

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



FEUP

Search Technologies, the Future Enterprise Applications

Luís Miguel Puim Alves

VERSÃO FINAL

Relatório de Dissertação

Mestrado Integrado em Engenharia Informática e Computação

Orientador: António Miguel Pontes Pimenta Monteiro (Prof. Doutor)

Março de 2010

Search Technologies, the Future Enterprise Applications

Luís Miguel Puim Alves

Relatório de Dissertação

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Ana Cristina Ramada Paiva Pimenta (Prof^a Doutora)

Arguente: Daniel Coelho Gomes (Doutor)

Vogal: António Miguel Pimenta Monteiro (Prof. Doutor)

1 de Março de 2010

Resumo

Com o ritmo veloz com que se vive actualmente, o mundo empresarial para prosperar tem que adaptar os seus processos de negócio a esse mesmo ritmo, seguindo a velha máxima que “tempo é dinheiro”. Os processos clássicos, morosos e dependentes de muitas toneladas de papel, passam a ser desempenhados por poderosos sistemas de informação, não só facilitando, mas também tornando mais céleres e amigos do ambiente estes processos.

Com o uso generalizado de plataformas de Tecnologia de Informação, a informação relevante para os processos da empresa tende a dispersar-se por essas plataformas, podendo ser encontrada numa forma estruturada (em bases de dados, XML, etc.), ou na maioria dos casos de forma desestruturada, como é o exemplo de ficheiros em diversos formatos, quer seja texto, áudio ou vídeo. Estudos apontam para que esta distribuição seja na ordem dos “20-80”, onde 20% da informação é estruturada e a restante é desestruturada.

Esta dissertação enquadra-se na área da recuperação de informação, mais precisamente no seu contexto empresarial, vulgarmente designado por “*Enterprise Search*”. Este modelo de pesquisa distingue-se, em vários aspectos, dos mais comuns modelos de pesquisa que são a “*Web Search*” e a “*Desktop Search*”.

A pesquisa empresarial utiliza uma ou mais fontes de informação, denominadas repositórios, onde a informação guardada pode estar em diversos formatos. Outro aspecto que a diferencia dos outros tipos de pesquisa é a segurança da informação, isso porque ao nível empresarial existe informação que só pode ser acedida por alguns funcionários ou grupos de funcionários (papéis). Desta forma, os resultados da pesquisa apenas podem apresentar resultados dos quais o utilizador, previamente autenticado, esteja autorizado a ver.

Dado que na PTSI existem diversos sistemas que suportam as suas actividades e possuem grande quantidade de informação em diversos formatos, tornam-se visíveis as mais-valias que uma plataforma deste tipo pode trazer para a empresa. O objectivo foi integrar numa ferramenta já existente, denominada SPOT, funcionalidades de pesquisa sobre vários repositórios de informação da PTSI, utilizando a plataforma FAST ESP considerada como uma das melhores na sua área.

Foi necessário investigar outros motores de pesquisa corporativa existentes no mercado e verificar as suas mais-valias e desvantagens face ao contexto aplicacional existente. Após a selecção da melhor plataforma foi necessária a sua configuração, de forma a obter os resultados de acordo com os requisitos impostos.

Para a implementação de um *front-end* de pesquisa foram utilizadas tecnologias recentes: o Silverlight em conjunto com *web services*. A utilização de Silverlight permitiu não só a completa integração no SPOT como também apresentar uma interface final bastante rica graficamente.

O *front-end* desenvolvido vai possibilitar aos colaboradores encontrar a informação que desejam, mesmo que esta se encontre dispersa por vários repositórios dentro da organização. Desta forma economizam tempo, pois não têm de aceder a diversos sistemas, e aumentam assim a sua produtividade.

Outro aspecto focado foi o da relevância dos resultados para cada um dos colaboradores, isto é, cada um dos colaboradores deve obter um conjunto de resultados que sejam os mais relevantes para a função que desempenha. Verificou-se que se trata de é uma tarefa que requer tempo e que necessita de continuar a ser aperfeiçoada até que sejam alcançadas as expectativas de cada um dos colaboradores.

Palavras-chave: Recuperação de Informação, Pesquisa Empresarial, Indexação, Relevância, Segurança, *Webservices*, Silverlight.

Abstract

With the fast paced lives that we live today, the business world to thrive needs to adapt his business process to keep up with this pace, following the old cliché that “time is money”. The classic processes, which were time consuming and relied on many tons of paper, are now being performed by powerful computer systems, not only making them easier but also speeding them up and being environmentally friendly.

With the generalized use of IT platforms, the relevant information for the company processes tend to disperse across those platforms, and may be found in a structured form (Databases, XML, among others), or in the most common scenarios in an unstructured form, in which we have the example of diversified file formats, whether it is text, audio or video. Studies point toward a "20-80" distribution, where 20% of the information is structured and the remaining is unstructured.

This project is included within the information retrieval field, more precisely on an enterprise context, commonly referred as "Enterprise Search". This search model distinguishes, in several aspects, from the most common search models called "Web Search" and "Desktop Search".

The Enterprise Search (ES) makes use of one or more sources of information, called repositories, where the stored information may be in several formats. Another aspect that differentiates it from the other types of search is the information security, because at the enterprise level there is information that can only be accessed by some employees or a group of employees (roles). Thus, the results of a query can only retrieve results that a user, formerly authenticated, is allowed to see.

Since in PTSI, there are a variety of systems that support their activities and store a large amount of information in several formats, the advantages that a platform of this kind can offer to the company becomes apparent. The goal of this project was to integrate in a tool that already existed, called SPOT, functionalities of search over some information repositories of PTSI, making use of the FAST ESP platform capabilities, considered as one of the best in its field.

It was necessary to survey other enterprise search platforms available in the market and weight their advantages and disadvantages given the existing application context. After selecting the best platform, it was necessary to do the required configurations, in a way to obtain the results in accordance with the requirements.

For the development of the search front-end, there were used recent technologies like: Silverlight together with web services. The use of Silverlight allowed not only the complete integration of the search front-end on Spot, as well to create a rich user interface.

The front-end developed will give a chance to the employees to find the information that they require even if it is disperse by several repositories inside the organization. Thus, they will save time, by not having to access a variety of systems, and will increase their productivity.

Concerning the aspect of the results relevance being different for different employees, meaning, each one of the employees should obtain a set of relevant results for the task they perform inside the organization, it was found that it's a feature that requires some time to be properly achieved and will continue being improved until the expectations of each one of the employees are met.

Keywords: Information Retrieval, Enterprise Search, Indexing, Relevance, Security, Web Services, Silverlight.

Agradecimentos

É com enorme satisfação que expresso aqui o mais profundo agradecimento a todos aqueles que tornaram a realização deste trabalho possível.

Gostaria de agradecer em primeiro lugar ao Professor António Miguel Pontes Pimenta Monteiro, orientador desta tese, pelo apoio e disponibilidade demonstrada em todas as fases.

Gostaria ainda de agradecer:

Ao Nuno Miguel Viana Baptista, orientador da tese na empresa pelos comentários e sugestões.

À empresa PTSI, pela oportunidade que me deu de mostrar o meu trabalho neste projecto.

Ao Alexandre Gonçalves, Miguel Vale, Gonçalo Conceição, Miguel Vieira, por todo o apoio prestado na empresa.

E por ultimo a todos os meus amigos e família que sempre me apoiaram.

Luís Alves

Índice

1	Introdução.....	1
1.1	Enquadramento	2
1.1.1	Empresa PT-SI.....	3
1.2	Motivação e Objectivos.....	3
1.3	Estrutura da Dissertação	4
2	Pesquisa Empresarial	5
2.1	Introdução à Recuperação de Informação.....	5
2.1.1	Indexação	5
2.1.2	Relevância	6
2.1.3	Processamento dos documentos e do texto de pesquisa	8
2.2	Estado da arte	10
2.2.1	Disponibilidade.....	10
2.2.2	Escalabilidade e Performance	10
2.2.3	Transparência.....	13
2.2.4	Segurança.....	14
2.2.5	Análise da Pesquisa	15
2.2.6	Pesquisa Semântica.....	17
2.2.7	Relevância	18
2.2.8	Refinação de resultados.....	19
3	Análise e Planeamento.....	20
3.1	Objectivos.....	20
3.2	Análise das plataformas de pesquisa	20
3.2.1	Resumo dos estudos da PTSl.....	21
3.3	Especificação de requisitos	22
3.3.1	Requisitos funcionais	22
3.3.2	Requisitos Não Funcionais	29

3.3.3	Requisitos de Informação	30
3.4	Análise Tecnológica	30
3.5	Arquitectura	31
3.5.1	Visão geral	31
3.5.2	Arquitectura Lógica	31
3.5.3	Arquitectura Física e Tecnologias.....	33
3.6	Planeamento do desenvolvimento	34
4	Desenvolvimento	36
4.1	<i>Front-end</i> de Pesquisa.....	36
4.1.1	Vista geral da ESP Search API	36
4.1.2	Search API Web Service Query Interface	38
4.1.3	Desenvolvimento do <i>Front-End</i>	39
4.2	Afinação dos Resultados da FAST ESP.....	43
4.2.1	<i>Search Business Center</i>	43
4.2.2	<i>Matching Engine Profile</i>	46
4.2.3	Trabalho de Afinação	48
5	Conclusões e Trabalho Futuro.....	51
5.1	Satisfação dos Objectivos.....	51
5.2	Trabalho Futuro.....	51
6	Referências.....	53

Lista de Figuras

Figura 1.1 - Distribuição 80-20 da informação [4]	2
Figura 2.1 - Fórmula matemática para o conceito de precisão [8].....	6
Figura 2.2 - Fórmula matemática para o conceito de <i>recall</i> [8].....	6
Figura 2.3 - Exemplo do Modelo Espaço Vectorial [9]	7
Figura 2.4 - Modelos de relevância [8]	8
Figura 2.5 - Matriz de nós de pesquisa [16].....	11
Figura 2.6 - Exemplos de <i>benchmarks</i> [18]	12
Figura 2.7 - Segurança não mapeada [15].....	15
Figura 2.8 - Segurança mapeada [15]	15
Figura 3.1 - Gráfico de comparação das plataformas de pesquisa empresarial [21].....	22
Figura 3.2 - Visão Geral dos Casos de Uso	23
Figura 3.3 - Maqueta: Efectuar Pesquisa Simples	24
Figura 3.4 - Maqueta: Utilizar auto completar numa pesquisa	25
Figura 3.5 - Diagrama de estados: Filtrar Resultados baseado em Navegadores	26
Figura 3.6 - Maqueta: Filtrar Resultados baseado em Navegadores.....	26
Figura 3.7 - Maqueta: Navegar pelas páginas de resultados	27
Figura 3.8 - Maqueta: Observar Resultados	28
Figura 3.9 - Maqueta: Pesquisa por Sugestão	28
Figura 3.10 - Modelo de Dados	30
Figura 3.11 - Visão geral da arquitectura.....	31
Figura 3.12 - Decomposição horizontal do sistema	32
Figura 3.13 - Decomposição vertical do sistema	33
Figura 3.14 - Diagrama da arquitectura física.....	34
Figura 3.15 - Planeamento do desenvolvimento do <i>front-end</i> de pesquisa (parte 1)	35
Figura 3.16 - Planeamento do desenvolvimento do <i>front-end</i> de pesquisa (parte 2)	35
Figura 4.1 - FAST ESP <i>framework</i> de integração [25].....	37
Figura 4.2 - Formas de acesso ao <i>Query & Results Server</i> [26]	37
Figura 4.3 - Arquitectura Search API Web Service [25]	38
Figura 4.4 - Arquitectura de segurança da <i>Search API Web Service</i> [25]	39
Figura 4.5 - Diagrama de classes do módulo cliente	40
Figura 4.6 - Diagrama de classes do módulo de serviços	41
Figura 4.7 - Pesquisa com Auto-Complete integrada no SPOT	42
Figura 4.8 - Aspecto do <i>front-end</i> de pesquisa integrado no SPOT	42

Figura 4.9 - Módulo de finação de pesquisas [26]	43
Figura 4.10 - Interface do SBC [32]	44
Figura 4.11 - Bloquear resultado ou colocá-lo no Top 10 [32].....	44
Figura 4.12 - Aplicar valor de <i>boost</i> manualmente [32]	45
Figura 4.13 - Funcionamento dos sinónimos [32]	45
Figura 4.14 - Definições de um <i>Search Profile</i> [32].....	46
Figura 4.15 - Exemplo de perfis de classificação de resultados em XML.....	47
Figura 4.16 - <i>Search Profiles</i>	48
Figura 4.17 - Sintaxe para atribuir pesos diferentes às colecções.....	50

Lista de Tabelas

Tabela 2.1 - Exemplo de índice	6
Tabela 2.2 - Estatísticas FAST ESP [14]	11
Tabela 2.3 - Conectores FAST ESP [14]	13
Tabela 2.4 - Mecanismos da análise qualitativa [14]	16
Tabela 2.5 – Descrição das métricas de classificação de resultado [14]	18
Tabela 3.1 - Vantagens e desvantagens de cada uma das plataformas [21]	22
Tabela 3.2 - Caso de utilização: Efectuar Pesquisa Simples	24
Tabela 3.3 - Caso de utilização: Utilizar auto completar numa pesquisa	24
Tabela 3.4 - Caso de utilização: Efectuar Pesquisa com Operadores	25
Tabela 3.5 - Caso de utilização: Filtrar Resultados baseado em Navegadores	25
Tabela 3.6 - Caso de utilização: Navegar pelas páginas de resultados	26
Tabela 3.7 - Caso de utilização: Abrir Resultado	27
Tabela 3.8 - Caso de utilização: Observar Resultados	27
Tabela 3.9 - Caso de utilização: Pesquisa por Sugestão	28
Tabela 4.1 – Descrição das métricas de classificação de resultado [14]	46
Tabela 4.2 - Alocação de repositórios aos papéis	49
Tabela 4.3 - Nome das colecções criadas	49

Abreviaturas e Símbolos

ACL	Access Control List
API	Application Programming Interface
AQG	Automatic Query Guidance
CLR	Common Language Runtime
CMS	Content Management System
DAH	Distributed Action Handler
DIH	Distributed Index Handler
ES	Enterprise Search
GB	Gigabyte
GSA	Google Search Appliance
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IDOL	Intelligent Data Operating Layer
MOSS	Microsoft Office SharePoint Server
MVVM	Model, View, View Model
OOTB	Out of the Box
PE	Pesquisa Empresarial
PT	Portugal Telecom
PT-SI	Portugal Telecom – Sistemas de Informação
RI	Recuperação de Informação
RPQ	Real-Time Predictive Query
SAS	Statistical Analysis System
SBC	Search Business Center
SFE	Search Front-End
SOAP	Simple Object Access Protocol
SPSS	Statistical Package for the Social Sciences
SSL	Secure Sockets Layer
SSO	Single sign-on
WSDL	Web Services Description Language
XAML	Extensible Application Markup Language
XDR	eXternal Data Representation
XML	Extensible Markup Language

XSD XML Schema Definition

1 Introdução

Há milhares de anos, que a Humanidade compreendeu a importância de armazenar de forma organizada a informação, para que posteriormente o seu acesso seja facilitado. Podemos remeter este conceito para 3000 a.C., onde os Sumérios criaram o primeiro arquivo historicamente conhecido. Este consistia numa área especial onde eram guardadas as inscrições feitas em tábuas de barro. Eles compreenderam que a organização era crítica para o uso eficiente da informação, desenvolvendo até formas de classificação para identificar cada tábua de barro e o seu conteúdo [1].

Com o passar dos séculos e com a invenção do papel, a necessidade de armazenar e aceder eficientemente à informação escrita começou a ser um objectivo de essencial importância [1], pois o seu volume aumentava de dia para dia.

Ainda na época do papel, apenas alguns profissionais lidavam com o problema de recuperação de informação, como por exemplo os bibliotecários, bancários, funcionários do registo civil e serviços relacionados [2].

Com o aparecimento do computador e principalmente da *World Wide Web*, a necessidade de encontrar informação tornou-se generalizada, pois praticamente todos os dias as pessoas fazem pesquisas no seu motor de busca, ou até mesmo no seu próprio correio electrónico.

Este conceito é denominado contemporaneamente por Recuperação de Informação e pode ser definido da seguinte forma:

“Recuperação de Informação (RI) consiste em encontrar material (normalmente documentos) de natureza desestruturada (normalmente texto) que satisfaz uma necessidade de informação a partir de uma enorme colecção de documentos (que actualmente, se encontram armazenados em computadores) [2]”.

As técnicas de Recuperação de Informação passaram a ser a forma dominante de acesso à informação, ao invés do acesso tradicional de informação estruturada em Bases de Dados [2], em que por exemplo, numa estação de Correios para o funcionário saber informação sobre a nossa encomenda tem de saber necessariamente o número de identificação da encomenda, não havendo a possibilidade de encontrá-la apenas com informações parciais, isto é, parte do número de identificação da encomenda e o último nome do destinatário. Note-se, que esta pesquisa seria feita sem se ter a noção que o número introduzido era um número de encomenda

e o nome era o último nome do destinatário, ou seja, a pesquisa seria feita sobre “informação desestruturada” e em texto corrido.

A “informação desestruturada” refere-se à informação que não tem uma estrutura evidente, semanticamente clara ou facilmente entendida por um computador. Ao contrário da “informação estruturada”, normalmente armazenada em base de dados relacionais [2].

Com o ritmo veloz com que se vive actualmente, o mundo empresarial para prosperar, tem que adaptar os seus processos de negócio a esse mesmo ritmo, seguindo a velha máxima que “tempo é dinheiro”. Os processos clássicos, morosos e dependentes de muitas toneladas de papel, passam a ser desempenhados por poderosos sistemas de informação, não só facilitando, mas também tornando mais céleres e amigos do ambiente estes processos.

