *frameworks* no desenvolvimento.

Depois de testados os módulos em vários browsers (Internet Explorer, Firefox, Opera e Chrome) e, embora não tenham sido testados na totalidade de *browser* existentes, concluiu-se que os módulos desenvolvidos não apresentam problemas de compatibilidade, como era pretendido. Isso deve-se principalmente ao uso de CSS para definir a apresentação e *frameworks* para implementação da maioria do código *JavaScript* e AJAX.

Capítulo 5

Interface obtidas

Este capítulo tem por finalidade apresentar os resultados da implementação de cada módulo, ou seja as interfaces gráficas desenvolvidas e as funcionalidades que cada uma apresenta.

5.1- Menu de pesquisa

O primeiro módulo que foi sujeito a testes foi o módulo de pesquisa.

Depois de efectuar *login* no *iPortalDoc-Light* o utilizador terá acesso à interface principal, e no canto superior direito aos menus de pesquisa. Como se pode observar na figura 5.1 o utilizador poderá efectuar uma pesquisa rápida inserindo o conteúdo da pesquisa no campo devido e clicando no ícone "PR" dando início a uma pesquisa rápida pela palavra chave pretendida. Ao clicar no ícone "PA" terá acesso ao menu de pesquisa avançada.

Figura 5.1 - Icon de pesquisa avançada

A interface de pesquisa avançada é mostrada na figura 5.2. Neste menu, o utilizador terá acesso a um conjunto de campos que poderá preencher consoante a pesquisa que pretenda efectuar. De ter em conta que a pesquisa incide sobre os documentos do directório em que o utilizador se encontra e não sobre a totalidade dos documentos. Como se pode ver, os campos sobre os quais o utilizador pode pesquisar são, como foi mostrado no capítulo 3: assunto; entidade; tipo de documento; *workflow*; estado (do *workflow*); data de implementação; data de elaboração; código; localização física; pesquisa. Dentro deste ultimo campo, o utilizador

poderá escolher onde incidirá a pesquisa, se sobre títulos, sumários, descrições ou palavras-chave.

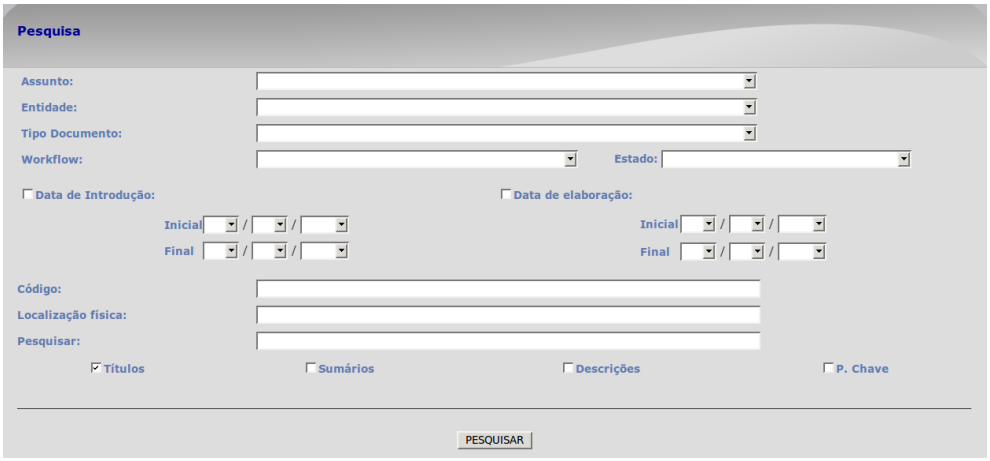

Figura 5.2 - Menu de pesquisa avançada

Na figura 5.3 podemos observar o resultado de uma pesquisa escolhendo no campo Tipo de documento, um documento do tipo Bilhete de identidade. Como se pode ver, são apresentados, não só os dados respectivos do documento, como também é dada a possibilidade de visualizar a informação e fazer download do documento.

Caso a pesquisa seja inconclusiva o utilizador poderá efectuar outra pesquisa sem sair da janela onde se encontra, para tal basta carregar no *link* ("NOVA PESQUISA") que se encontra no fundo da página.

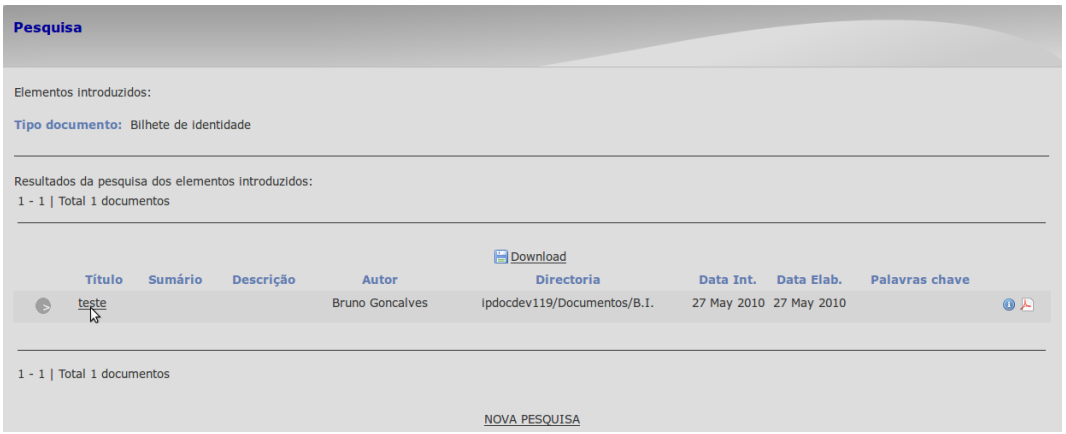

Figura 5.3 - Resultados de uma pesquisa

5.2- Workflow virtual

De seguida será apresentada as interfaces do módulo *workflow* virtual.

O módulo desenvolvido encontra-se inserido no menu que permite configurar o *iPortalDoc-Light*, este só estará disponível para um utilizador com permissões de *super user*.

No menu "Definições -> iPortalDoc-Light" o utilizador terá acesso a um conjunto de *links* que dão acesso a configurações do *iPortalDoc-Light*. O *link* com o nome "Workflows" dará acesso a uma interface onde o utilizador poderá escolher, de entre uma lista dos *workflows*

que se encontram já associados, o *workflow* ao qual pretende associar um *workflow* virtual. Esta interface pode ser vista na figura5.4.

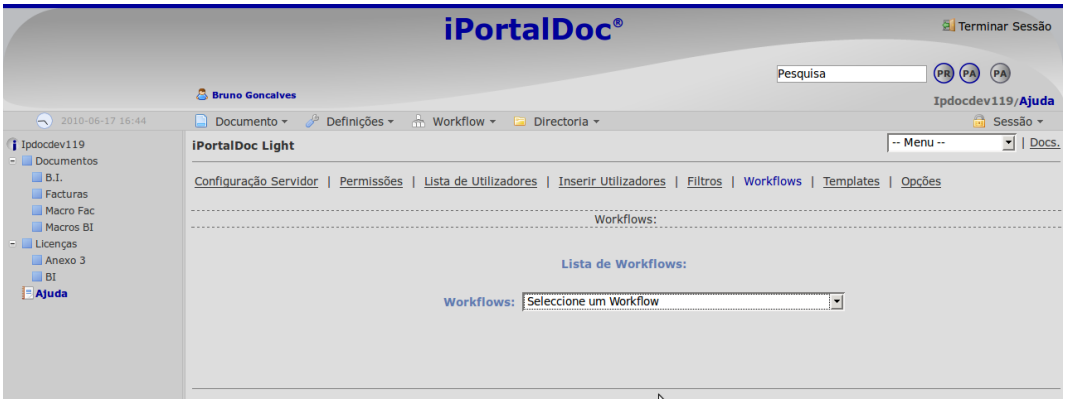

Figura 5.4 - Menu de configuração *iPortalDoc-Light*

Depois de escolhido um *workflow* da lista apresentada na figura 5.4, caso o *workflow* não tenha um *workflow* virtual já associado o utilizador terá acesso à interface da figura 5.5 onde poderá escolher criar ou não um *workflow* virtual. Caso queira criar um *workflow*, ao clicar no botão "Sim", ser-lhe-á apresentado um painel onde poderá inserir o nome do *workflow* virtual a criar. Depois de inserido o nome do *workflow*, ao clicar em "Criar" o utilizador será redireccionado para a interface mostrada na figura 5.7.