Com o uso generalizado e exponencialmente crescente de plataformas de Tecnologia de Informação, a informação relevante para os processos da empresa tende a dispersar-se por essas plataformas, podendo ser encontrada numa forma estruturada (em bases de dados, XML, etc.), ou na maioria dos casos de forma desestruturada, como é o exemplo de ficheiros em diversos formatos, quer seja texto, áudio ou vídeo. Estudos apontam que esta distribuição seja na ordem dos “20-80”, onde 20% da informação é estruturada e a restante é desestruturada [3].

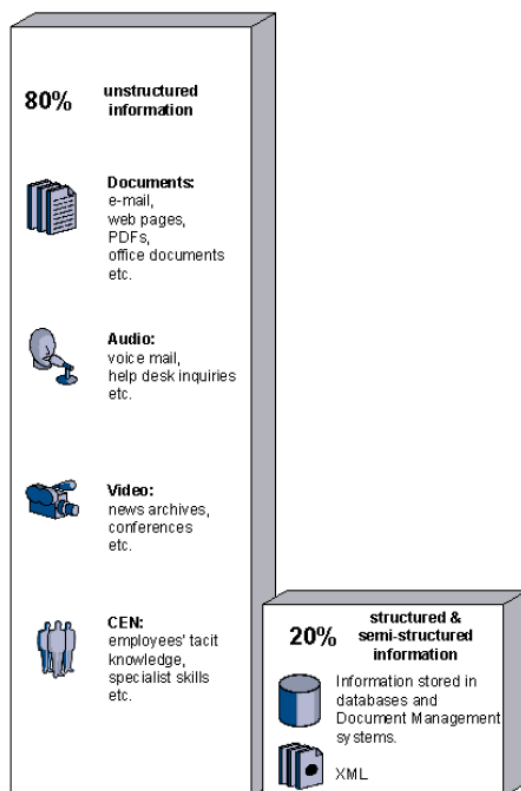


Figura 1.1 - Distribuição 80-20 da informação [4]

Com o grande volume de informação e distribuição que apresenta, numa empresa haver uma eficaz plataforma de pesquisa, torna-se um factor de grande relevo no que diz respeito a competitividade e prestação de serviço. Encontrar rapidamente a informação que necessitamos beneficia de forma óbvia a eficiência do serviço prestado [3].

1.1 Enquadramento

O presente projecto enquadra-se na área de recuperação de informação, mais precisamente no seu contexto empresarial, geralmente designado por “Enterprise Search”. Este modelo de

pesquisa distingue-se, em vários aspectos, dos mais vulgares modelos de pesquisa que são a “*Web Search*” e “*Desktop Search*”. Apesar de distintos, para os utilizadores finais todos eles parecem idênticos, pois têm a mesma funcionalidade: introduzir uma frase de pesquisa e obter os resultados [3]. Então, as diferenças destes modelos encontram-se nos “bastidores” de toda a acção, sendo apresentados já de seguida as principais diferenças entre eles.

A “*Enterprise Search*” utiliza uma ou mais fontes de informação, denominados repositórios, onde a informação guardada pode estar em diversos formatos. Outro aspecto crítico é a segurança da informação, isso porque ao nível empresarial, existe informação que só pode ser acedida por alguns funcionários ou grupos de funcionários (papéis). Assim, os resultados da pesquisa podem apenas apresentar resultados dos quais o utilizador, previamente autenticado, tenha privilégios para aceder. Estes dois aspectos, segurança e lidar com vários formatos de informação, são cruciais para que a plataforma de “*Enterprise Search*” obtenha sucesso [3].

No “*Desktop Search*”, a pesquisa é muito mais focada, procurando o utilizador algo que à partida sabe que vai encontrar e sendo a estrutura da informação conhecida e bem definida [3]. Em termos de segurança esta é definida localmente pelo sistema operativo.

Actualmente, talvez o modelo mais banal seja o “*Web Search*”, isto porque com o aumento da informação disponibilizada na Internet foi necessário criar mecanismos para encontrar a informação que procuramos. Ao aceder à Internet, o mais comum é utilizar um motor de busca onde introduzimos uma frase ou conjunto de termos sobre o tema que pretendemos encontrar. Neste tipo de pesquisa não sabemos se vamos encontrar aquilo que procuramos, nem existe problema algum a nível de segurança, pois todos os conteúdos são públicos.

A empresa PT-SI desenvolveu uma ferramenta de *Knowledge Management* baseada em MOSS (*Microsoft Office SharePoint Server*), de forma a poder endereçar a necessidade de disponibilização estruturada de conteúdos não estruturados. Após essa etapa, sentiu a necessidade de alargar as funcionalidades existentes, de modo a enriquecer o produto e adaptá-lo constante evolução do mercado. Um dos requisitos que é fundamental para a evolução do produto, é dotar esta ferramenta de uma potente pesquisa corporativa, que permita indexar e apresentar conteúdos relevantes a partir de outros repositórios empresariais [5].

1.1.1 Empresa PT-SI

A PT-SI, criada a 22 de Dezembro de 1999, é a empresa do Grupo Portugal Telecom responsável pelo fornecimento de Soluções de Tecnologias e Sistemas de Informação ao mercado empresarial [6].

Beneficiando da vasta experiência e *know-how* do Grupo PT, procura contribuir para a competitividade dos seus Clientes, criando valor através da integração de Tecnologias e Sistemas de Informação com Telecomunicações, tendo sempre em conta as melhores práticas existentes no mercado, a inovação e assegurando sempre a melhoria contínua e a natural evolução tecnológica [6].

A presença do Grupo PT no mercado das telecomunicações em África, na América do Sul e na Ásia funcionou como causa próxima para a internacionalização da PT-SI, que veio assim complementar as operações do Grupo. Para além destas presenças internacionais, a PT-SI tem desenvolvido projectos de SI em Empresas europeias de referência, tais como a Renault, Fnac e Artsana [6].

As parcerias estabelecidas com intervenientes reconhecidos pelo mercado, entre os quais se destacam a Microsoft, SAP, Oracle e Primavera, entre outros, representam uma grande mais-valia da PT-SI [6].

1.2 Motivação e Objectivos

A pesquisa empresarial (*Enterprise Search*) foi um tópico investigado internamente pela PT-SI durante aproximadamente um ano. Dado que existem diversos sistemas que suportam as

actividades da PT-SI e possuem grande quantidade de informação em diversos formatos, tornam-se visíveis as mais-valias que uma plataforma deste tipo pode trazer para a empresa [3].

Este projecto tem assim como objectivo integrar numa ferramenta ou desenvolver uma ferramenta que disponibilize funcionalidades de pesquisa avançada, baseada nas últimas tecnologias de pesquisa existentes no mercado. É necessário investigar os motores de pesquisa corporativa existentes no mercado, verificar as suas mais-valias e desvantagens face ao contexto applicacional existente, testar a adequabilidade técnica dos mesmos assim como avaliar o esforço de implementação associado, assim como o de manutenção. Finalmente, é necessária a implementação em ambiente produtivo do motor de pesquisa seleccionado, assim como garantir a eficiente indexação dos conteúdos escolhidos, tal como assegurar que os resultados produzidos são os relevantes para cada utilizador, dependendo do papel que desempenha [5]. É de acrescentar também o facto da necessidade de segurança sobre a informação ser crítico, devendo cada utilizador apenas poder aceder a informação à qual está autorizado a visualizar.

1.3 Estrutura da Dissertação

Para além da introdução, esta dissertação é composta por mais quatro capítulos. O capítulo 2 é composto por uma breve introdução à recuperação de informação e o estado da arte ao nível da área de pesquisa empresarial. No capítulo 3 é apresentada a análise do problema, planeamento e a razão da implementação da solução.

O quarto capítulo, intitulado Desenvolvimento, apresenta o trabalho efectuado a nível prático e toda a informação técnica necessária à compreensão da solução implementada.

No capítulo 5 é feita uma avaliação de todo o trabalho realizado e apresentadas as conclusões necessárias. São também referidos alguns pontos de melhoramento para um futuro trabalho que possa ser feito.

2 Pesquisa Empresarial

A pesquisa empresarial tem as suas raízes na área de recuperação e acesso à informação, que é uma disciplina que dá enfoque ao desenvolvimento de *frameworks* para representar documentos, pesquisa e as suas relações. É sem dúvida alguma uma ferramenta que pode trazer um grande valor para uma empresa, pois permite que os seus utilizadores encontrem mais rapidamente o que necessitam. Ajuda as empresas na tomada de decisões e transforma a forma como estas tratam os seus processos de negócio [7].

As tecnologias de pesquisa empresarial mais actuais permitem: a exploração e descoberta de informação, pesquisa sobre informação estruturada e desestruturada [7], controlo de acesso à informação, análise semântica das frases de pesquisas, mecanismos avançados de classificação dos resultados, e pesquisa sobre conteúdos áudio e vídeo, só para referir algumas das facilidades habitualmente disponíveis.

Neste capítulo, serão abordados alguns conceitos básicos sobre recuperação de informação e pesquisa empresarial, e feita uma análise das principais funcionalidades que uma plataforma deste tipo deve ter.

2.1 Introdução à Recuperação de Informação

Esta secção tem como objectivo introduzir alguns conceitos sobre recuperação de informação e pesquisa empresarial, para que posteriormente alguns aspectos sejam mais facilmente compreendidos.

2.1.1 Indexação

A utilização de um índice é a forma encontrada para evitar percorrer linearmente todos os documentos à procura da palavra que pretendemos encontrar. Apesar do grande poder dos computadores actuais, isto seria ainda muito custoso computacionalmente [2]. Um exemplo muito simplificado de um índice poderá então ser o que se descreve.

Supondo que temos os seguintes documentos:

Documento 1: Aquele computador é maior

Documento 2: O computador avariou

Documento 3: Esta mala de computador é maior

Poder-se-ia construir o índice mostrado na Tabela 2.1.

Tabela 2.1 - Exemplo de índice

Termo	<i>Documento</i>
Aquele	<i>Documento 1</i>
computador	<i>Documento 1, Documento 2, Documento 3</i>
é	<i>Documento 1, Documento 3</i>
maior	<i>Documento 1, Documento 3</i>
O	<i>Documento 2</i>
avariou	<i>Documento 2</i>
Esta	<i>Documento 3</i>
mala	<i>Documento 3</i>
de	<i>Documento 3</i>

Observando agora a tabela, podemos constatar a vantagem de ter um índice. Se pesquisarmos pelo termo “mala”, basta ir à coluna de termos, encontrar o termo “mala” e ver que está no *Documento 3*. Evitamos assim, ir ao documento 1 e 2 procurar o termo “mala”.

2.1.2 Relevância

A relevância num sistema de RI é a forma de ordenar os resultados encontrados, pois quase sempre existe mais do que um resultado que pode satisfazer as nossas necessidades de informação. Nos primeiros sistemas de RI que utilizavam um tipo de pesquisa booleana, que só verificavam a existência ou não do termo, normalmente os documentos eram ordenados por outros campos, como por exemplo a data. Ou seja, os documentos mais recentes eram mais importantes, o que podia não ser verdade.

Assim, sentiu-se a necessidade de criar modelos que permitissem atribuir relevo a cada um dos documentos devolvidos como resultado da nossa pesquisa.

Existem vários modelos actualmente, alguns dos quais usam modernos conceitos de Inteligência Artificial. No entanto, um dos mais utilizados é o modelo Espaço Vectorial, que é também dos mais simples de compreender e explicar.

Ainda dentro do aspecto da relevância existem dois aspectos que também são importantes, que são a Precisão e o *Recall*. A Precisão entende-se como: “Que fracção dos resultados devolvidos são relevantes para satisfazer as necessidades de informação?” [2].

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

Figura 2.1 - Fórmula matemática para o conceito de precisão [8]

O *Recall* entende-se como: “Que fracção dos documentos relevantes na colecção são devolvidos pelo sistema?” [2].

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

Figura 2.2 - Fórmula matemática para o conceito de *recall* [8]

2.1.2.1 Modelo Espaço Vectorial

Neste modelo todo o texto é representado como um vector de termos. O modelo não define o que são termos, mas normalmente são palavras ou frases. Considerando que os nossos termos são palavras, então todas as palavras num vocabulário são transformadas numa dimensão de enorme e multidimensional espaço vectorial. Todo o texto pode ser assim representado por um vector. Se um termo pertence a um texto, então este tem um valor (não nulo) no vector texto ao longo da dimensão correspondente ao termo [1].

Todos os textos têm um número limitado e variado de termos, que serão o nosso vocabulário. Podemos então imaginar que um texto representado no nosso modelo vai ser um conjunto muito disperso de vectores. Os sistemas baseados neste modelo normalmente utilizam apenas o quadrante positivo do vector espaço, ou seja, não existem termos com valor negativo [1].

Para atribuir um valor de relevância a um documento para um determinado texto de pesquisa, o modelo mede a semelhança entre o vector do texto de pesquisa (como é texto também, pode ser convertido para um vector) e o vector do documento [1]. A forma como se calcula a similaridade não está definida no modelo mas tipicamente é usado o ângulo formado entre os dois vectores. Assim, quanto maior o ângulo, menor é a semelhança entre o texto de pesquisa e o documento. Se em vez de usarmos o ângulo, usarmos o co-seno do ângulo, podemos dizer que o resultado da operação é o grau de semelhança do texto da pesquisa com o texto do documento. Além disso o co-seno tem a propriedade de ser 1 em vectores idênticos e 0 em vectores ortogonais, logo o nosso peso de relevância vai variar de 0 a 1, sendo agora possível ordenar os resultados devolvidos [1].

Um exemplo com dois documentos, em que cada um deles tem no máximo três termos (para poder ser representado graficamente) está representado na Figura 2.3. [1].

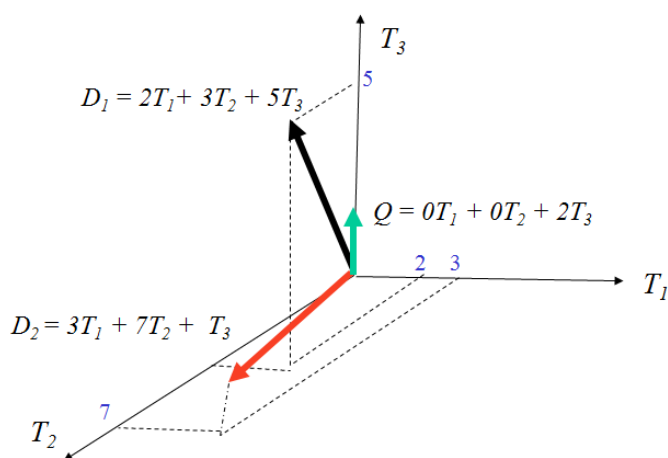


Figura 2.3 - Exemplo do Modelo Espaço Vectorial [9]

Não sendo relevante para este documento detalhar todos os modelos existentes, é importante saber quais os modelos que existem apresentando-se de seguida uma tabela com os vários modelos existentes.

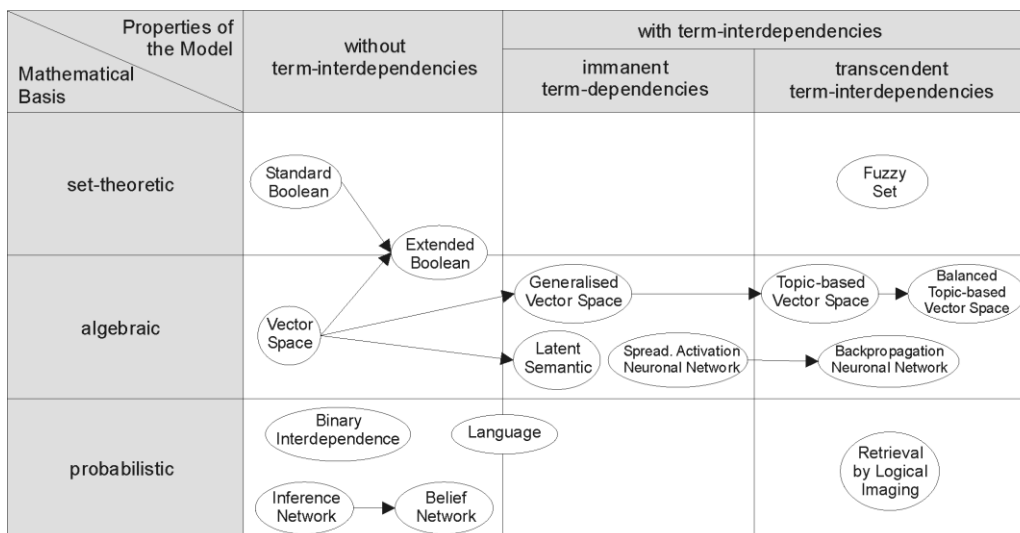


Figura 2.4 - Modelos de relevância [8]

2.1.3 Processamento dos documentos e do texto de pesquisa

Esta secção apresenta algumas técnicas de processamento de textos em Recuperação de Informação.

2.1.3.1 *Tokenization*

A *tokenization* é utilizada para decompor um documento em cada termo (palavra) que o compõe. Os delimitadores utilizados para a *tokenization* geralmente são: o espaço em branco entre os termos, quebras de linhas, tabulações, e alguns caracteres especiais [10].

2.1.3.2 *Stemming*

Numa pesquisa o utilizador, frequentemente, especifica uma palavra que não existe no documento que pretende encontrar, mas existe uma variante desta palavra que é apresentada nesse documento. Por exemplo: formas plurais, formas de gerúndio e sufixos de tempo verbal passado. Estas variações sintáticas que impedem uma perfeita combinação entre uma palavra da pesquisa e uma palavra do respectivo documento. Devido a este facto, é utilizado o *stemming* que consiste no processo de redução de um termo ao seu radical (base), removendo as terminações, afixos, e vogais temáticas. Com sua utilização, os termos derivados de um mesmo radical serão contabilizados como um único termo [11].

2.1.3.3 *Lemmatization*

A *Lemmatization* é uma especialização (melhoramento) do *Stemming*, em que o processo reduz o termo para o seu radical mas tem em conta o contexto em que este se encontra. Este processo é muito mais complexo que o *Stemming* [12].

2.1.3.4 Correção ortográfica

A correção ortográfica pode ser aplicada antes de ser efectuada a pesquisa e isto permite que não sejam feitas pesquisas com palavras mal escritas, que não vão devolver resultados. Ou então, pode ser feita após a pesquisa como forma de sugestão ao utilizador e este decide ou não aceitar a sugestão [13].

A primeira solução evita que o sistema despenda recursos e devolva um resultado vazio. Mas existe a desvantagem de a correção automática não ser a correcta e o utilizador não encontre os resultados que pretende.

A segunda solução obriga sempre a despendar os recursos à procura dos resultados, mas evita a situação de devolver resultados que não são os pretendidos pelo utilizador.

2.2 Estado da arte

O volume de informação das grandes organizações duplica de ano para ano. Armazenada de forma dispersa pela organização, torna o seu acesso mais difícil, o que pode pôr em causa a capacidade da organização responder rapidamente ao mercado e a mudanças do meio [14].

As mais avançadas plataformas de pesquisa empresarial existentes no mercado, permitem à organização, e aos seus utilizadores internos ou externos, obter respostas relevantes, precisas, seguras e de forma rápida às pesquisas efectuadas. A Plataforma concentra, num único ponto de acesso, toda a informação estruturada e desestruturada, permitindo às organizações tomarem melhores decisões, melhorar os seus processos de negócio e aumentar os seus lucros [14].

Nesta secção são introduzidas os principais requisitos e funcionalidades que as mais avançadas plataformas de pesquisa empresarial devem possuir.

Uma vez que as plataformas FAST ESP e a Autonomy IDOL são consideradas as mais avançadas em PE (Pesquisa Empresarial) [15] [7], serão introduzidos alguns dados referentes às mesmas.

2.2.1 Disponibilidade

Uma plataforma deste tipo constitui um portal de acesso a toda a informação contida na organização, logo um aspecto importante é a sua disponibilidade. Os utilizadores querem em todos os momentos poder aceder à informação. A plataforma deve assim permitir mecanismos de redundância para ter uma elevada disponibilidade. Estes mecanismos estão também dependentes da arquitectura utilizada.

2.2.1.1 FAST ESP

A Plataforma FAST ESP foi desenhada para ser tolerante a falhas, sendo esta tolerância efectuada em diferentes níveis, que podem ser: a aquisição, a indexação e a pesquisa [14].

2.2.1.2 Autonomy IDOL

A plataforma Autonomy Idol suporta tolerância a falhas. Uma das formas de a atingir é através de replicação local. Esta técnica, denominada de “*Mirroring*”, cria réplicas em servidores primários e secundários, para que em caso de falha a informação esteja sempre disponível [15].

A outra forma consiste em efectuar réplicas remotas. Encontrando-se dispersas geograficamente, as organizações devem utilizar a informação local sempre que possível. As cópias remotas desta informação só devem ser utilizadas caso o sistema local falhe [15].

2.2.2 Escalabilidade e Performance

As plataformas mais avançadas utilizam uma arquitectura distribuída de forma a poderem crescer, pois a informação contida no seio das organizações não é estática, e está cada vez a aumentar mais o seu volume. Assim as plataformas que permitem a pesquisa sobre essa informação têm que se adaptar e serem capazes de suportar toda essa informação.

2.2.2.1 FAST ESP

A plataforma FAST ESP no contexto actual, é uma plataforma de pesquisa muitíssimo eficiente e escalável. Isso pode ser constatado através das estatísticas apresentadas na Tabela 2.2.

Tabela 2.2 - Estatísticas FAST ESP [14]

Maior <i>dataset</i>	40 <i>petabytes</i>
Número máximo de documentos	10 biliões de documentos
Maior débito de pesquisas simultâneas	2,000 pesquisas/segundo
Maior compressão	250M documentos/máquina
Maior débito de actualização	500 actualizações /segundo
Melhor análise analítica	200 parâmetros em informação em tempo real

Esta foi desenhada para correr em *hardware* de baixo custo e num ambiente do tipo *grid*. Se o ritmo de pesquisas for muito elevado, o sistema é capaz de as distribuir por várias linhas dos nós de pesquisa. Se por sua vez, o índice cresce demasiado, este pode ser particionado em diversas colunas dos nós de pesquisas.

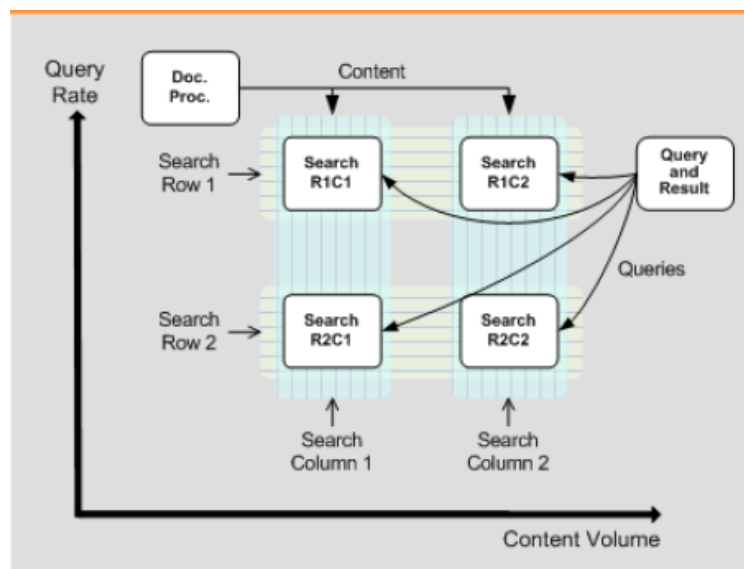


Figura 2.5 - Matriz de nós de pesquisa [16]

2.2.2.2 Autonomy IDOL

Na plataforma Autonomy IDOL, a performance e capacidade podem ser duplicadas bastando apenas replicar uma instalação existente, pois a escalabilidade é linear. Isso permite que se façam facilmente previsões de crescimento, sem haver preocupações com *bottlenecks*. Como os dados se encontram replicados por vários servidores pode haver um balanceamento de pedidos por parte dos utilizadores, garantindo assim um excelente desempenho e redução de latência [17].

	 Platform Specification	 Supporting
Active Service	4x Intel/LINUX 2x 3GHz CPU 4GB RAM	>1,000,000 queries per day 20,000,000+ documents >500,000 users
News Content Portal	Sun v440 4x 1.6GHz ultraSPARC 111i 4GB RAM	5,000,000 news articles 55+ queries per second <.01 sec response time 200 GB/Hour index rate
News Service	IBM AIX pSeries Server 4x CPU 4GB RAM AIX 5.2	1,200,000 documents 200 queries per second 60 concurrent queries <0.5 sec response time 26 GB/Hour index rate
Internet News Portal	Sun v440 4x 1.6GHz ultraSPARC 111i 4GB RAM	2.4 million node taxonomy 200,000 daily churn
Corporate Portal	Win 2003 Server 4x 3GHz CPU 8 GB RAM	2,800,000 documents 700 GB/Hour index rate 180 queries per second
Web-facing Portal	Intel/LINUX 2x 3GHz CPU 4GB RAM	1,500,000+ documents 70 GB/hour index rate 100 queries per second
Email compliance and record system	Win 2003 Server 4x 3GHz CPU 4 GB RAM	20 billion mail objects 56 million per day 2.3 million per hour 650 per second
Web-facing Portal	Linux 64 AMD Opteron 2 x CPU	170+ million queries per day 2,000 query's per second
Government Information System	Linux 64 AMD Opteron 4 x CPU	1 billion documents 300+ queries per second

Figura 2.6 - Exemplos de *benchmarks* [18]

2.2.3 Transparência

Uma plataforma de ES tem de estar preparada para a globalização e até mesmo para a heterogeneidade de sistemas que compõem as organizações modernas. Deve suportar vários sistemas operativos, como são exemplo o Windows, o Red Hat Enterprise Linux, o Suse Linux Enterprise, o Solaris, etc.

Suportar diferentes idiomas (e várias codificações para cada um deles) para pesquisa e ordenação. Suportar também mecanismos avançados de linguística.

Deve suportar várias linguagens de programação e tecnologias, para permitir a sua integração e desenvolvimento. Para a aquisição de conteúdos deve possuir uma vasta biblioteca de conectores.

2.2.3.1 FAST ESP

Além de poder ser instalada em diversos sistemas operativos e suportar vários idiomas, a Fast ESP possui os conectores apresentados na Tabela 2.3 e permite a criação de novos conectores.

Tabela 2.3 - Conectores FAST ESP [14]

FAST Web Crawler	Permite aceder a sítios públicos ou protegidos
File Traverser	Permite aceder ao sistema de ficheiros local
XML content	Permite a indexação de ficheiros XML no seu formato nativo
Conectores	FAST disponibiliza uma vasta gama de conectores que inclui:
Oracle	MS Sharepoint 2003
JDBC	MS Sharepoint 2007
Lotus Notes	MS CMS
Documentum	MS Exchange
Meridio	StarTeam
CaliberRM	
Os parceiros FAST disponibilizam também outros conectores tais como:	
Kapow	OpenText Live Link
Vignette	InterwovenTeamSite
FileNet	Communique
SAP EP	SAP PLM
BEA Weblogic	IBM WebSphere

Possui ainda componentes para o reconhecimento de imagens, vídeo e áudio.

2.2.3.2 Autonomy IDOL

A plataforma Autonomy IDOL permite o acesso a todas as fontes de informação (através de conectores), onde normalmente 80% da informação dentro de uma empresa é desestruturada, consistindo em texto, áudio e vídeo [15].

É independente do idioma, pois as organizações actuais podem estar dispersas por todo o mundo e a conduzir os seus negócios utilizando diversos idiomas [15].

Pode, além disso, ser instalada em diversos sistemas operativos [15].

2.2.4 Segurança

A segurança da informação é um factor determinante para algumas organizações. Tendo em conta isso, uma plataforma ES deve oferecer um sistema de segurança rigoroso e robusto, que permita proteger o conteúdo presente na organização. Deve permitir aplicar medidas de segurança que permitam proteger os servidores, o conteúdo alojado nos servidores e ainda segurança a nível das transacções de aquisição de conteúdos. Os utilizadores finais apenas podem ver e aceder a conteúdo a que possuem privilégios de acesso.

2.2.4.1 FAST ESP

A plataforma Fast ESP fornece segurança através do Security Access Module (SAM), que garante a segurança através da filtragem de resultados, recorrendo a duas técnicas:

- *Per user search security filter* – Quando filtra durante a pesquisa do utilizador, o SAM intersecta a pesquisa do utilizador com um filtro de segurança para limitar o conjunto de resultados. Isto é, os documentos indexados possuem um campo com um filtro gerado quando foi feita a indexação e é intersectado com um filtro gerado para cada utilizador.
- *Post query result removal* – Quando filtra após a pesquisa ser efectuada, o SAM “questiona” a fonte de cada resultado para determinar se o utilizador tem permissão para ver ou não o resultado.

2.2.4.2 Autonomy IDOL

Na plataforma Autonomy IDOL são disponibilizados três tipos de modelos de segurança: a não mapeada, em cache e a mapeada.

A não mapeada é o método mais comum, usado pelos repositórios fonte e os motores de pesquisa. Por cada potencial resultado para uma dada pesquisa são feitos pedidos ao repositório fonte para decidir quais os privilégios de acesso a esse resultado (documento). Este método pode ser bastante problemático a nível de performance e escalabilidade, visto que por cada pesquisa existe uma sobrecarga de pedidos para definir os privilégios de acesso aos repositórios.

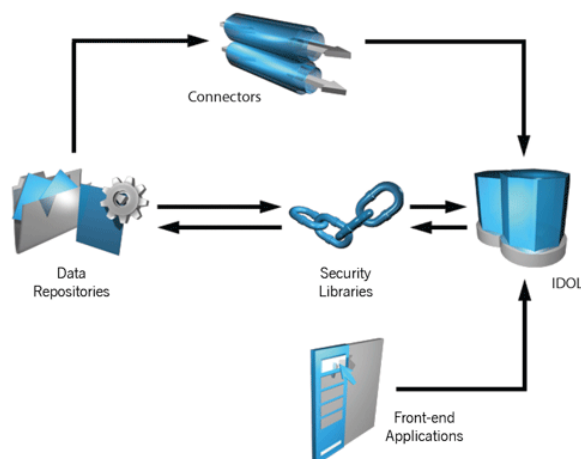


Figura 2.7 - Segurança não mapeada [15]

A segurança em cache é um dos métodos eleitos para sistemas legados. É uma evolução do método anterior, que vai guardando os privilégios dos resultados de pesquisas já efectuadas. Desta forma, caso a pesquisa já tenha sido efectuada anteriormente, não é necessário “perguntar” quais os privilégios ao repositório fonte, pois estes já estão em *cache*.

O método mais eficiente é o de segurança mapeada, que ao indexar os conteúdos dos repositórios anexa um campo extra com a informação referente aos privilégios de acesso. Assim, não existe necessidade de contactar com o repositório para decidir os privilégios de acesso.

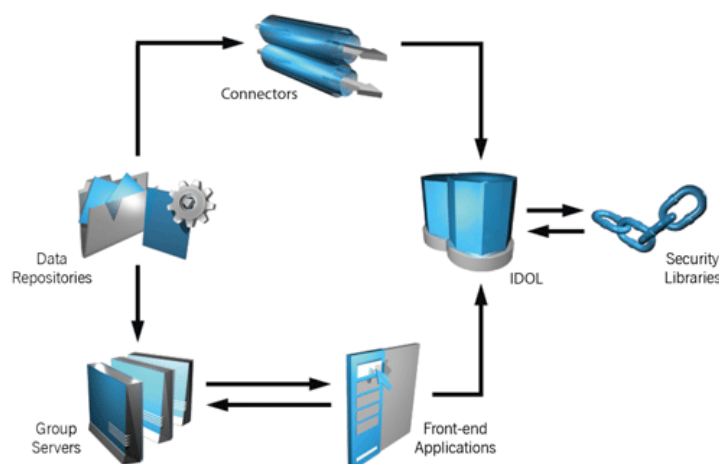


Figura 2.8 - Segurança mapeada [15]

Por último, como a arquitectura é modular existe necessidade de comunicação entre os vários subsistemas, muitas vezes sobre redes inseguras. Para impedir quebras de confidencialidade o IDOL suporta comunicações cifradas utilizando SSL.

2.2.5 Análise da Pesquisa

Ao introduzir o texto na caixa de pesquisa o utilizador pode cometer erros ortográficos. Estes devem ser corrigidos de forma automática para que a pesquisa devolva resultados relevantes. Outro caso é o dos sinónimos, em que por vezes o documento que procuramos tem um termo que é sinónimo do que procuramos, e este documento deve ser devolvido na lista de

resultados. Logo, a plataforma de pesquisa tem de tratar este tipo de problemas. O género, o número e a conjugação verbal podem também influenciar os resultados, por isso a plataforma tem que ter isso em atenção.

2.2.5.1 FAST ESP

O FAST ESP integra dois tipos de análise num único ambiente: a quantitativa e a qualitativa. A sua arquitectura foi desenhada de forma a possibilitar que motores de *data mining* externos possam ser também integrados [14].

Para a análise qualitativa, são suportadas vários mecanismos linguísticos [14], como os que se apresentam na Tabela 2.4.

Tabela 2.4 - Mecanismos da análise qualitativa [14]

Verificação ortográfica	Detecta e corrige palavras e frases escritas incorrectamente
<i>Lemmatization</i>	Aplica inflexões gramaticais regulares ou irregulares
Detecção de frases	Detecta nomes e frases e ainda pode automaticamente reescrever a pesquisa ou fornecer uma dica para a pesquisa
Pesquisa Fonética	Permite devolver resultados baseados em similaridades fonéticas
<i>Anti-Phrasing</i>	Remove todas as palavras que não tem relevo para o significado da pesquisa
Linguagem natural	Reconhece construções de frase do tipo “Quem?”, “O quê?”, “Onde?”, “Quando?” e “Porquê?”
Sinónimos	Dicionário de sinónimos personalizado que define termos específicos ao negócio

A análise quantitativa, ou *data mining*, lida puramente com números, mais especificamente com as suas distribuições estatísticas. A plataforma suporta cálculos básicos como a média e total, mas os cálculos mais complexos são feitos por motores analíticos e bibliotecas matemáticas como a SPSS e SAS, permitindo a arquitectura da plataforma integrar outros motores externos [14].

2.2.5.2 Autonomy IDOL

A plataforma possui duas tecnologias inovadoras. A Automatic Query Guidance (AQG) e a Real-Time Predictive Query (RPQ) [15].

As pesquisas feitas através de uma a três palavras-chave muitas vezes falham porque não disponibilizam contexto suficiente. A funcionalidade AQG disponibiliza automaticamente sugestões orientadas às palavras-chave, para que os utilizadores encontrem os documentos mais relevantes para a sua pesquisa. Por exemplo: ao pesquisar pela palavra ‘Madonna’, o utilizador

será questionado sobre qual o contexto associado: Madonna, a cantora; Madonna, o ícone religioso; ou algo mais obscuro como Madonna, uma lavandaria na Baixa-Chiado [19].