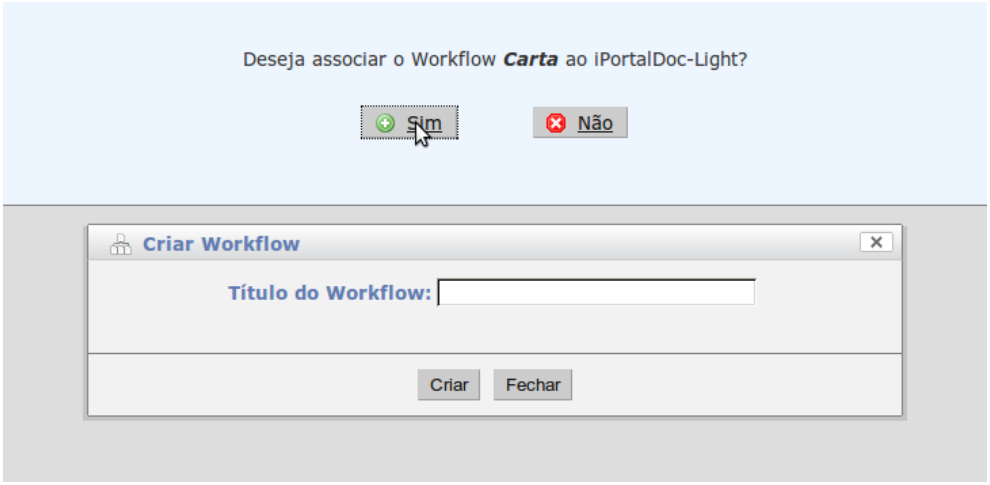

Figura 5.5 - Criar Workflow virtual

Caso, ao escolher um *workflow* do *drop down* menu da figura 5.4, este já tenha uma *workflow* virtual associado, o utilizador será reencaminhado directamente para a interface apresentada na figura 5.7, sendo que terá já ao seu dispor o *workflow* virtual previamente criado.

Na figura 5.6 é apresentada a interface onde o utilizador poderá visualizar, na primeira *tab*, o *workflow* original que escolheu. Isto permitirá ao utilizador criar o *workflow* virtual mais rapidamente e facilitará a associação de estados entre os *workflows*.

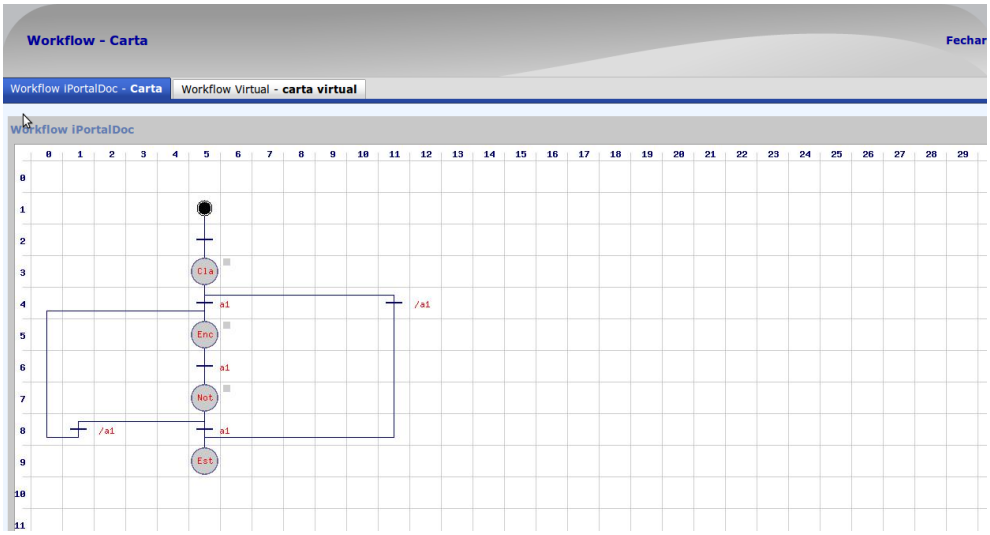

Figura 5.6 - Interface principal: *workflow iPortalDoc*

Na figura 5.7 pode ser vista a interface que permite ao utilizador desenvolver o *workflow* virtual.

Acima da grelha podem ser vistos os menus que permitem adicionar, configurar e eliminar elementos ao *workflow* virtual, do lado esquerdo são encontrados os dois elementos que podem ser arrastados para a grelha, sendo o de cima o estado e o de baixo a função de transferência.

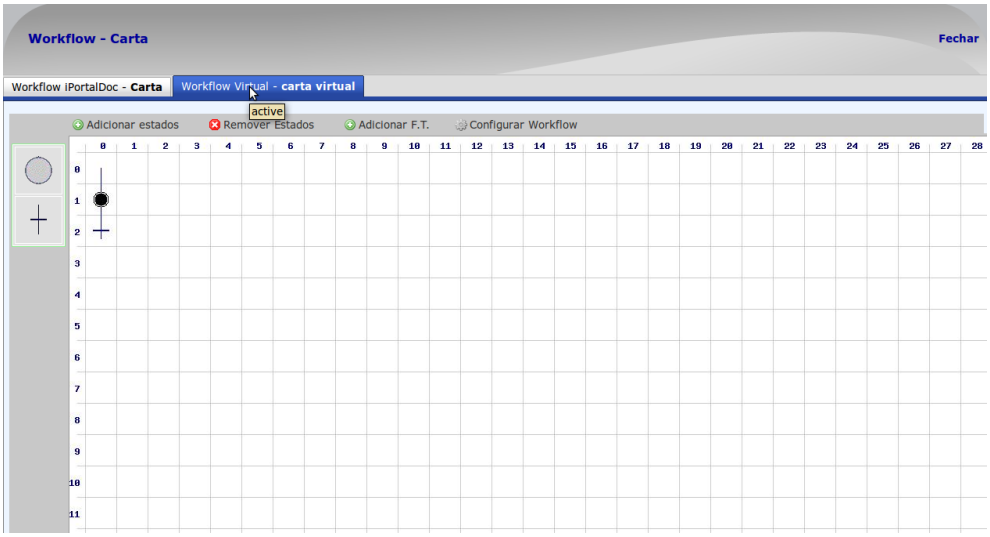

Figura 5.7 - Interface principal: menu de criação de Workflow virtual

Ao clicar no ícone “Adicionar estados” o utilizador terá acesso ao painel em que irá configurar a informação relativa ao estado, este painel pode ser visto na figura 5.8.Os campos a preencher são: descrição; ajuda; coordenadas x e y; estados a associar. O primeiro é o nome que será atribuído ao estado e tem de ser obrigatoriamente preenchido caso contrário não poderá prosseguir, o segundo campo, de preenchimento facultativo, é a ajuda que será mostrada quando o estado for visualizado no *iPortalDoc-Light*, de seguida serão necessárias as coordenadas x e y da posição do estado na grelha e por último será necessário escolher os estados do *workflow* original, que pretende associar ao estado que está a criar.

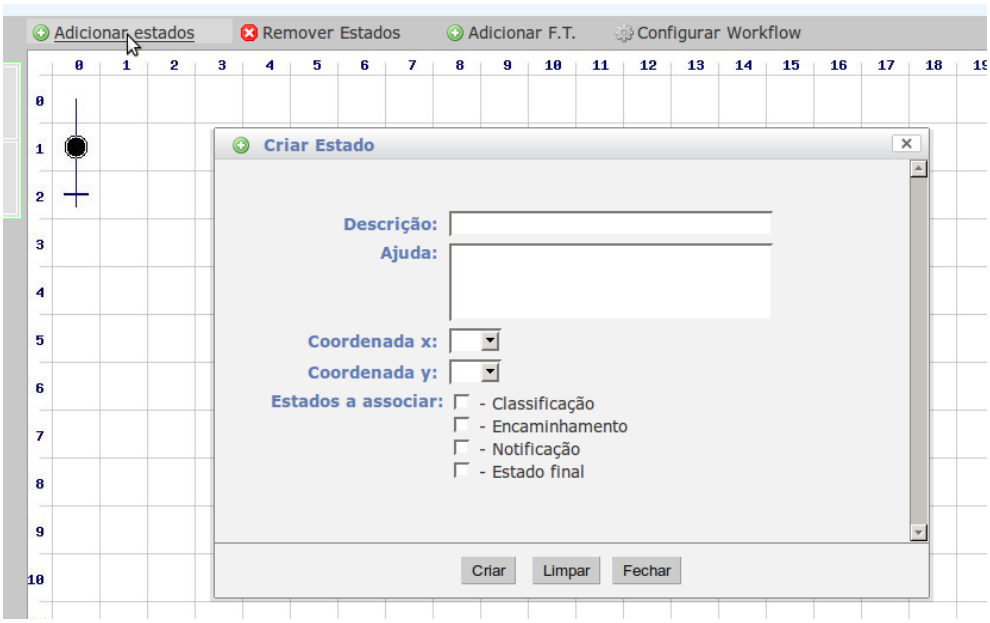

Figura 5.8 - Adicionar estado

Outra forma de inserir o estado é através da funcionalidade de *drag and drop*, para tal é apenas necessário arrastar o estado para a posição onde pretende inserir o estado. Ao fazer o "*Drop*" será mostrado o painel de criar estado em que as coordenadas x e y estão já preenchidas com a posição. De salientar que os estados que tiverem sido previamente associados estarão desactivados, evitando assim conflitos na inserção na base de dados e posteriormente na apresentação do ciclo de vida do *workflow*. Isto pode ser visto na figura 5.9.

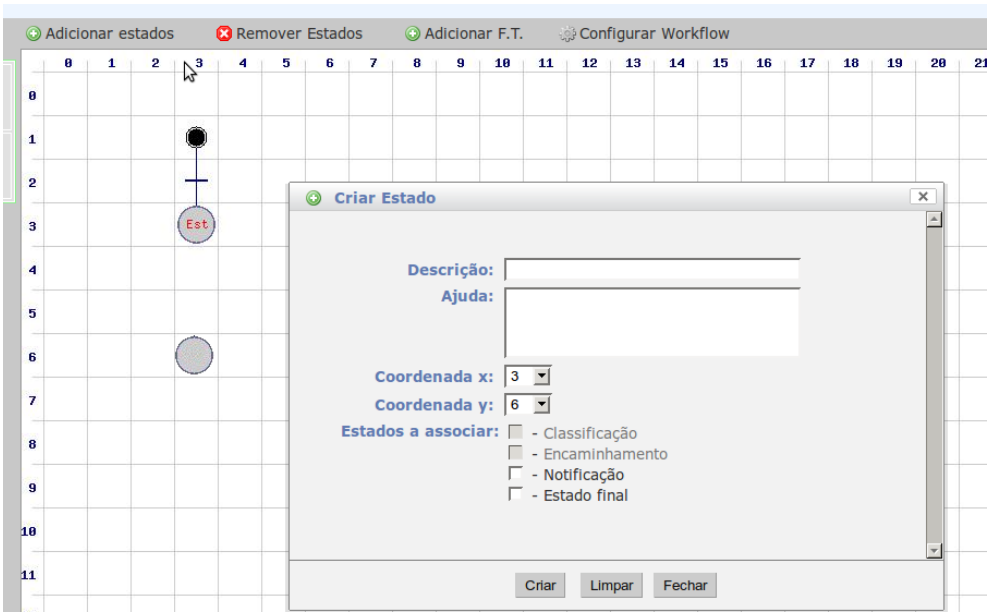

Figura 5.9 - Adicionar estado com *drag and drop*

Outra funcionalidade consiste na inserção de funções de transferência que farão a ligação entre estados. Para tal o utilizador poderá clicar no *link* "Adicionar F.T." ou arrastar o

elemento correspondente à função de transferência. Será de seguida mostrado um painel (figura 5.10) onde o utilizador deverá escolher o estado anterior e estado seguinte que pretende ligar e as coordenadas onde ficará localizada a função de transferência (caso seja escolhida a funcionalidade *drag and drop* para inserir a função de transferência as coordenadas serão automaticamente inseridas no painel).

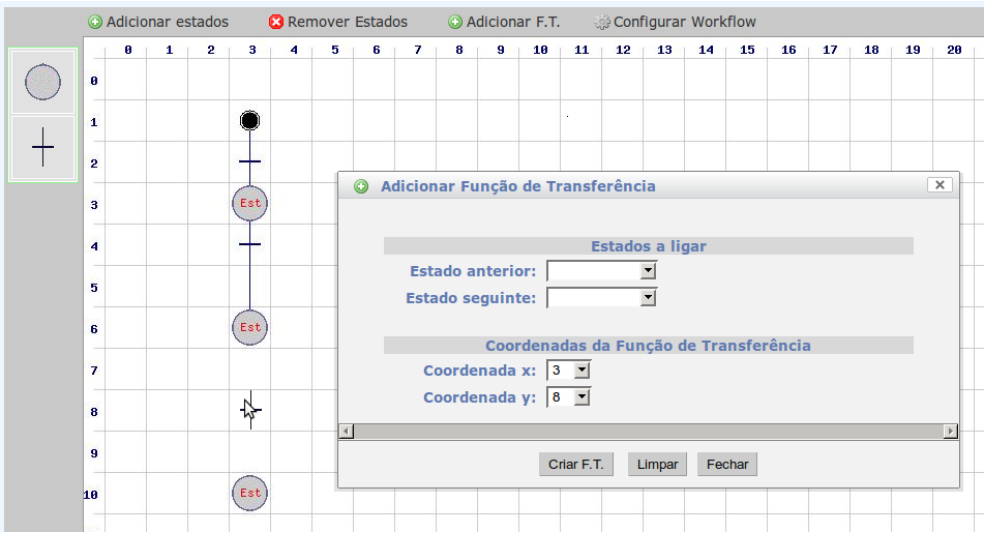

Figura 5.10 - Inserir função de transferência

De modo a facilitar a modificação de estados aquando da criação do *workflow* virtual foi implementada uma funcionalidade onde o utilizador poderá eliminar o número de estados que desejar, ao invés de os eliminar individualmente. Para tal basta clicar no *link* "Remover Estados" e será mostrado um painel com uma listagem dos estados já criados (figura 5.11). Depois de seleccionados os estados que pretende eliminar basta clicar em "Remover" e os estados seleccionados serão removidos. Caso os estados a remover tenham funções de transferência associadas, estas também serão eliminadas.