A RPQ permite mostrar resultados à medida que o utilizador introduz caracteres na caixa de texto de pesquisa. Desta forma os utilizadores já não necessitam pressionar o botão de pesquisa para que os resultados sejam apresentados [19].

2.2.6 Pesquisa Semântica

Uma plataforma de Pesquisa Empresarial moderna deve disponibilizar os métodos de pesquisa legados em conjunto com os mais recentes, permitindo ao utilizador escolher aquele que prefere num dado momento. Nos métodos legados, deve suportar a pesquisa com palavras-chave utilizando operadores, como são exemplo o: AND, OR, NOT, NEAR, DNEAR, SOUNDEX, FUZZY e RANGE. Suportar também, restrições booleanas sobre múltiplos campos de metadados [19].

Mas se imaginamos uma pesquisa deste tipo: “Que empresas trabalham com o iPod”. Os motores de busca convencionais, no mínimo, devolviam todos os documentos que incluíssem os termos “empresa” e “iPod”. Mas numa plataforma de última geração espera-se que o resultado devolvido seja uma lista dos nomes das empresas que trabalham com o iPod e para cada uma delas a frase que identifica a relação. Este tipo de funcionalidade introduz a possibilidade de colocar questões em linguagem natural ao motor de busca, através de análise semântica da frase colocada [14].

2.2.6.1 FAST ESP

A plataforma FAST ESP permite pesquisa semântica. É criado um índice semântico para cada escopo, que pode ser um documento, um registo da base de dados, uma página Web ou até mesmo uma frase ou parágrafo. Estes escopos em conjunto com todas as entidades encontradas e extraídas automaticamente, permitem as respostas a perguntas em linguagem natural. As entidades podem ser por exemplo: nomes de pessoas, número de telefone, localizações geográficas e nomes de empresas. A FAST ESP suporta aproximadamente 40 tipos de entidades e de futuro pretende suportar ainda mais. É ainda possível criar as nossas próprias entidades através de uma interface aberta. Em linguagem natural são suportadas perguntas do tipo “Quem?”, “O quê?”, “Onde?”, “Quando?” e “Porquê?” [14].

São suportados os operadores AND, OR, NOT, NEAR/ONEAR, para citar alguns [20].

2.2.6.2 Autonomy IDOL

O Autonomy IDOL suporta os operadores referidos anteriormente e suporta também restrições booleanas sobre múltiplos campos de metadados [19].

No campo da pesquisa conceptual a solução IDOL possibilita elevados níveis de precisão e sofisticação, fazendo uso de uma tecnologia escalável que reconhece conceitos. Como consegue dar contexto à informação, a plataforma permite uma potente capacidade de recuperação de informação conceptual, que inclui: recuperação por linguagem natural, em que aceita frases, parágrafos, ou até mesmo excertos de discursos em formato áudio e devolve documentos conceptualmente relacionados; suporta pesquisas por exemplo e permite refinar os resultados através da selecção de exemplos que representam o contexto pretendido [19].

2.2.7 Relevância

Não chega devolver resultados, os resultados têm de ser relevantes para quem os pesquisou. O ideal é o resultado que pretendemos ser o primeiro da lista de resultados, mas por vezes existem outros resultados que também podem ser relevantes, logo tem que haver um mecanismo para ordenar os resultados e colocar os mais relevantes no topo da lista.

2.2.7.1 FAST ESP

O FAST ESP possui um modelo de relevância único, adaptável, completamente transparente e multi-dimensional, que mede e classifica em termos de relevância. O modelo é baseado em métricas, que podem ser afinadas [14]. Algumas delas estão definidas na Tabela 2.5.

Tabela 2.5 – Descrição das métricas de classificação de resultado [14]

Frescura	O quão fresco é o documento comparado com a idade da pesquisa?
Autoridade	O documento é considerado uma autoridade para esta pesquisa?
Plenitude	O quão bem, a pesquisa corresponde a um contexto superior (titulo, URL)?
Estatísticas	O quão bem, o conteúdo do documento corresponde à pesquisa na sua globalidade?
Qualidade	Qual é a qualidade do documento? O quão importante é ele?
Geo-espacial	O quão longe, o resultado está de uma referência geográfica?

2.2.7.2 Autonomy IDOL

O Autonomy IDOL utiliza complexos algoritmos baseados numa combinação de métodos da teoria de informação Bayesiana para automaticamente classificar a relevância de cada documento. Na prática a relevância pode ser vista como a medida conceptual do quão bem o texto de pesquisa se identifica com o documento em questão [19].

Os administradores da plataforma podem ajustar os mecanismos de relevância. Podem empregar mecanismos de *bias*¹ ou multiplicadores para impulsionar a relevância de alguns documentos para determinadas pesquisas [19].

Esta capacidade, melhora a experiencia do utilizador na medida em que este encontra resultados que melhor satisfazem as suas expectativas. Os administradores podem criar perfis diferentes para diferentes utilizadores e papéis [19].

¹ Bias é um termo usado em estatística para expressar o erro sistemático ou tendenciosidade. No contexto empregue refere-se ao facto de podermos afectar a relevância dos resultados de forma tendenciosa [36]. Por exemplo, mostrar primeiro os resultados mais acedidos pelos utilizadores. O que não garante que os resultados são mesmo relevantes.

2.2.8 Refinação de resultados

Apesar do utilizador saber sempre o que quer encontrar, por vezes não sabe a melhor forma de o pesquisar. Então, procura por algo mais abrangente, e depois tem de andar a ver uma grande quantidade de resultados até encontrar o que quer. Para evitar o transtorno, as plataformas de pesquisa modernas incorporam mecanismos que permitem refinar os resultados iterativamente. Por exemplo, num site de uma loja de informática procuramos por “computador”. Como resultados são devolvidos todos os computadores disponíveis para venda na loja. A refinação da pesquisa é feita através de campos relacionados com computador, como por exemplo: de secretária ou portátil. Ao indicar 'portátil' aparecem só os computadores portáteis, e depois podemos refinar a pesquisa outra vez, através de outro campo.

2.2.8.1 FAST ESP

A plataforma FAST ESP suporta três tipos de navegação [14]:

- Navegação taxonómica – que utiliza conceitos e categorias de uma taxonomia como navegadores.
- Busca facetada – que expõe os campos da base de dados como navegadores. Os navegadores são os possíveis valores, como definidos pelos resultados de pesquisas anteriores.
- Descoberta de entidades – que extrai entidades textuais dos resultados de pesquisas anteriores. Ao contrário das taxonomias e facetas, a extracção das entidades retira os seus navegadores directamente dos resultados.

2.2.8.2 Autonomy IDOL

Por sua vez, o Autonomy IDOL permite a navegação em tempo real, sobre diversas taxonomias, não requerendo a intervenção humana, tornando a navegação dos resultados numa ferramenta extremamente poderosa [19].

Permite a integração de informação estruturada com informação desestruturada e melhora a experiência do utilizador final da plataforma, ordenando os resultados de uma forma mais intuitiva. Por exemplo, a o pesquisar por “vinho”, cria navegadores automaticamente, que podem por exemplo ser a cor ou região onde o vinho foi produzido [19].

3 Análise e Planeamento

Como podemos constatar no capítulo anterior, as plataformas de Pesquisa Empresarial têm progredido bastante nos últimos anos. Com todo este avanço, estas plataformas além de poderosas a desempenhar o seu papel, ganharam uma elevada complexidade quer para o seu lançamento quer para a sua manutenção. Com isto quero dizer, que este tipo de produto apesar de trazer algumas funcionalidades *Out Of The Box* (OOTB), as mais interessantes necessitam de algum planeamento e esforço para a sua implementação.

3.1 Objectivos

Este género de plataforma, para ser completo transporta consigo uma interface de pesquisa OOTB para que o utilizador possa tirar partido das suas funcionalidades. Além desta não ser apelativa e ostentar a imagem da empresa, o problema é que estas interfaces não se encontram integradas nos sistemas que os utilizadores da empresa normalmente usam. Esta interface está vocacionada apenas para a pesquisa. O que acontece na PTSI é que foi desenvolvido um *dashboard* personalizado para cada utilizador que pretende concentrar várias fontes de informação e funcionalidades, facilitando assim a vida dos seus colaboradores. Estes deixam de ter a necessidade de ir a diversos locais para cumprir as suas tarefas, encontrando tudo o que necessitam neste *dashboard*. Sendo assim, o que parece mais lógico é que o ponto de acesso para a pesquisa empresarial seja feito neste *dashboard*, e para que este seja possível, é necessário haver algum desenvolvimento. Este é um dos objectivos primordiais.

Como foi referido, o *dashboard* é para ser utilizado por vários utilizadores independentemente do cargo que ocupam. Mas será que pesquisando pela mesma palavra um administrador quer obter os mesmos resultados que um consultor? A resposta pode ser negativa. Isto porque, cada um tem as suas funções e tarefas, bem diferenciadas, e como tal, espera-se que os resultados devolvidos sejam os mais relevantes para cada um deles. Este é o segundo objectivo, calibrar e afinar os mecanismos de relevância, baseados em papéis, para que os utilizadores consigam tirar o melhor partido da ferramenta e com isso criar valor.

3.2 Análise das plataformas de pesquisa

O projecto de ES na PTSI, já se encontra a ser analisado há algum tempo, existindo já estudos sobre as plataformas de pesquisa preferenciais, que são a Autonomy IDOL, Fast ESP e

Google GSA. Para tomar a decisão de qual plataforma a utilizar foram tomados em consideração esses estudos já realizados.

A plataforma Fast ESP destacou-se das outras em termos de custos de licenciamento e pela questão de ser suportada pela Microsoft, que é parceira da PTSI. É de destacar também que a plataforma FAST vai ser utilizada no Sharepoint 2010, que certamente será utilizado pela PTSI, logo é mais um factor importante a tender para a utilização da FAST ESP.

3.2.1 Resumo dos estudos da PTSI

De forma muito resumida os estudos efectuados pela PTSI às principais plataformas de pesquisa empresarial, podem condensar-se nas secções seguintes.

3.2.1.1 Requisitos [21]

- **Repositórios de informação:**
 - MOSS 2007
 - Filenet P8
 - Intranet (*Web Crawl*)
 - EPM (MS EPM 2003)
- **Funcionalidades**
 - Pesquisa Segura
 - Apresentação de resultados em menos e 2 segundos
 - Resultados baseados em papéis
 - Mecanismo de sugestão de pesquisa
 - Agrupamento automático de resultados
- **Vectores de comparação entre as plataformas**
 - Mecanismos de relevância
 - Segurança
 - Performance
 - Custos de Licenciamento e Manutenção
 - Curva de aprendizagem
 - Capacidade de expansão da plataforma

3.2.1.2 Resultados [21]

De acordo com a análise efectuada foram obtidos os resultados apresentados graficamente e de forma resumida na Figura 3.1.

Na opinião da PTSI as plataformas FAST ESP e Autonomy IDOL são sem dúvida alguma mais maduras. Ambas as plataformas disponibilizam um conjunto razoável de funcionalidades OOTB e permitem com alguma facilidade a sua personalização e integração de novas funcionalidades. A Google GSA é uma plataforma muito menos personalizável, que apenas se destaca pela sua simplicidade, que pode ser interessante para cenários mais normalizados. A Tabela 3.1 apresenta a visão por parte da PTSI de cada uma das plataformas.

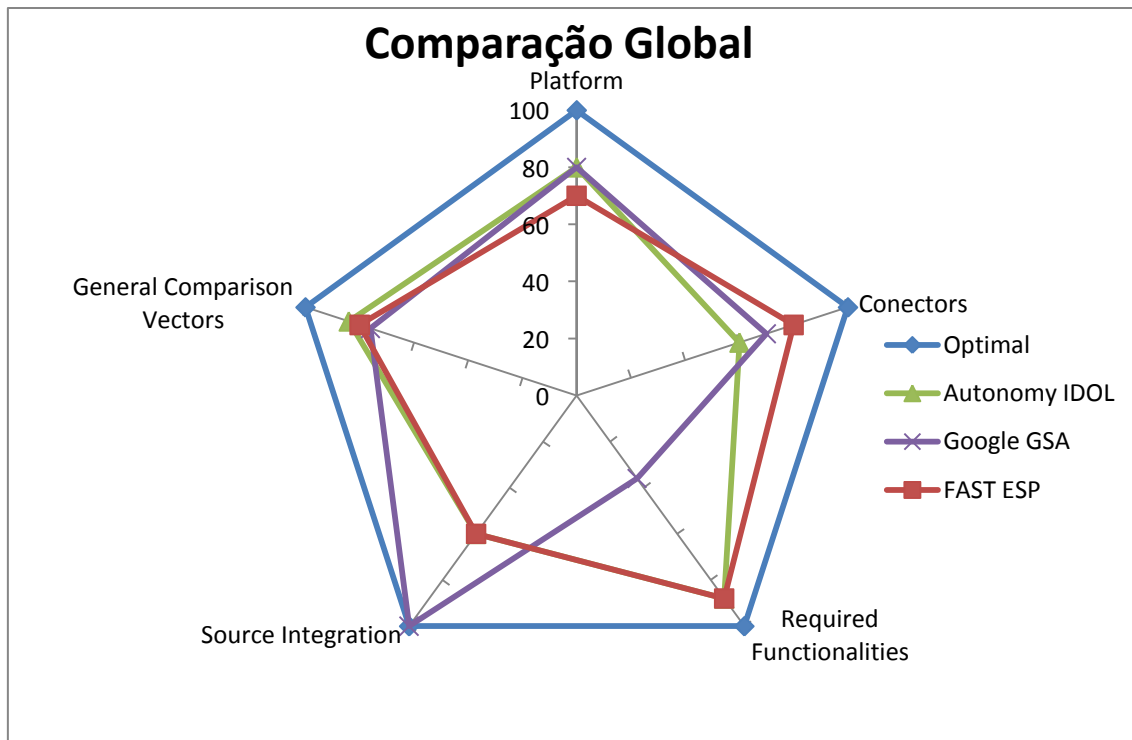


Figura 3.1 - Gráfico de comparação das plataformas de pesquisa empresarial [21]

Tabela 3.1 - Vantagens e desvantagens de cada uma das plataformas [21]

	FAST ESP	Google GSA	Autonomy IDOL
Pros	- Open architecture; highly customizable - Good number of OOTB features	- Almost “plug-and-play” solution - Easy to use	- Mathematical approach sets the difference - Good number of OOTB features
Cons	High learning curve (from a developer view)	- Low level of customization - Reduced number of OOTB features	High learning curve (from a developer view)

3.3 Especificação de requisitos

Esta secção tem como objectivo a definição de todos os requisitos do *front-end* de pesquisa a ser integrado no software SPOT.

3.3.1 Requisitos funcionais

Os requisitos funcionais foram identificados recorrendo ao modelo de casos de utilização. Assim, o modelo de apresentação dos casos de utilização, vai seguir uma forma tabular, com a seguinte informação:

- Identificador
- Nome
- Descrição
- Cenário

- Actores
- Prioridade
- Pré-condições
- Pós-condições
- Pressupostos
- Diagramas²
- Maquetas³

3.3.1.1 Visão geral

O diagrama de casos de utilização geral, a partir do qual foram especificados os requisitos funcionais da aplicação, pode ver-se na Figura 3.2.

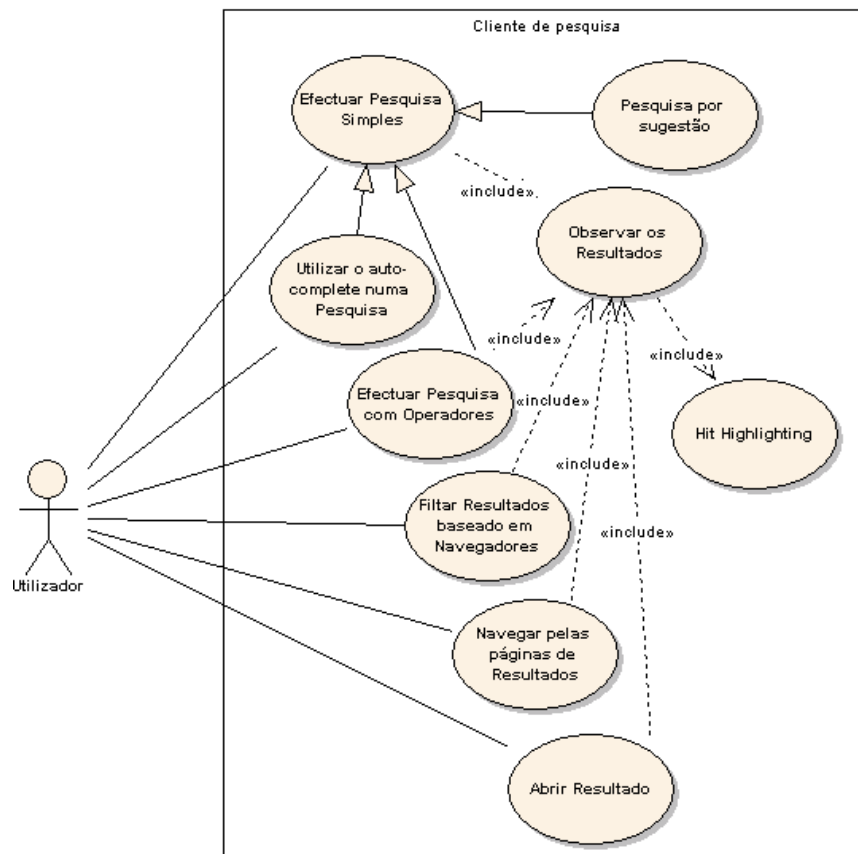


Figura 3.2 - Visão Geral dos Casos de Uso

3.3.1.2 Actores

De acordo com a análise feita, só existe um tipo de actor que será denominado Utilizador. Este utilizador por sua vez pode ter vários papéis, mas o seu papel não influencia o funcionamento do *front-end* de pesquisa. Influencia apenas a plataforma de pesquisa empresarial que suporta a lógica de negócio da pesquisa.