Figura 5.11 - Eliminar estados

Outro dos menus criado foi o de configuração de *workflow* virtual. Este menu é acedido clicando no ícone "Configurar *workflow*" e dará acesso a um painel onde o utilizador poderá modificar o nome do *workflow* e apagar o *workflow* seleccionado. Este menu pode ser visto na figura 5.12.

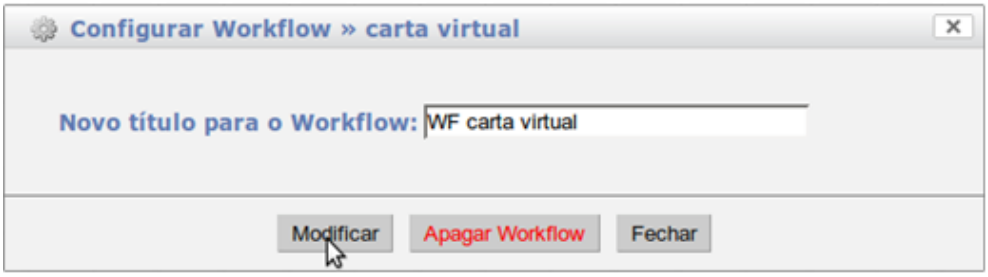

Figura 5.12 - Configurar *workflow*

Todos os campos de inserção obrigatória assim como todas as acções de eliminar ou modificar estão sujeitas a uma confirmação de modo a garantir que a acção que pretende desenvolver é a correcta e não um erro. Na figura 5.13 pode ser visto um menu de alerta que é despontado quando o utilizador clica no botão "Modificar" do menu "Configurar *Workflow*".

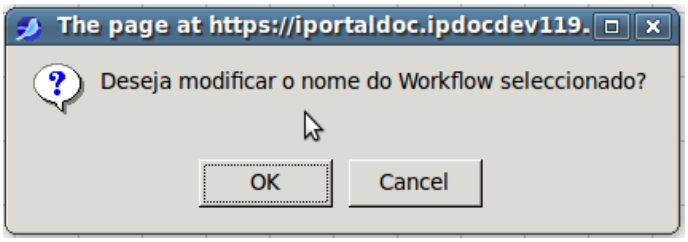

Figura 5.13 - Alert window

Na figura 5.14 é possível ver o painel que permite modificar um estado previamente criado. Para tal basta a um utilizador clicar num estado e ser-lhe-á apresentado um painel com os campos já preenchidos com os dados do estado escolhido e onde será também possível apagar o estado, sendo para tal apenas necessário clicar no botão "Apagar Estado".

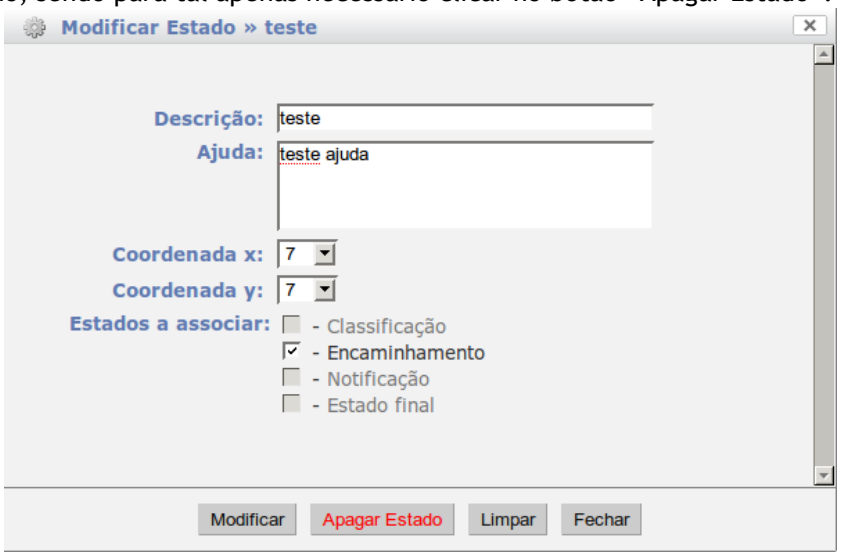

Figura 5.14 - Modificar estado

Para a modificação das funções de transferência foi criado um painel idêntico ao mostrado na figura 5.10, no entanto o utilizado terá também ao seu dispor um botão que lhe permitirá eliminar a função de transferência escolhida. Tal como para a modificação de estados, o acesso ao painel referido acima é realizado clicando numa função de transferência previamente criada.

No *iPortalDoc-Light* foi criada uma interface que permite a um utilizador visualizar o *workflow* virtual que esteja associado ao documento que pretende visualizar. Esta opção encontra-se disponível só se o *workflow* virtual estiver criado, caso contrário será mostrado o *workflow* original. Na figura 5.15 é mostrada a interface de informação de um documento que possui associado a si um *workflow* virtual. Como se pode ver no canto superior direito existe um *link* para o *workflow* virtual ("*Workflow Light*"), caso não existisse um *workflow* virtual associado apareceria o link "*Workflow*" que apontaria para o *workflow* original.

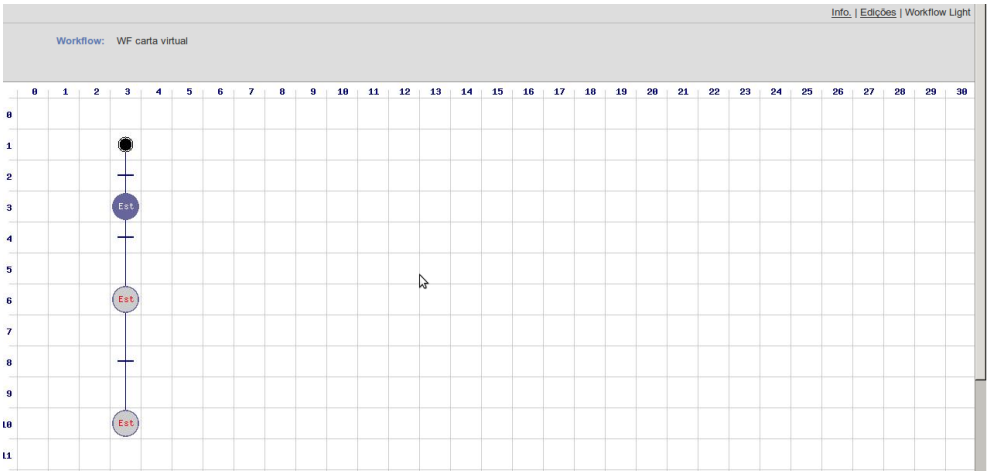

Figura 5.15 - *Workflow* virtual *iPortalDoc-Light*

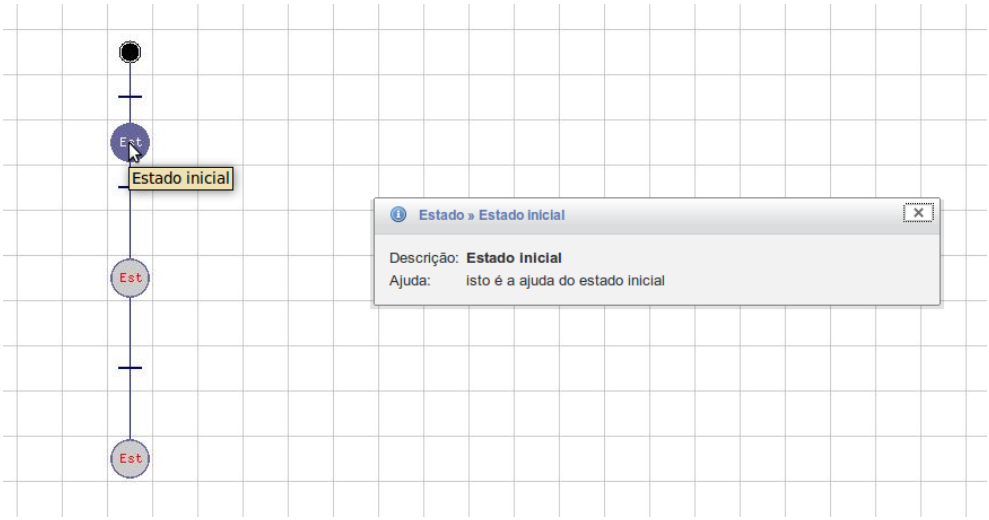

Figura 5.16 - Ajuda do *workflow* virtual

Como pode ser visto na figura 5.15, ao clicar num estado será mostrado o nome desse estado, bem como um painel de ajuda onde se pode ver o nome do estado e a respectiva ajuda.

5.3- Menu de templates

De seguida serão apresentadas as interfaces do módulo menu templates.

Tal como o módulo anterior, este encontra-se inserido no menu que permite configurar o *iPortalDoc-Light* ("Definições -> iPortalDoc-Light -> Templates"), e só estará disponível para um utilizador com permissões de *super user*.

Na figura 5.17 pode ser vista a interface principal deste módulo. É constituída por duas *tabs* onde o utilizador tem acesso, na primeira, à interface onde poderá configurar o modo como a inserção de documentos será apresentada no *iPortalDoc-Light* e, na segunda, onde poderá modificar um documento previamente criado.

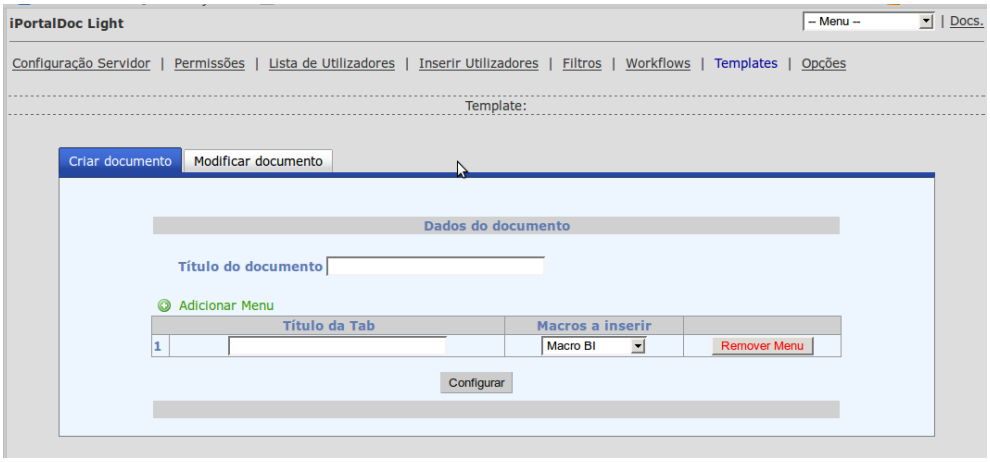

Figura 5.17 - Interface inicial

Na imagem abaixo pode ser visto como são criados novos campos, para tal basta clicar no *icon* "Adicionar Menu" e é automaticamente criado um menu que o utilizador deverá preencher. Do lado direito de cada menu existe um botão "Remover Menu" (figura 5.19) que permite eliminar o menu seleccionado. Todos os campos são de preenchimento obrigatório, sendo que é impossível configurar o documento sem ter todos os campos preenchidos. Caso o utilizador clique no botão "Configurar" tendo algum campo por preencher, será apresentado um alerta de modo a avisar o utilizador que um dos campos se encontra em branco. Ao clicar em "Configurar" e depois de todos os campos terem sido validados, o documento será criado.

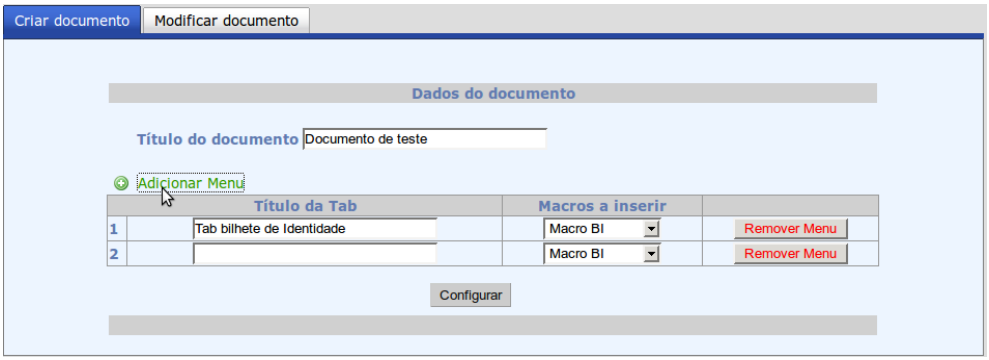

Figura 5.18 - Adicionar menu

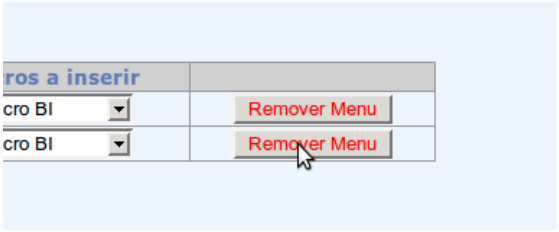

Figura 5.19 - Remover menu

Na figura 5.20 pode ser vista a interface onde é possível modificar um documento previamente criado. Para tal é necessário escolher um documento de uma lista que é apresentada em forma de *drop down* menu. Depois de seleccionado um documento é apresentado uma interface, em tudo semelhante ao criar documento, onde os dados do documento seleccionado já se encontram nos campos, tornando assim a sua modificação mais fácil e intuitiva. Esta interface permite também eliminar um documento, bastando para tal clicar no botão "Eliminar" sendo a acção confirmada através de uma caixa de confirmação, como pode ser visto na figura 5.21.