² Nos casos em que a utilização facilite a compreensão do requisito

³ Nos casos em que o requisito implique uma interface gráfica

3.3.1.3 Casos de Utilização

3.3.1.3.1 Efectuar Pesquisa Simples

Tabela 3.2 - Caso de utilização: Efectuar Pesquisa Simples

Identificador	CU01
Nome	Efectuar Pesquisa Simples
Descrição	Apresentar uma caixa de texto, onde um utilizador pode introduzir um conjunto de caracteres, que forma o texto da sua pesquisa. Deve incluir também um botão que executa a pesquisa.
Cenário	Após a introdução de texto e o premir do botão de pesquisa o utilizador irá ver os resultados referentes ao texto pesquisado caso existam.
Actores	Utilizador
Prioridade	Elevada
Pré-Condições	O utilizador tem que preencher a caixa de texto
Pós-Condições	Apresentar ao utilizador o conjunto de resultados de acordo com a pesquisa efectuada, o papel do utilizador e os seus privilégios de acesso aos recursos apresentados nos resultados
Pressupostos	O texto introduzido é válido



Figura 3.3 - Maqueta: Efectuar Pesquisa Simples

3.3.1.3.2 Utilizar auto completar numa Pesquisa

Tabela 3.3 - Caso de utilização: Utilizar auto completar numa pesquisa

Identificador	CU02
Nome	Utilizar auto completar numa pesquisa
Descrição	Na caixa de texto de uma pesquisa simples, à medida que o utilizador escreve, deve aparecer um conjunto de sugestões de pesquisa fornecidas pela plataforma de pesquisa empresarial. O utilizador pode seleccionar qualquer uma das sugestões e assim efectuar a pesquisa.
Cenário	Após a introdução de texto de algum texto na caixa de pesquisa, são apresentadas sugestões de pesquisa relacionadas com o texto introduzido.
Actores	Utilizador
Prioridade	Baixa
Pré-Condições	O utilizador tem que preencher a caixa de texto com um mínimo de dois caracteres
Pós-Condições	Apresentar ao utilizador o conjunto de resultados de acordo com a pesquisa efectuada, o papel do utilizador e os seus privilégios de acesso aos recursos

	apresentados nos resultados
Pressupostos	N/A



Figura 3.4 - Maqueta: Utilizar auto completar numa pesquisa

3.3.1.3.3 Efectuar Pesquisa com Operadores

Tabela 3.4 - Caso de utilização: Efectuar Pesquisa com Operadores

Identificador	CU03
Nome	Efectuar Pesquisa com Operadores
Descrição	Na caixa de texto de pesquisa o utilizador pode utilizar os operadores: AND, OR, NOT e ASPAS (""). Este último operador implica que todo o texto contido nele é pesquisado como um todo, ou seja uma frase.
Cenário	Na caixa de texto de pesquisa simples, o utilizador pode introduzir operadores previamente especificados, que permitem um melhor controlo sobre os resultados devolvidos.
Actores	Utilizador
Prioridade	Elevada
Pré-Condições	O utilizador tem que preencher a caixa de texto com texto para além dos operadores
Pós-Condições	Apresentar ao utilizador o conjunto de resultados de acordo com a pesquisa efectuada, o papel do utilizador e os seus privilégios de acesso aos recursos apresentados nos resultados
Pressupostos	Os operadores são colocados de forma correcta pelo utilizador

3.3.1.3.4 Filtrar Resultados baseado em Navegadores

Tabela 3.5 - Caso de utilização: Filtrar Resultados baseado em Navegadores

Identificador	CU04
Nome	Filtrar Resultados baseado em Navegadores
Descrição	Na interface de apresentação de resultados, deve ser disponibilizado um conjunto de Navegadores relacionados com os resultados devolvidos. Este navegadores são por exemplo: o tipo de ficheiro do resultado, o tamanho do ficheiro do resultado, a localização dos resultados, etc. O utilizador pode seleccionar um ou mais destes navegadores para refinar a sua pesquisa.
Cenário	Ao visualizar os resultados é disponibilizado um conjunto de navegadores seleccionáveis pelo utilizador que permitem o refinamento iterativo da sua pesquisa.
Actores	Utilizador
Prioridade	Média

Pré-Condições	O utilizador tem que efectuar uma pesquisa
Pós-Condições	Filtrar o resultado de acordo com o navegador e apresentar este “novo” resultado, respeitando o papel do utilizador e os seus privilégios de acesso aos recursos apresentados nos resultados
Pressupostos	Existem navegadores para a pesquisa efectuada.

Para melhor compreensão do funcionamento dos navegadores pode ver-se o seu diagrama de estados na Figura 3.5.

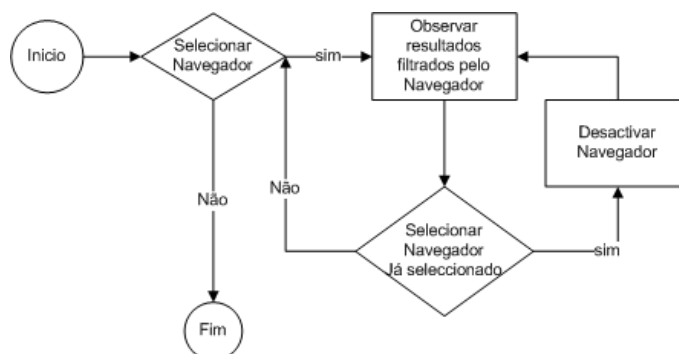


Figura 3.5 - Diagrama de estados: Filtrar Resultados baseado em Navegadores

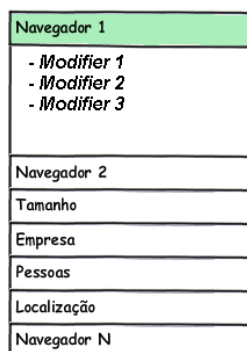


Figura 3.6 - Maqueta: Filtrar Resultados baseado em Navegadores

3.3.1.3.5 Navegar pelas páginas de resultados

Tabela 3.6 - Caso de utilização: Navegar pelas páginas de resultados

Identificador	CU05
Nome	Navegar pelas páginas de resultados
Descrição	Na apresentação de resultados deve ser disponibilizada uma barra e navegação de resultados no caso de uma pesquisa devolver muitos resultados. Por omissão uma página de resultados tem apenas 10 resultados. Os seguintes 10 estão na página seguinte, e por ai adiante.
Cenário	Após efectuar uma pesquisa que retorna muitos resultados, este devem ser paginados e o utilizador deve ser capaz de navegar entre todos os resultados devolvidos, através de uma barra de navegação.
Actores	Utilizador

Prioridade	Elevada
Pré-Condições	O utilizador tem que efectuar uma pesquisa
Pós-Condições	Mostrar os resultados da página seleccionada
Pressupostos	O conjunto de resultados ocupa mais que uma página



Figura 3.7 - Maqueta: Navegar pelas páginas de resultados

3.3.1.3.6 Abrir Resultado

Tabela 3.7 - Caso de utilização: Abrir Resultado

Identificador	CU06
Nome	Abrir Resultado
Descrição	Nos resultados apresentados o utilizador deve ser capaz de carregar num deles e ser direccionado para a página onde ele se encontra numa janela externa.
Cenário	O utilizador está a observar o conjunto de resultados devolvidos e selecciona o que pretende ver através de um <i>click</i> com o rato. Este resultado é aberto numa nova página externa ao SPOT
Actores	Utilizador
Prioridade	Elevada
Pré-Condições	Existir resultados
Pós-Condições	O resultado aberto é aquele que o utilizador seleccionou
Pressupostos	N/A

3.3.1.3.7 Observar os Resultados

Tabela 3.8 - Caso de utilização: Observar Resultados

Identificador	CU07
Nome	Observar Resultados
Descrição	Ao efectuar uma pesquisa, é apresentado ao utilizador um conjunto de resultados ordenados por relevância. Inclui o <i>highlighting</i> dos termos pesquisados.
Cenário	Após a introdução de texto e o premir do botão de pesquisa o utilizador observa os resultados referentes ao texto pesquisado caso existam.
Actores	Utilizador
Prioridade	Elevada
Pré-Condições	O utilizador tem que efectuar uma pesquisa
Pós-Condições	N/A
Pressupostos	A pesquisa devolve resultados

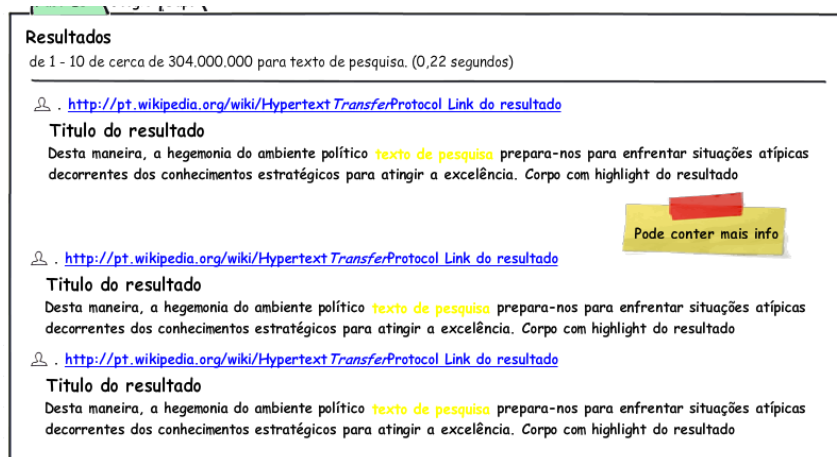


Figura 3.8 - Maqueta: Observar Resultados

3.3.1.3.8 Pesquisa por Sugestão

Tabela 3.9 - Caso de utilização: Pesquisa por Sugestão

Identificador	CU08
Nome	Pesquisa por Sugestão
Descrição	Deve ser sugerido um novo texto de pesquisa (correção ortográfica) após ser feita uma pesquisa. O utilizador deve ser capaz de carregar nessa sugestão de pesquisa e observar os resultados.
Cenário	Após o utilizador efectuar uma pesquisa, podem ser sugeridas outras pesquisas através de correção ortográfica. O utilizador pode então carregar na sugestão e observar os resultados resultantes da pesquisa sugerida.
Actores	Utilizador
Prioridade	Baixa
Pré-Condições	Efectuar uma pesquisa inicial que devolva uma sugestão
Pós-Condições	Apresentação de resultados de acordo com os parâmetros utilizados
Pressupostos	N/A

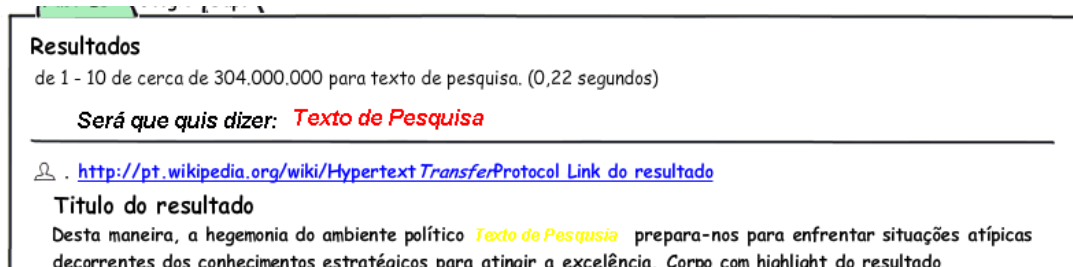


Figura 3.9 - Maqueta: Pesquisa por Sugestão

3.3.2 Requisitos Não Funcionais

3.3.2.1 Usabilidade

Usabilidade é a capacidade de um sistema ser simples e fácil de aprender, proporcionar uma interacção com o utilizador fácil de recordar, bem como potenciar uma experiência de utilização agradável. De forma a cumprir este requisito, o *front-end* de pesquisa terá uma interface, perfeitamente integrada no SPOT e com mecanismos de interacção intuitivos e um aspecto claro na visualização dos resultados.

3.3.2.2 Desempenho

Desempenho é a capacidade de um sistema ter um tempo de resposta dentro dos parâmetros esperados e estabelecidos. Como tal, o tempo de resposta a uma pesquisa deverá ser o menor possível, não devendo ultrapassar os 2 segundos até que se visualize resultados, de forma a manter o sistema atractivo para os utilizadores. As restantes acções desencadeadas pelo utilizador deverão ter uma resposta imediata. Note-se, que o desempenho não está estritamente dependente do *front-end*, mas sim da plataforma e pesquisa e das condições de carga da rede.

3.3.2.3 Fiabilidade

Fiabilidade é a capacidade de um sistema ser robusto e consistente, isto é, garantir que todas as funcionalidades estão disponíveis e a funcionar na totalidade em qualquer instante, sendo este mais um factor relevante para a satisfação final do utilizador.

3.3.2.4 Manutenção

Pretende-se que o *front-end* de pesquisa seja implementado de maneira a facilitar possíveis alterações futuras (por exemplo: implementar mais funcionalidades), e um dos factores cruciais para cumprir este requisito é a definição de uma boa arquitectura.

3.3.2.5 Requisitos de Sistema

Como a aplicação SPOT, em que o *front-end* de pesquisa vai ser embutido, foi desenvolvida utilizando a plataforma .NET 3.5 (Silverlight, WCF, etc), é necessário ter instalada a *framework* .NET 3.5 ou superior e utilizar o sistema operativo Windows. É também necessário ter instalado um browser, preferivelmente o Internet Explorer, pois o suporte para Mozilla Firefox, ou outros, não é garantido ainda.

3.3.2.6 Acessibilidade

O *front-end* de pesquisa a desenvolver deverá ter todas as suas funcionalidades e suporte na língua materna dos seus utilizadores, o Português. Todavia, todo o código desenvolvido deverá estar em Inglês, incluindo os comentários, facilitando o desenvolvimento de versões seguintes da aplicação.

3.3.2.7 Segurança

De acordo com os *stakeholders* a segurança é um dos elementos mais importantes. Será indexada e pesquisada um vasto leque de informação, alguma dela sensível, que deverá ser protegida.

Apesar, da segurança estar dependente da plataforma utilizada, o *front-end* de pesquisa deve também possuir todos os requisitos para garantir que não existe fuga de informação, utilizando para isso os mais modernos mecanismos de segurança.

3.3.3 Requisitos de Informação

Os requisitos de informação necessários ao funcionamento do *front-end* de pesquisa são apresentados esquematicamente através de um diagrama de classes que se mostra na Figura 3.10.

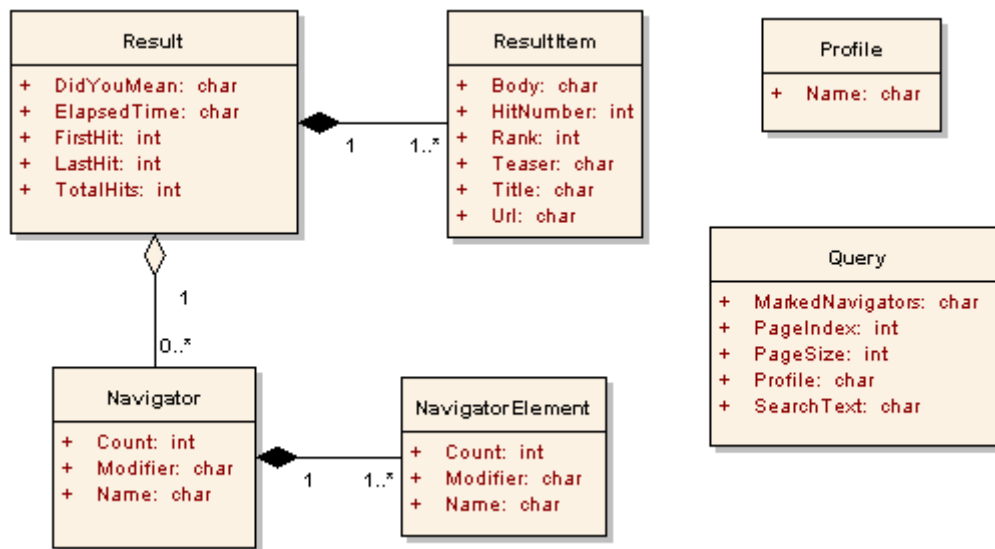


Figura 3.10 - Modelo de Dados

3.4 Análise Tecnológica

Como foi referido na secção anterior, a PTSI já possui um *dashboard* interno que disponibiliza algumas funcionalidades e concentra conteúdos de várias fontes de informação. Embora este *dashboard* tenha sido desenvolvido em Silverlight o *front-end* de pesquisa pode ser implementado recorrendo a outra tecnologia como por exemplo ASP.NET ou até mesmo Adobe Flex. Dadas as várias opções foi feito um estudo à tecnologia Silverlight para determinar se esta era a indicada.

A utilização de Silverlight tem as seguintes vantagens [22]:

- Aplicações multi-plataforma: suporte para Windows, Mac e Linux
- Linguagem XML para definir a interface
- Integração com vídeo
- Linguagem orientada para objectos
- Interfaces ricas graficamente
- Acesso facilitado a *web services* XML

- Ferramentas dedicadas à compilação
- Suporte para Visual Basic e C#
- Facilidade de integração com ASP.NET
- Suporte directo com Java e AJAX

E as seguintes desvantagens:

- Incompatível com versões antigas do Windows, como o Windows 2000 e 98.
- Necessidade de software específico para suporte.
- Fraca penetração no mercado a curto prazo.
- Tecnologia recente, logo ainda pouco desenvolvida

Pesando as vantagens e desvantagens do Silverlight e dada a facilidade de integração no *dashboard* SPOT de novas funcionalidades recorrendo a esta tecnologia, foi tomada a decisão de utilizar esta tecnologia para implementar o *front-end* de pesquisa.