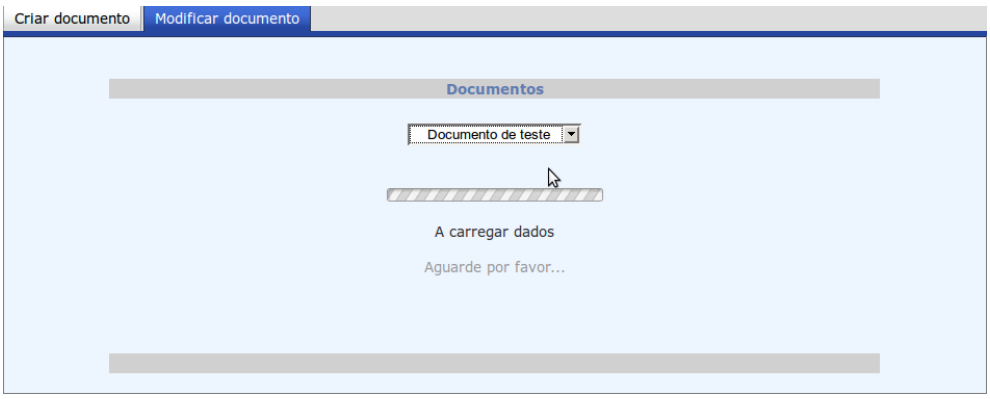

Figura 5.20 - Modificar documento

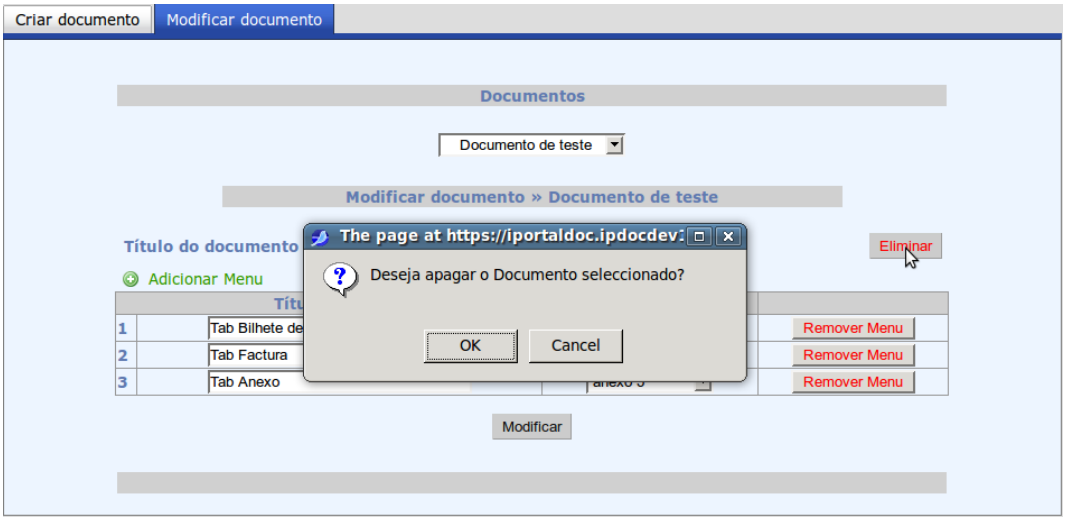

Figura 5.21 - Eliminar documento

No *iPortalDoc-Light* foi criado um menu que permite seleccionar um dos documentos previamente criados no *iPortalDoc* (figura 5.22).

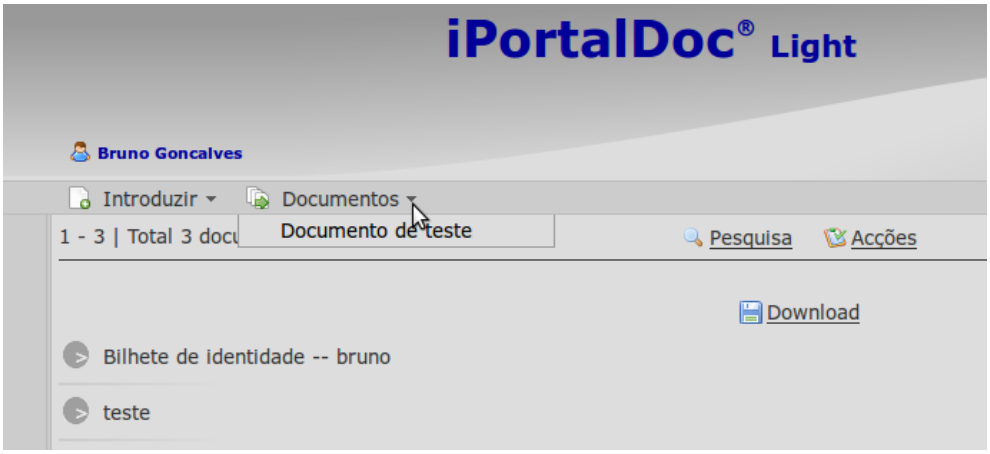

Figura 5.22 - Seleccionar documento no *iPortalDoc-Light*

Depois de seleccionado um documento será mostrada no menu central do *iPortalDoc-Light* uma interface baseada em *tabs*, onde em cada *tab* o utilizador poderá adicionar um diferente documento, como pode ser visto na figura 5.23.

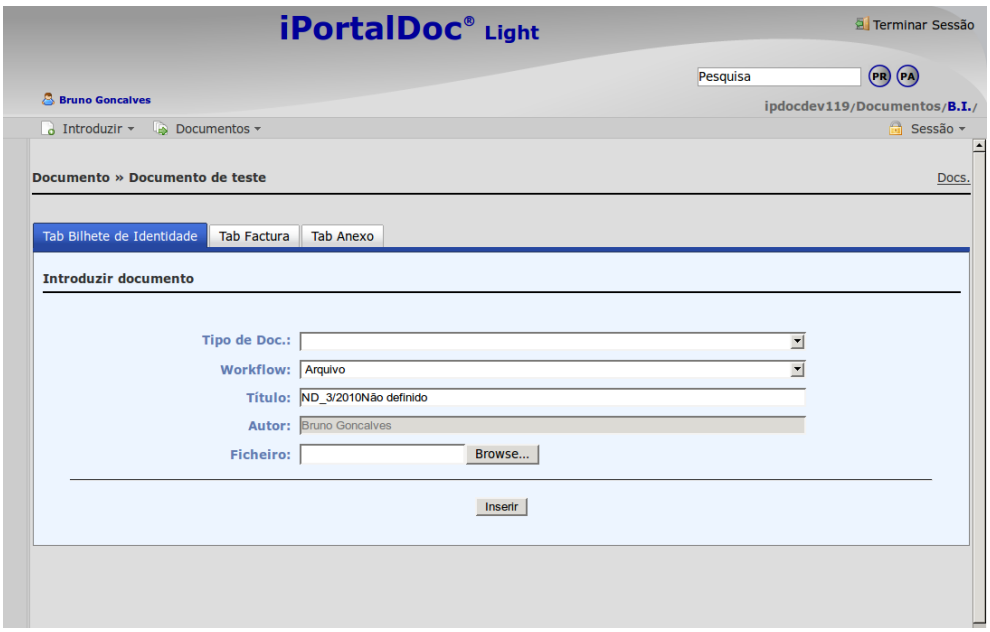

Figura 5.23 - Menu de tabs

Figura 5.24 - Transição entre *tabs*

Na figura 5.24 pode ser vista a transição entre tabs.

Visto ter sido usado AJAX para efectuar a transição entre tabs e para compensar os efeitos de uma latência elevada, durante o tempo em que é feito o pedido e é recebida a resposta é mostrado uma mensagem de espera, isto pode ser visto no centro da figura 5.24.

5.4- Conclusões

Como foi demonstrado nos pontos anteriores, as interfaces desenvolvidas para cada um dos módulos, cumprem os requisitos que foram inicialmente propostos para cada um.

O uso do componentes da Framework YUI, como o “tabview” que permitiu tornar as transições entre menus mais rápidas e intuitivas ou o “panel” que facilitou a apresentação de formulários ao utilizador, tornaram as aplicações desenvolvidas mais funcionais, rápidas e interactivas.

No que toca à compatibilidade, foram efectuados testes em vários browsers de modo a validar o correcto funcionamento em todos eles, ou seja, um utilizador não necessita de se restringir ao uso de um browser específico para usufruir dos módulos desenvolvidos. No entanto, como foi referido no capítulo 2, existem versões mais antigas de browser que não são suportadas pelas *frameworks* e por conseguinte o uso dessas versões pode levar a falhas no correcto funcionamento das aplicações.