3.5 Arquitectura

Nesta secção é definida a arquitectura a seguir na implementação do *front-end* de pesquisa. Numa primeira fase é apresentada uma visão geral da arquitectura sendo posteriormente apresentados mais detalhes.

3.5.1 Visão geral

A Figura 3.11 apresenta o diagrama com uma visão de alto nível do sistema a implementar. A aplicação SPOT, implementada em *silverlight*, vai incluir o *front-end* de pesquisa, que por sua vez vai aceder à camada de serviços disponibilizada pelo servidor SPOT. Esta camada de serviços funciona quase como *relay* para a camada de serviços da Fast ESP, mas contém alguma lógica de negócio.

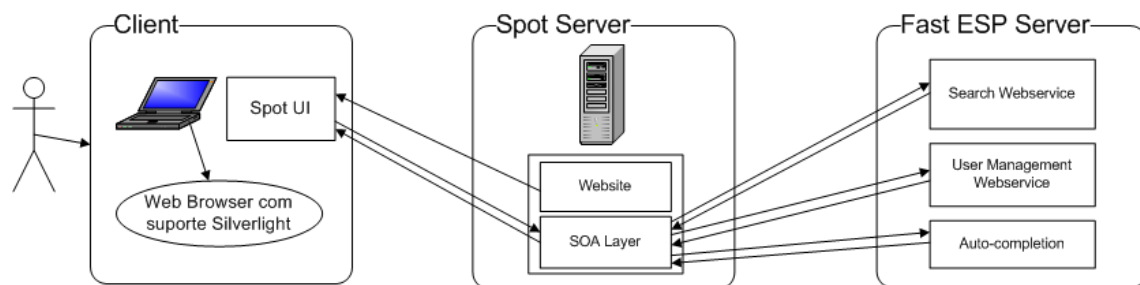


Figura 3.11 - Visão geral da arquitectura

3.5.2 Arquitectura Lógica

A arquitectura lógica tem como objectivo apresentar os vários componentes constituintes do projecto e a forma como se relacionam. Essa apresentação documenta a estrutura lógica de alto nível do sistema de software. Esta arquitectura pode ser subdividida em arquitectura horizontal e vertical o que permite obter uma maior visão do sistema. A arquitectura horizontal divide o sistema em camadas de forma relativamente independente das funcionalidades do

sistema, enquanto a arquitectura vertical divide o sistema por funcionalidades e abrange todas as camadas de implementação.

3.5.2.1 Decomposição Horizontal

A Figura 3.12 apresenta o diagrama com a decomposição horizontal. O Módulo Cliente é constituído por três camadas, todas implementadas em Silverlight e seguindo o padrão MVVM (Model, View, View Model) (ver a secção 3.5.2.1.13.5.2.1.1). A camada *User Interface* é implementada utilizando apenas XAML (*Extensible Application Markup Language*) que permite desenhar a interface gráfica. Como é independente das outras camadas o designer gráfico pode desenvolver em paralelo com o programador. A camada *Business Logic*, implementa a lógica das funcionalidades implementadas. A camada *Data Access Layer*, possui todos os meios para o acesso aos serviços disponibilizados pelo módulo de serviços.

A camada de serviços é também constituída por 3 camadas, em que a do topo disponibiliza os serviços para o Módulo Cliente, a do meio implementa alguma lógica, para o tratamento da informação que vem dos serviços disponibilizados pela plataforma de pesquisa. A camada *Data Access Layer* permite aceder aos serviços disponibilizados pela FAST ESP.

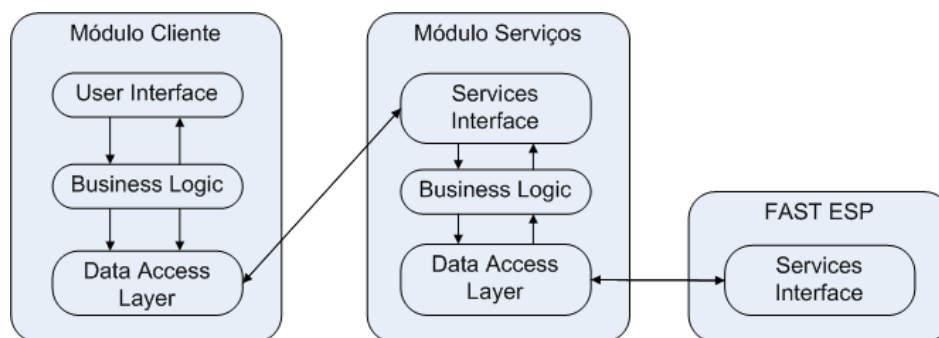


Figura 3.12 - Decomposição horizontal do sistema

3.5.2.1.1 Model, View, ViewModel

O *Model View ViewModel* (MVVM) é um padrão arquitectural usado em engenharia de software, criado pela Microsoft. Este padrão tem como objectivo separar duas áreas: a de apresentação (*design*) e a de lógica de negócio e dados. Assim, o designer pode concentrar-se em criar uma interface rica e agradável e o programador pode concentrar-se em toda a lógica da aplicação e os dados que este acede, sendo que ambos podem trabalhar em paralelo, pois as camadas são independentes [23].

3.5.2.2 Decomposição Vertical

Na decomposição vertical dos dois módulos definidos na secção anterior, são descritas as funcionalidades e responsabilidades de cada um deles. Note-se que esta é uma visão de alto nível.

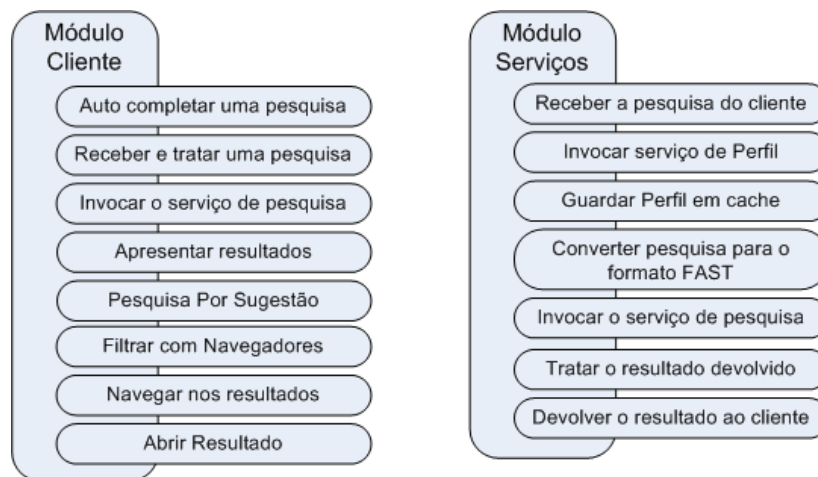


Figura 3.13 - Decomposição vertical do sistema

No módulo do cliente, a primeira funcionalidade “Auto completar uma pesquisa” permite ao utilizador não ter de introduzir todo o texto de pesquisa. É contactado o servidor de *auto-completion* que devolve textos de pesquisa que são iniciados pelo texto introduzido pelo utilizador, e este pode seleccionar um destes e assim, não necessita escrever todo o texto de pesquisa. “Receber e tratar uma pesquisa” implica receber o texto do cliente e criar uma estrutura que seja compreendida pelo serviço de pesquisa alojado na camada de serviços. Após ter esta estrutura o “Invocar o serviço de pesquisa” trata de passar esta estrutura de pesquisa para a camada de serviços e aguarda a resposta. Ao receber a resposta contendo uma estrutura que tem de ser tratada e “*renderizada*” para que sejam apresentados os resultados, esta tarefa está a cargo de “Apresentar resultados”. A funcionalidade “Pesquisa por Sugestão” permite que o utilizador submeta uma nova pesquisa sugerida pelo sistema.

É dada a opção de filtrar os resultados através de navegadores, “Filtrar com Navegadores” implica fazer uma nova pesquisa, agora passando o filtro do navegador. É necessário guardar o estado dos navegadores pois o serviço de pesquisa não mantém o estado (*stateless*). Como a lista de resultados pode ser longa é necessário permitir a navegação pelas várias páginas dos resultados, “Navegar nos resultados” trata esta componente. Por último, “Abrir Resultado” permite que o utilizador carregue num dos resultados e este seja aberto numa nova página do *web browser*.

No módulo de serviços, “Receber a pesquisa do cliente” disponibiliza o serviço e o esquema de dados ao cliente. Posteriormente, é necessário saber qual o perfil do utilizador que esta a invocar o serviço, então, “Invocar serviço de Perfil” trata de fazer isso. Para não estar sempre a invocar o serviço de Perfil, o “Guardar o perfil em cache” trata de salvar por um tempo definido o nome do perfil do utilizador a usar o serviço naquele momento. “Converter a pesquisa para formato Fast” trata de criar uma estrutura conhecida pelo serviço de pesquisa do FAST, para que “Invocar o serviço de Pesquisa” possa requerer os resultados da pesquisa efectuada. Como a plataforma FAST devolve o resultado em XML, é necessário tratar este resultado para um formato mais fácil de trabalhar e isto é feito pelo “Tratar o resultado devolvido”. Por último, o resultado pode ser devolvido ao cliente (“Devolver o resultado ao cliente”).

3.5.3 Arquitectura Física e Tecnologias

Esta secção tem como objecto apresentar uma visão da arquitectura física de todo o sistema e as tecnologias utilizadas em cada subsistema.

Na Figura 3.14 podemos constatar que a aplicação SPOT pode ser acedida através da rede, por vários utilizadores. Estes necessitam ter um *web browser* instalado na sua máquina (note-se que de momento apenas o Internet Explorer é completamente suportado) e é necessário ter o

Silverlight instalado para poder visualizar a aplicação. O FAST Server, aloja a plataforma de pesquisa empresarial, que indexa e disponibiliza para pesquisa os conteúdos seleccionados. Utiliza o servidor Web Apache com o servidor de aplicações Tomcat que disponibiliza os *webservices* de Pesquisa e de Perfil.

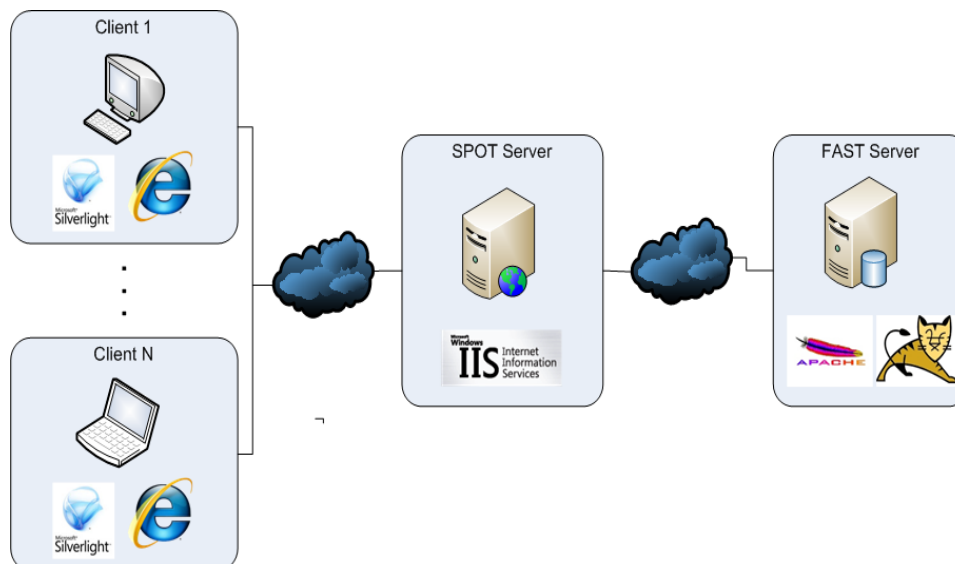


Figura 3.14 - Diagrama da arquitectura física

3.5.3.1 Tecnologias

Apresentam-se de seguida as tecnologias utilizadas no desenvolvimento da aplicação, integrada no SPOT, que serve de *front-end* de pesquisa à plataforma FAST ESP.

3.5.3.1.1 Microsoft Silverlight

O Microsoft Silverlight é uma *framework* aplicacional para a *Web* que disponibiliza funcionalidades semelhantes às do Adobe Flash. Esta tecnologia é suportada por diversos *browsers Web* em diversos sistemas operativos. A lógica de negócio pode ser implementada através das linguagens da plataforma .NET e as suas ferramentas de desenvolvimento [24].

3.5.3.1.2 Webservices

Os *Webservices* constituem uma tecnologia que é utilizada na integração de sistemas diferentes e possivelmente em meio heterogéneo, que utiliza o XML em conjunção com o HTTP. O XML serve de linguagem universal que permite que aplicações diferentes possam “falar” entre si e o HTTP disponibiliza o protocolo de comunicação.

3.6 Planeamento do desenvolvimento

Na figura Figura 3.15 apresenta-se o diagrama de Gantt contendo o planeamento do desenvolvimento do *front-end* de pesquisa com as tarefas necessárias à sua implementação e a sua respectiva alocação temporal na Figura 3.16. As tarefas indicadas a azul, são subtarefas.

ID	Task Name	Start	Finish	Duration
1	Efectuar Pesquisa Simples	04-01-2010	22-01-2010	15d
2	Proxy para Serviço Fast	04-01-2010	04-01-2010	4h
3	Parse do resultado em XMLFast	04-01-2010	06-01-2010	2d
4	Criar o WCF search Service	06-01-2010	08-01-2010	2d 4h
5	Criar o WCF profile Service	11-01-2010	12-01-2010	2d
6	Caching do profile	13-01-2010	14-01-2010	2d
7	Consumir o search Service	15-01-2010	15-01-2010	1d
8	Apresentar resultados	18-01-2010	22-01-2010	5d
9	Utilizar auto-complete numa pesquisa	25-01-2010	29-01-2010	4d 4h
10	Criar serviço auto-complete	25-01-2010	27-01-2010	2d
11	Consumir o serviço de auto-complete	27-01-2010	28-01-2010	1d 4h
12	Renderizar o auto-complete service para a auto-complete text box	29-01-2010	29-01-2010	1d
13	Filtrar Resultados baseado em Navegadores	01-02-2010	04-02-2010	3d 4h
14	Navegar pelas páginas de resultados	04-02-2010	10-02-2010	4d
15	Efectuar Pesquisa com Operadores	10-02-2010	11-02-2010	1d 4h
16	Abrir Resultado	25-01-2010	25-01-2010	4h
17	Seleccionar Opções da Plataforma de Pesquisa	12-02-2010	15-02-2010	1d 4h
18	Testes	15-02-2010	19-02-2010	4d 4h

Figura 3.15 - Planeamento do desenvolvimento do *front-end* de pesquisa (parte 1)

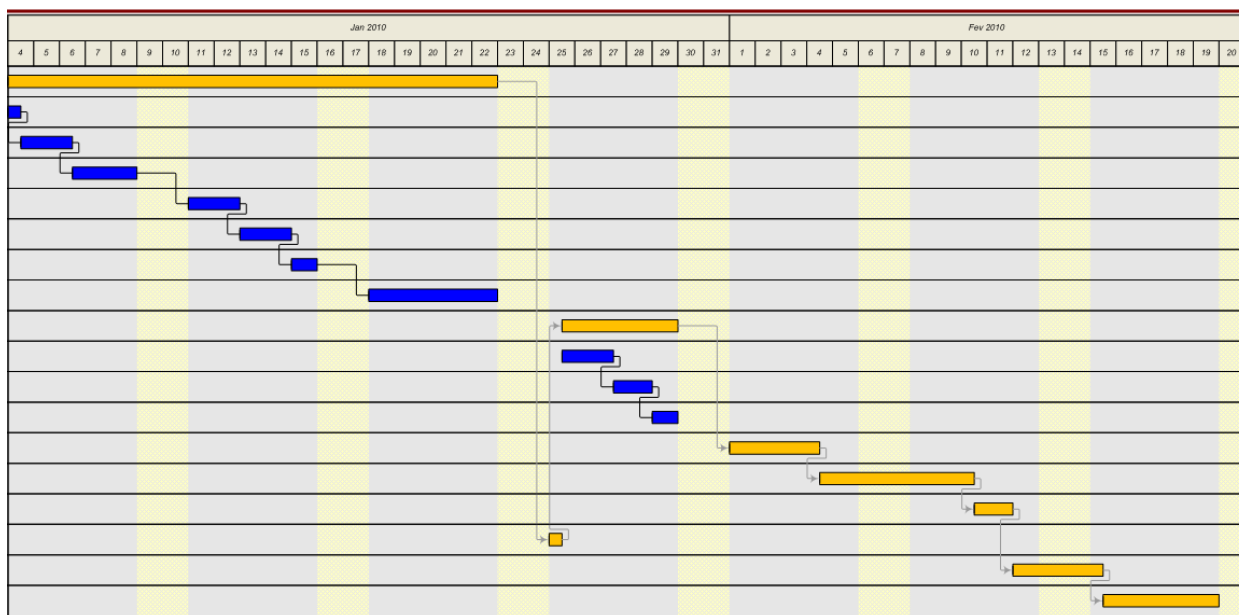


Figura 3.16 - Planeamento do desenvolvimento do *front-end* de pesquisa (parte 2)

4 Desenvolvimento

Este capítulo tem como objectivo apresentar alguns detalhes dos sistemas utilizados, relevantes para o desenvolvimento do projecto. É também apresentada, a forma como foram implementadas as soluções e os resultados do trabalho que foi implementado ao longo do projecto.

4.1 *Front-end* de Pesquisa

A plataforma FAST ESP disponibiliza um *Web front-end* OOTB, intitulado *Search Front-End* (SFE). Este pode ser utilizado em produção ou para testes. Por questões de melhoramento de usabilidade ou necessidade de integração noutras aplicações, que podem ser executadas em dispositivos fixos ou móveis, a plataforma disponibiliza uma API para desenvolvimento de *front-ends*.