Capítulo 6

Conclusões e trabalho futuro

6.1- Conclusões

Actualmente, com o uso em massa de computadores e análogo crescimento da internet, a utilização de aplicações *Web* para melhorar a eficiência e auxiliar no bom funcionamento de uma empresa tornam-se cada vez mais relevantes. Como foi mostrado neste documento o recurso a sistemas electrónicos de gestão documental e de *workflow* estão, cada vez mais, a tornar-se uma ferramenta essencial ao bom funcionamento de uma empresa, melhorando a interacção entre pessoas e processos e reduzir a utilização de papel num ambiente empresarial.

O *iPortalDoc* é um SGDW que fornece todas as características relevantes num sistema deste tipo, desde gestão documental, fluxos de trabalho, integração de templates, motor de pesquisa, entre muitas outras. Tem, também, um módulo *Web* inovador que permite a consulta de informação a entidades externas, o *iPortalDoc-Light*. Com o intuito de melhorar as características funcionais desta aplicação foram desenvolvidos módulos que quando integrados com o *iPortalDoc-Light* permitirão melhorar o funcionamento deste.

No desenvolvimento destes módulos esteve presente a preocupação de criar funcionalidades que fossem interactivas, práticas e compatíveis com os vários browsers, de modo a facilitar a integração dos mesmos com as características dos utilizadores e empresas a que são dirigidas.

No que toca à escolha das linguagens de programação usadas no desenvolvimento dos módulos, depois do estudo efectuado no capítulo 2, ponto 2.3, esta recaiu sobre o PHP e o *JavaScript*, duas linguagens essenciais ao desenvolvimento de qualquer aplicação *Web*. Foram também utilizadas as *frameworks YUI* e *Prototype* de modo a melhorar a funcionalidade da linguagem *JavaScript* e de forma a permitir o desenvolvimento de interfaces mais funcionais e interactivas. Embora as *frameworks* utilizadas tenham já as suas qualidades estabelecidas, a escolha teve como principal factor de preferência o já se encontrarem em uso nas aplicações disponibilizadas pela empresa. No entanto, o uso de *jQuery* poderia ser uma mais-valia, não só pela sua robustez e inúmeras aplicações, como também pela elevada qualidade de documentação disponível. O recurso ao CSS possibilitou o desenvolvimento de interfaces com um aspecto apelativo e permitiu que estas se comportassem da mesma forma nos diferentes

browsers. A escolha do sistema de base de dados recaiu sobre o *PostgreSQL*, mais uma vez por ser o sistema já suportado pela aplicação.

Durante a implementação foram encontrados alguns problemas com a compatibilidade entre *browsers*, principalmente no Internet Explorer (IE), isto deveu-se em grande parte ao uso de *JavaScript* e do não reconhecimento, por parte do IE, de certas funcionalidades desta linguagem. De forma assegurar que as aplicações se comportavam de igual forma em todos os browsers, foi necessário adaptar o código a cada browser, no entanto, este problema foi maioritariamente solucionado recorrendo ao uso de *frameworks*, permitindo obter um correcto funcionamento nos diferentes *browsers*. Um exemplo de um problema com que nos deparamos foi no uso de AJAX e "innerHTML" que funcionando perfeitamente em browsers com o Firefox e Opera, não funcionava no IE. Este problema foi solucionado com o uso da framework *Prototype* para a implementação do pedido AJAX. No entanto, é de referir que, como pode ser visto no estudo presente no capítulo 2 sobre *JavaScript*, as *frameworks* só suportam as versões mais actuais dos browsers, como tal será de ter em atenção o funcionamento em versões mais antigas dos mesmos, pois poderão surgir alguns problemas.

O balanço final é positivo tendo em conta todas as funcionalidades implementadas e a realização dos requisitos definidos para cada uma por parte da empresa, desde a facilitação do menu de pesquisa para os utilizadores, a gestão de interfaces de templates e a gestão de *workflows* virtuais. Todas estas medidas são implementações práticas e funcionais, tendo sempre em vista a optimização do software de gestão documental sobre o qual incidem.

6.2- Trabalho futuro

Só através da reflexão do trabalho realizado se poderá inferir acerca do que pode ser melhorado ou complementado.

Com o estudo da arquitectura do sistema, com o levantamento exaustivo das funcionalidades actuais do software, com o estudo dos meios onde sistemas deste tipo podem ser integrados, com o levantamento das linguagens de programação, deduz-se que este projecto proporcionou medidas relevantes e interessantes que poderão ser levadas em conta na melhoria estratégica do *iPortalDocLight*.

Como trabalho futuro e tendo em conta todo o trabalho desenvolvido durante estes meses, será produtivo a continuação de projectos idênticos com vista à melhoria contínua de sistemas de gestão documental como este.

No módulo de pesquisa, os campos de pesquisa podem ser alargados alem dos requisitos que foram inicialmente propostos e poderá ser implementada a funcionalidade de *auto-complete* ao menu de pesquisa facilitando assim a pesquisa e tornando-a mais intuitiva.

No menu de *workflow* poderá ser dada a opção de associar a cada *workflow* mais do que um *workflow* virtual, alargando assim a abrangência desta funcionalidade. Desta forma poderia ser associado a um documento tipos diferentes de *workflow* virtual, diferenciando assim a informação fornecida, consoante a entidade que consultasse o *iPortalDoc-Light*.

A implementação do módulo de assinatura digital deverá ser integrada com o *iPortalDoc-Light*.

Tornar o *iPortalDoc-Light* um módulo autónomo. Actualmente o *iPortalDoc-Light* funciona interligado com *iPortalDoc*, sendo que todas as configurações são realizadas no *iPortalDoc*. A

passagem destas configurações para o *iPortalDoc-Light* torná-lo-ia um módulo realmente externo.

Quanto mais se inovar e se facilitar a utilização de sistemas como este, mais serão as vantagens na sua utilização.

Anexo A

A.1

IPBrick.IC - é o sistema operativo da *IPBrick* que fornece através de uma única tecnologia Intranet, Comunicações Unificadas e Segurança no acesso à Internet. Deste modo, a *IPBrick* tem a sua componente de *IPBrick.I* que é um servidor de *Intranet*: suporta as aplicações de negócio essenciais para a gestão de uma empresa, fornece as Ferramentas Colaborativas (E-mail, Contactos e Agenda) e é servidor de domínio, ficheiros, impressoras e base de dados. Garante ainda o *backup* das áreas de trabalho. Tem ainda a componente da *IPBrick.C*, um servidor de comunicações e segurança: fornece a protecção - *Firewall*, IDS, VPN, *Anti-Spam* e Anti-vírus pré-instalado e *Proxy* - para as suas comunicações - E-mail, Voz (serviço VoIP), *Chat* Profissional, *Fax2Mail* e *Mail2Fax*, *Mail2SMS* e Vídeo.

Para responder aos vários segmentos do mercado onde opera, a *IPBrick* desenvolveu cinco *appliances*, com o objectivo de vender um produto fechado (software *IPBrick.IC* + hardware) segundo as necessidades dos seus clientes:

IPBrick.GT - Fornece Voz, Vídeo, Fax, Mail, *Web*, SMS e *Instant Messaging* numa única *appliance*, sendo que a *IPBrick* foi pioneira na implementação do conceito UCoIP (sigla anglo-saxónica para Unified Communications over IP). A *IPBrick.GT* tanto pode desempenhar funções de média gateway, como também integra com centrais telefónicas existentes, garantindo a compatibilidade com qualquer equipamento já instalado.