4.1.1 Vista geral da ESP Search API

A aplicação criada pode aceder à FAST ESP através de uma das APIs disponibilizadas ou directamente por HTTP. As APIs permitem um maior nível de abstracção e facilitam a utilização de mecanismos avançados da plataforma com é o caso da *Navigation* e *Result Clustering* [25].

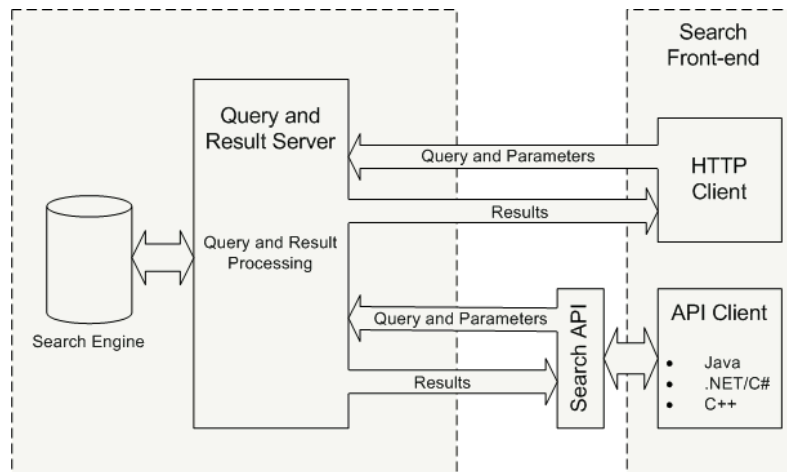


Figura 4.1 - FAST ESP *framework* de integração [25]

A *Search API* disponibiliza interfaces e métodos para criar *queries* e iterar sobre o resultado destas *queries*. A API comunica com o *Query and Results Server* através de uma interface HTTP. É possível também utilizar directamente pedidos sobre a interface HTTP, mas neste caso os resultados podem ser devolvidos num formato de texto compactado ou XML. As APIs permitem a utilização de SSL para garantir segurança [25].

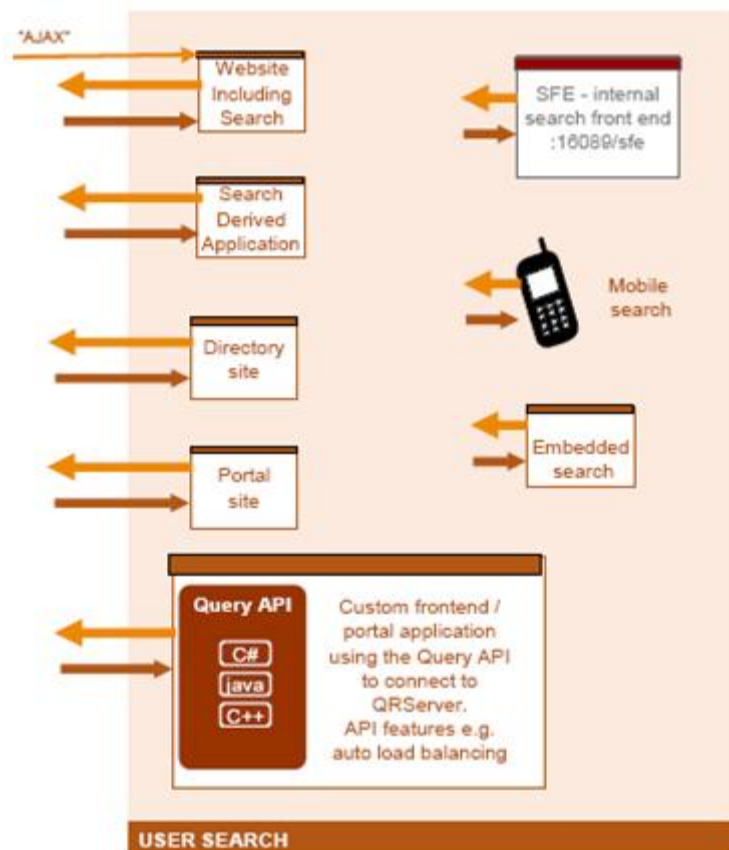


Figura 4.2 - Formas de acesso ao *Query & Results Server* [26]

4.1.2 Search API Web Service Query Interface

A *Search API Web Service* permite um acesso universal ao motor de pesquisa de uma qualquer aplicação que seja capaz de seguir a especificação WSDL fornecida [25].

4.1.2.1 Arquitectura

A sequência de passos da *Search API Web Service* está ilustrada na Figura 4.3. É (passo 1) executada num contentor de um servidor de aplicações externo à FAST ESP (passo 2), e descreve a sua interface para um documento WSDL (passo 3) disponível no servidor de aplicações. O serviço não mantém estado (*stateless*). O cliente do *web service* (passo 4) descarrega a especificação WSDL, para poder construir uma *query*. A *query* é então enviada para a *Search API Web Service* utilizando o protocolo SOAP [25].

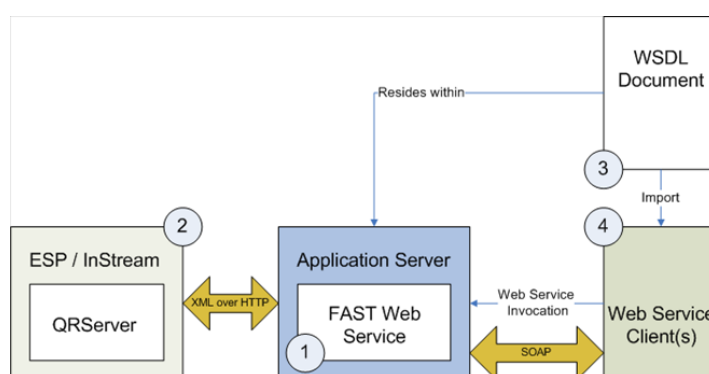


Figura 4.3 - Arquitectura Search API Web Service [25]

A *Search API Web Service* recebe o pedido e efectua uma chamada bloqueante ao *Query and Result Server* (QR-server) sobre uma interface HTTP interna. O XML resultante do QR-server é devolvido para o cliente do *web service* [25]. O *schema* completo do XML pode ser visto no Anexo A.

4.1.2.2 Funcionalidade e Performance

A *Search API Web Service* não consome significantes ciclos de CPU e não mantém o estado (*stateless*) de invocação em invocação. É acrescentado algum *overhead* em relação à utilização das FAST ESP APIs directamente, mas não é relevante se contra balançarmos com o aumento de portabilidade e outras funcionalidades que podem ser implementadas. No entanto, existe uma limitação que é a de não suportar *Result Clustering* para "*offset>0*"⁴ [25]. Desta forma, só é possível ter *Result Clustering* se todos os resultados forem apresentados de uma só vez, não existindo então a paginação. No entanto, com alguma programação é possível superar esta limitação.

Não existem mecanismos de autenticação e autorização de ponto a ponto, mas é possível utilizar mecanismos de segurança disponibilizados pelo servidor de aplicações e o cliente do *web service*. A comunicação entre *Search API Web Service* e o FAST ESP pode ser configurada para usar *Secure Sockets Layer* (SSL) [25].

⁴ Entenda-se *offset* como parte da paginação. Isto é, o *offset* igual a zero é equivalente ter a primeira página com um determinado número de resultados. Para a página seguinte o *offset* será igual ao número de resultados na primeira página, ou seja, o valor de deslocamento que temos que aplicar desde o primeiro resultado.

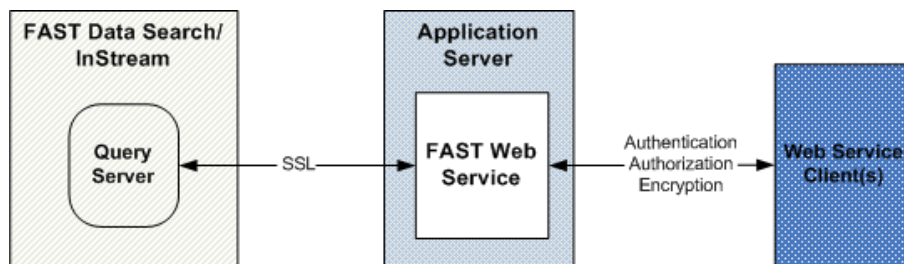


Figura 4.4 - Arquitectura de segurança da *Search API Web Service* [25]

4.1.3 Desenvolvimento do *Front-End*

Para o desenvolvimento do *front-end* foi utilizado um conjunto de ferramentas para a construção do seu aspecto final.

4.1.3.1 Ferramentas

4.1.3.1.1 Visual Studio Team System

O Microsoft Visual Studio é um pacote de programas da Microsoft, para desenvolvimento de software, especialmente dedicado à *framework* .NET e às linguagens Visual Basic (VB), C, C++, C# (C Sharp) e J# (J Sharp). Permite também o desenvolvimento de aplicações *web-based*, usando a plataforma ASP.NET [27]. Inclui um sistema de gestão de versões.

Foi utilizada para desenvolver todo o código produzido durante o projecto e para controlo de versões.

4.1.3.1.2 WSDL Tool

A WSDL Tool permite gerar código para *web services* XML e clientes de Web services XML a partir de ficheiros de contracto WSDL, esquemas XSD e documentos *.dicomap discovery* [28].

Um ficheiro *.wsdl* é um documento escrito em XML denominada *Web Services Description Language* (WSDL). Este ficheiro, define a forma como o *web service* se comporta e informa os clientes de como interagir com o serviço [28].

Ao utilizar esta ferramenta é possível criar uma classe *proxy*, definida apenas num ficheiro e na linguagem que pretendemos (C#, Visual Basic, JScript ou Visual J#) [28].

Esta ferramenta foi utilizada para criar as duas classes *proxy*, a primeira para o serviço de pesquisa e a segunda para o serviço de gestão de utilizadores.

4.1.3.1.3 SOAP UI

O SOAP UI é uma ferramenta *open source* para testar *web services* numa arquitectura orientada a serviços criada para auxiliar os programadores. As suas funcionalidades permitem inspeccionar os *web services*, invocar, desenvolver e simular, para citar algumas [29].

Foi utilizada essencialmente para observar e compreender os conteúdos que eram devolvidos pelos *web services* utilizados.

4.1.3.1.4 Wireshark

O Wireshark (conhecido previamente por Ethereal) é um analisador de tráfego da rede [30]. Este programa foi utilizado para detectar lacunas de segurança no trânsito da informação pela rede.

4.1.3.2 Diagrama de Classes

A Figura 4.5 e Figura 4.6 apresentam os diagramas de classes para cada um dos módulos desenvolvidos.

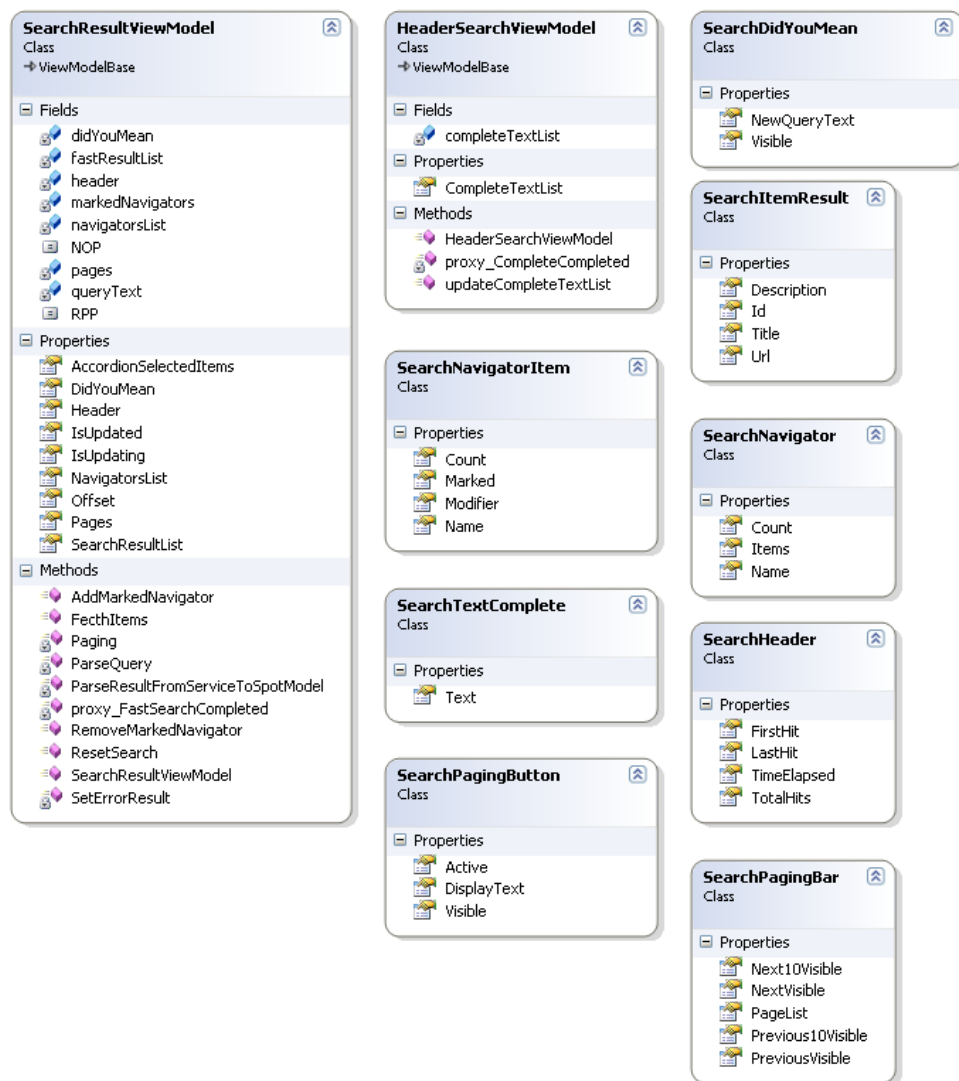


Figura 4.5 - Diagrama de classes do módulo cliente

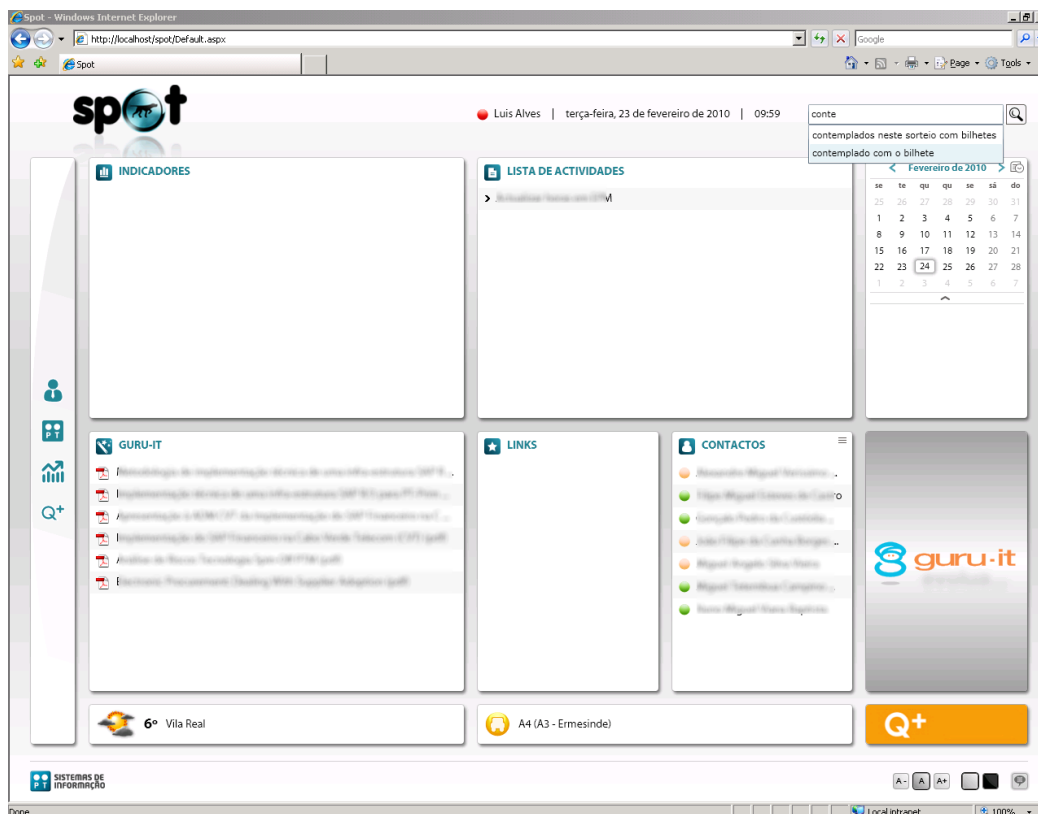


Figura 4.7 - Pesquisa com Auto-Complete integrada no SPOT

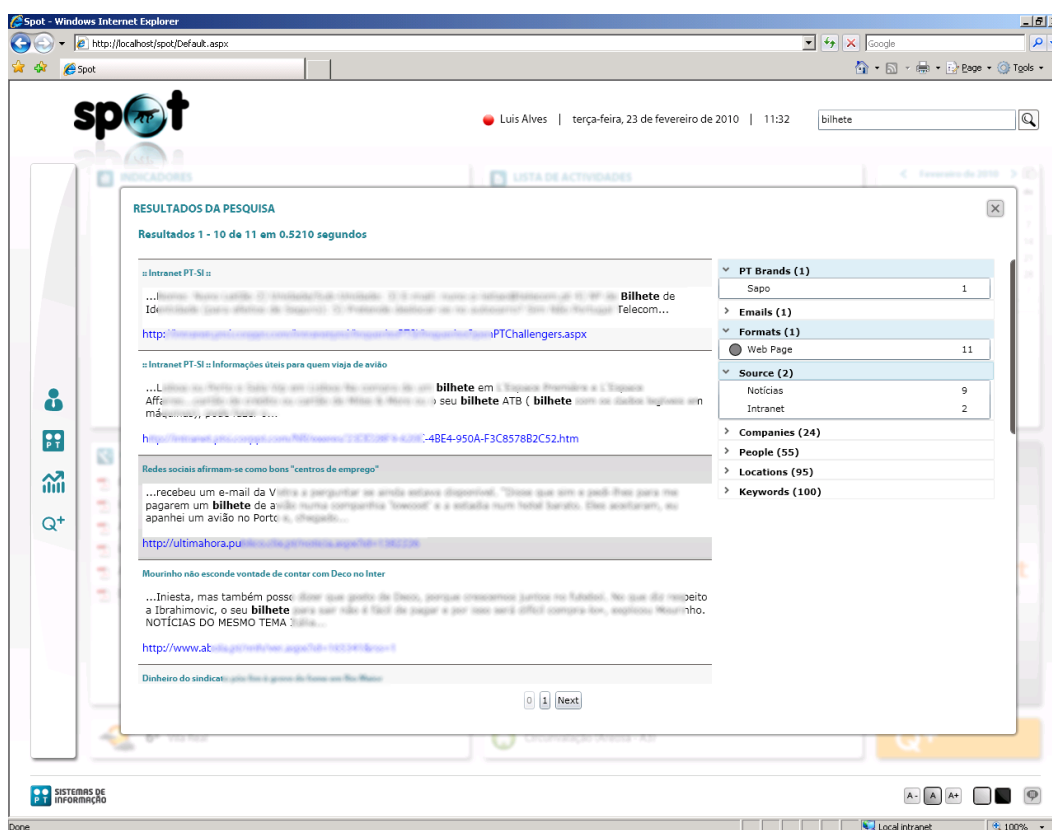


Figura 4.8 - Aspecto do front-end de pesquisa integrado no SPOT

4.2 Ajuste dos Resultados da FAST ESP

Uma plataforma de pesquisa empresarial deve devolver resultados o mais relevantes possíveis, para que possa satisfazer os seus utilizadores e as necessidades do negócio. Para isso, são utilizados modelos de classificação para ordenação dos resultados, que determinam se um determinado documento devem ou não pertencer ao resultado de uma pesquisa. Podemos definir a relevância, como o quão bem o conjunto de resultados responde ao intuito de uma dada pesquisa. Então, dependendo do utilizador ou do papel que este desempenha, a ordem ou relevo que se dá a determinados documentos, para a mesma pesquisa, pode variar [31].