IPBrick.KAV - É uma *appliance* de segurança: impede que as estações de trabalho se liguem directamente à Internet, não permitindo que programas do tipo trojan estabeleçam túneis com o exterior, ou abram *backdoors* que permitam o acesso à rede da empresa a partir do exterior. Nesse sentido, a *IPBrick.KAV* dispõe de um conjunto de serviços funcionando em modo Proxy. O software de Segurança da *Kaspersky* Anti-Virus elimina na protecção perimetral assumida pela *IPBrick.KAV* todos os conteúdos de malware (*trojans*, *worms*, *spyware*, *phishing*, vírus, etc) que se destinam a infectar os computadores da rede interna. Para além da nova função de auditor de Segurança para Intranet, é capaz de verificar e identificar quais as estações de trabalho infectadas, e que comprometem a segurança da i*ntranet*.

IPBrick.H - Solução de Gestão de Internet em *Hotspots* (por cabo ou por *wireless*), através de uma interface muito simples. Emite *vouchers* de utilização quer em valores (dinheiro como, por exemplo, num cibercafé) ou em horas (por exemplo uma biblioteca).

IPBrick.LIVE - É uma solução para *Corporate* TV: é uma solução simples e completa que gere e reproduz conteúdos multimédia: exibe filmes, publicidade, notícias e qualquer tipo de

informações. Além disso, exibe informações produzidas pelo *iMeter* e suporta sistemas de bilhética.

IPBrick.SCHOOL - A *IPBrick.SCHOOL* é uma tecnologia usada na aprendizagem de TI. Esta solução fornece um *desktop* multi-sessão e está pensada para o ambiente escolar do futuro, baseado num conjunto de terminais ligeiros ligados a um servidor *IPBrick* (em vez de uma dispendiosa infra-estrutura de computadores). Com esta tecnologia, qualquer escola pode instalar um servidor em 30 minutos ligado a mais de 64 terminais com dois ou mais sistemas operativos diferentes nas estações de trabalho dos estudantes! Tudo num processo 100% automatizado.

IPBrick.SOHO - *IPBrick.SOHO* é uma solução que fornece num único equipamento duas ferramentas essenciais em todos os escritórios - PBX e Fax. Para além disso, também fornece e-mail, *sms*, é servidor *Web* e de mensagens instantâneas, todos integrados num único aparelho. O servidor de intranet da *IPBrick.SOHO* possui servidores de ficheiros, impressoras e bases de dados. Integra um ambiente verdadeiramente colaborativo com e-mail, livro de endereços e calendário. Permite ainda fazer *backup* remoto de dados e configurações. A *IPBrick.SOHO* fornece um conjunto completo de serviços de *Proxy* para evitar que as estações de trabalho liguem directamente à Internet. O servidor VPN proporciona acesso remoto completo aos recursos digitais da empresa.

Referências Bibliográficas

[1] **Mathieu, Miles L.** *THE PAPERLESS OFFICE: ACCEPTING DIGITIZED DATA*.

[2] **Teixeira, João Parreira e Miguel.** Porquê Gestão Documental? *Porquê Gestão Documental?* http://www.dalera.com/cache/imagens/XPQnHCP8s7888Bu21XzJtmwZKU.pdf. [Acesso em: 05-21-2010].

[3] **System, Enterprise Content Management.** What are the business benefits of using a Document Management System (DMS)? *Contentmanager*. Contentmanager, 2000. http://www.contentmanager.eu.com/dmsbens.htm. [Acesso em: 05-22-2010].

[4] **Coalition, Workflow Management.** *Workflow Management Coalition Terminology & Glossary*. 1999.

[5]**Administrativa, Agência para a Modernização.** Cartão do Cidadão. *Cartão do Cidadão*. http://www.cartaodecidadao.pt/index.php?option=com_content&task=view&id=38&Itemid=35&lang=pt. [Acesso em: 02-08-2010].

[6] **IPBRICK, iPortalMais -.** IPBRICK - Simplesmente Convincente! *iPortalMais*. iPortalMais. [Acesso em: 02-07-2010].

[7] **IPBrick, iPortalMais -.** IPBrick. *IPBrick*. iPortalMais. http://www.ipbrick.com/. [Acesso em: 03-12-2010].

[8] **iPortalDoc, iPortalMais -.** iPortalDoc. *iPortalDoc*. iPortalMais, 2006. http://www.iportalmais.pt/index.php?oid=88. [Acesso em: 02-08-2010].

[9] **PHP-Wikipedia.** PHP-Wikipedia, the free encyclopedia. *Wikipedia*. Wikipedia, 2010. http://en.wikipedia.org/wiki/Php. [Acesso em: 02-08-2010].

[10] **Perl.org.** What is Perl. http://faq.perl.org/perlfaq1.html#What_is_Perl_. [Acesso em: 04 02 2010].

[11] **Martin Arlitt, Lance Titchkosky e Carey Williamson.** *A Performance Comparison of Dynamic Web Technologies*. 2003.

[12] **IBM.** Compare JavaScript frameworks. *IBM*. IBM. http://www.ibm.com/developerworks/web/library/wa-jsframeworks/. [Acesso em: 06-21-2010].

[13] **JavaScript-Wikipedia.** JavaScript-Wikipedia, the free encyclopedia. *Wikipedia*. Wikipedia, 2010. http://en.wikipedia.org/wiki/JavaScript. [Acesso em: 02-08-2010].

[14] **W3C.** HTML & CSS. *W3C*. W3C. http://www.w3.org/standards/webdesign/htmlcss. [Acesso em: 04-21-2010.]

[15] **Canonical.** Ubunto. *Ubunto*. Canonical, 2010. http://www.ubuntu.com/. [Acesso em: 06-21-2010].

[16] **DEV, KDE WEB.** Quanta plus. *Quanta plus*. KDE WEB DEV, 2005. http://quanta.kdewebdev.org/. [Acesso em: 06-21-2010].

[17] **Foundation, Mozilla Europe e Mozilla.** Firefox o navegador Web. *mozilla-europe*. Mozilla Europe e Mozilla Foundation. http://www.mozilla-europe.org/pt/firefox/. [Acesso em: 06-21-2010].

[18] **W3Schools.** Browser Statistics. *W3Schools*. W3Schools. http://www.w3schools.com/browsers/browsers_stats.asp. [Acesso em: 07 02 2010.]

[19] **Group, PostgreSQL Global Development.** PostgreSQL. PostgreSQL Global Development Group, 1996. http://www.postgresql.org/about/. [Acesso em: 07 02 2010.]

[20] **Cervisia.** Cervisia. *Cervisia*. Cervisia, 2005. http://cervisia.kde.org/. [Acesso em: 06-21-2010].

[21] **w3schools.** JavaScript Popup Boxes. *w3schools*. w3schools. http://www.w3schools.com/js/js_popup.asp. [Acesso em: 06-21-2010].

[22] **Project, The jQuery.** jQuery. *jQuery*. 2010. http://jquery.com/. [Acesso em: 2-7-2010].

[23] **Lennon, Joe.** Compare JavaScript frameworks. *IBM*. IBM, 02 02, 2010. http://www.ibm.com/developerworks/web/library/wa-jsframeworks/#ibm-pcon. [Acesso em: 06 02, 2010].

[24] **Firebug.** Firebug. *What is firebug*. Mozilla, 2005. http://getfirebug.com/whatisfirebug. [Acesso em: 06-21-2010].