Observando do ponto de vista do contexto em que está inserida a plataforma, uma página *Web* de notícias ou uma loja *Web*, o modo como o modelo de classificação de resultados actua deve ser diferente. Isto porque, numa página de notícias, talvez o mais importante seja a actualidade das mesmas, assim as mais recentes devem aparecer primeiro. Já numa loja *Web* talvez seja interessante que os produtos mais antigos apareçam primeiro, para efectuar o seu escoamento, ou até mesmo os produtos em promoção [31].

Temos então dois vectores de ajuste de pesquisas, o primeiro baseado no utilizador e o seu perfil, o segundo no contexto em que está a ser utilizada a plataforma. A FAST ESP disponibiliza estes dois tipos de ajuste, a primeira através do *Search Business Center* (SBC) e a segunda através de um ficheiro em XML que pode ser carregado através da plataforma administrativa e posteriormente utilizado em cada perfil no SBC.

4.2.1 *Search Business Center*

O SBC é a ferramenta onde podem ser feitas ajustes de resultados, monitorização de pesquisas e relatórios. Podem ser geridos os mecanismos de classificação e ordenação dos resultados, sinónimos e navegadores, entre outras coisas. Podemos configurar e testar todas as alterações feitas e depois publicar para o sistema de produção [32].

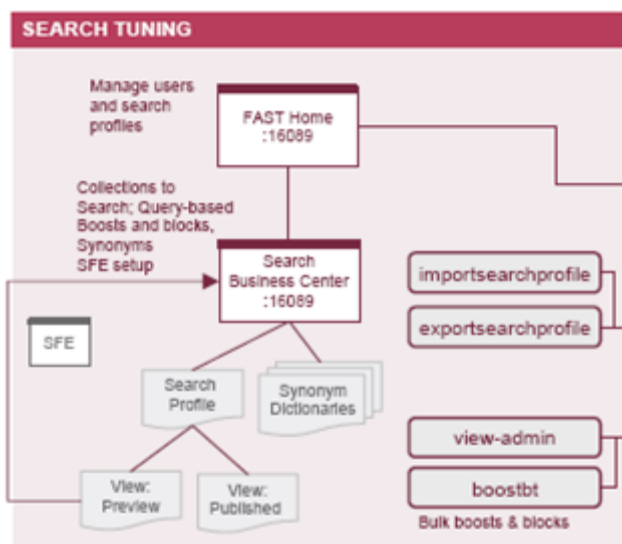


Figura 4.9 - Módulo de ajuste de pesquisas [26]

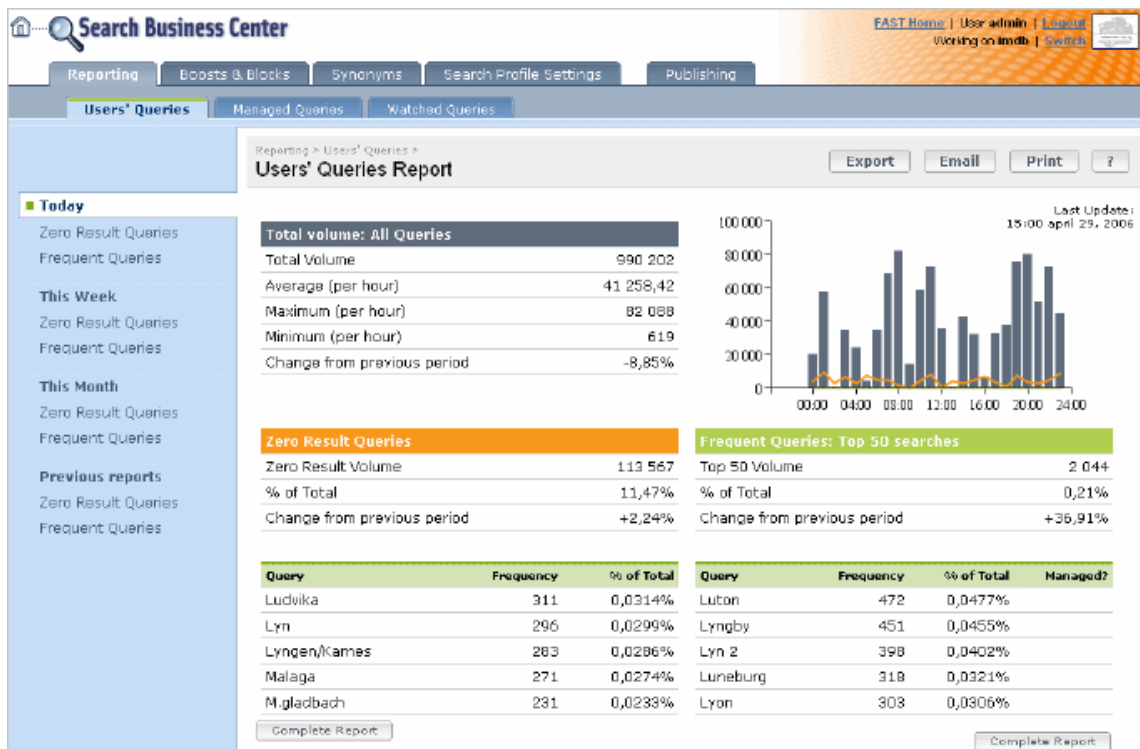


Figura 4.10 - Interface do SBC [32]

Para afinar um perfil, podemos alterar os resultados para uma dada pesquisa através do *Boost & Block*. Este permite bloquear resultados ou então “impulsionar” (*boost*) o resultado através da alteração do valor do seu nível interno ou colocando-o no top 10 dos resultados. Este tipo de afinação pode requerer algum tempo e experiência por parte do administrador, mas pode ser bastante útil, por exemplo numa loja *Web* para colocar produtos em promoção no top 10 [32].

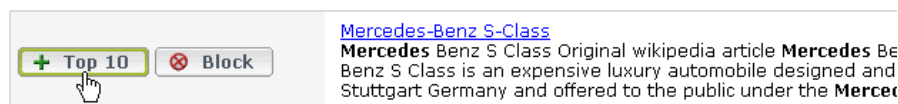


Figura 4.11 - Bloquear resultado ou colocá-lo no Top 10 [32]

☐ Top 10 by position
 ☒ Any number, by rank points
 Recalculate

1 Rank points Base Score: 4409 Boosts: #1 Document Boost: 10000 Total Score: # 1 Block	Mercedes-Benz S-Class Mercedes-Benz S-Class Original wiki Class" The Mercedes-Benz S-Class i designed and built by DaimlerChrysl the public... Size: Location: http://osl-skialg-dt.oslo.fa Similar: Find Refine Exclude
2 Rank points Base Score: 4420 Boosts: 2000 Document Boost: 10000 Total Score: 16420	Mercedes-Benz 450Sel 6.9 Mercedes-Benz 450Sel 6.9 Original v 450Sel 6.9" Mercedes-Benz 450SEL spec 450SEL 6.9. North American ver Size: Location: http://osl-skialg-dt.oslo.fa Similar: Find Refine Exclude

Figura 4.12 - Aplicar valor de *boost* manualmente [32]

É possível acrescentar sinónimos, que permitem que o texto de uma dada pesquisa seja alterado por algo semelhante que melhore os resultados da pesquisa. Por exemplo, se pesquisarmos por “*notebook*” pode ser interessante incluir resultados que incluam a palavra “*portable*”. Estes sinónimos podem ser retirados de um dicionário ou então podemos criar sinónimos específicos para a terminologia do negócio em questão [32].

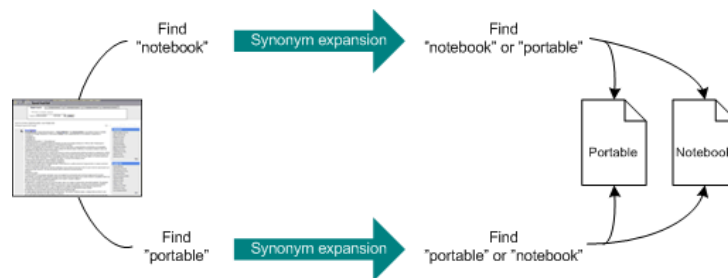


Figura 4.13 - Funcionamento dos sinónimos [32]

Existem três modos de sinónimos, que permitem funcionalidades diferentes [32]:

- **Two-way expansion** – uma pesquisa com a palavra-chave ou qualquer um dos sinónimos vai devolver resultados que contêm a palavra-chave e/ou os sinónimos.
- **One-way expansion** – uma pesquisa com a palavra-chave vai devolver resultados que contêm a palavra-chave e/ou os sinónimos.
- **Rewrites to** – a palavra introduzida não é utilizada, sendo substituída pelos sinónimos

Cada perfil tem uma série de definições que podem ser configuradas na aba *Search Profile Settings*. Podemos definir o nome do perfil, as colecções sobre as quais são feitas as pesquisa, os navegadores utilizados, filtros a utilizar, e o perfil de nível, para citar alguns [32].

Search Business Center

FAST Home | User Working on n

Reporting Boosts & Blocks Synonyms Search Profile Settings Publishing

General Query Handling Navigator Output Result View Rank Profile Preview Ap

Search Profile Settings > General > **General** manitest Save Preview

Search Profile Name **manitest**

Display Name News Search Ex. "Company Intranet"

Display Thumbnail Filename  c:\sites\news_thumb.jpg Browse... Max size of image is 15KB.

Published Search Profile URL http://osl-100-dt.london

Collections searched

Company Available --> --< wiki News To Search

Index field to be searched content

Figura 4.14 - Definições de um *Search Profile* [32]

4.2.2 Matching Engine Profile

Como foi referido na Introdução, a plataforma de ES pode estar introduzida em diversos contextos e para cada contexto as métricas que avaliam a relevância do resultado podem ser diferentes.

Os mecanismos da plataforma FAST ESP possuem o conjunto de métricas para a avaliação da relevância dos resultados, que se mostra na Tabela 4.1.

Tabela 4.1 – Descrição das métricas de classificação de resultado [14]

Frescura	O quão fresco é o documento comparado com a idade da pesquisa?
Autoridade	O documento é considerado uma autoridade para esta pesquisa?
Plenitude	O quão bem, a pesquisa corresponde a um contexto superior (título, URL)?
Estatísticas	O quão bem, o conteúdo do documento corresponde à pesquisa na sua globalidade?
Qualidade	Qual é a qualidade do documento? O quão importante é ele?
Geo-espacial	O quão longe, o resultado está de uma referência geográfica?

Estes podem ser configurados através de um ficheiro XML, onde podemos atribuir pesos a cada um deles e assim definir critérios para a classificação e ordenação dos resultados. A plataforma já possui alguns perfis OOTB, mas estes podem não cumprir as nossas necessidades. Podemos fazer uma analogia com o equalizador de um sistema áudio, em que existem já alguns estilos pré definidos (*flat, rock, full bass, classical*) e depois existe um modo em que podemos ser nós a afinar cada uma das componentes [31]. Apresenta-se na Figura 4.15 a descrição em formato XML de alguns dos perfis OOTB. Note-se que existem secções destacadas a amarelo vivo, que consistem no nome dos perfis e as características que mais os definem e destacam dos outros.

```
<!-- Rank profiles. -->
<rank-profile name="news" rank-model="news" default="no"
  stop-word-threshold="2E6"
  position-stop-word-threshold="2E7">
  <authority weight="50" field-ref="anchortext"/>
  <freshness weight="200" field-ref="docdatetime" auto="yes"/>
  <composite-rank composite-field-ref="content">
    <proximity weight="50" />
    <context weight="50">
      <field-weight field-ref="body" value="5"/>
      <field-weight field-ref="description" value="30"/>
      <field-weight field-ref="urlkeywords" value="40"/>
      <field-weight field-ref="keywords" value="50"/>
      <field-weight field-ref="title" value="60"/>
    </context>
  </composite-rank>
</rank-profile>
<rank-profile name="site" rank-model="site" default="no"
  stop-word-threshold="2E6"
  position-stop-word-threshold="2E7">
  <authority weight="70" field-ref="anchortext"/>
  <freshness weight="0" field-ref="docdatetime" auto="yes"/>
  <composite-rank composite-field-ref="content">
    <proximity weight="60" />
    <context weight="70">
      <field-weight field-ref="body" value="5"/>
      <field-weight field-ref="description" value="30"/>
      <field-weight field-ref="urlkeywords" value="40"/>
      <field-weight field-ref="keywords" value="50"/>
      <field-weight field-ref="title" value="60"/>
    </context>
  </composite-rank>
</rank-profile>
<rank-profile name="yellowpages" rank-model="contextboost"
  default="no"
  stop-word-threshold="2E6"
  position-stop-word-threshold="2E7">
  <authority weight="0" field-ref="anchortext"/>
  <freshness weight="0" field-ref="docdatetime" auto="yes"/>
  <composite-rank composite-field-ref="content">
    <proximity weight="0" />
    <context weight="100">
      <field-weight field-ref="body" value="5"/>
      <field-weight field-ref="description" value="30"/>
      <field-weight field-ref="urlkeywords" value="40"/>
      <field-weight field-ref="keywords" value="50"/>
      <field-weight field-ref="title" value="60"/>
    </context>
  </composite-rank>
</rank-profile>
```

Figura 4.15 - Exemplo de perfis de classificação de resultados em XML

Segundo os fornecedores da plataforma estes campos só devem ser alterados por especialistas em RI.

4.2.3 Trabalho de Afinação

O trabalho de afinação decorreu em duas fases. Numa primeira fase definiram-se os requisitos e numa segunda efectuou-se o desenvolvimento do trabalho de afinação, dando origem aos resultados obtidos.

4.2.3.1 Requisitos e Desenvolvimento

O trabalho de afinação foi iniciado através da definição dos papéis existentes no âmbito do funcionamento da aplicação SPOT. Foi organizada uma reunião com o orientador do projecto na PTSI, pois só este possuía a experiencia e o conhecimento interno necessário, para a definição destes papéis. Ficaram definidos quatro papéis:

- Consultor
- Gestor
- Director
- Administrador

No SBC foram criados e configurados cada um dos papéis acordados na reunião, como pode ser visto na Figura 4.16.



Figura 4.16 - Search Profiles

Após a definição dos papéis, foram definidas as fontes (coleções) de informação que cada um dos papéis teria acesso.

Tabela 4.2 - Alocação de repositórios aos papéis

	Consultor	Gestor	Director	Administrador
Intranet	✓	✓	✓	✓
Guru-It ⁵	✓	✓	✓	✓
SharePoint:				
adm				✓
cdn			✓	✓
int	✓	✓	✓	✓
inf			✓	✓
gp			✓	✓
mint		✓	✓	✓
mext		✓	✓	✓
out	✓	✓	✓	✓
mc			✓	✓
devprod	✓	✓	✓	✓
rcf				✓
gtr			✓	✓
siq	✓	✓	✓	✓
pcoperacional		✓	✓	✓
dgr			✓	✓
dom	✓	✓	✓	✓
dmwline	✓	✓	✓	✓
dws	✓	✓	✓	✓
de	✓	✓	✓	✓
dmi	✓	✓	✓	✓

As coleções (ver Tabela 4.3) foram criadas e configuradas na ferramenta de administração da plataforma de pesquisa e associadas a cada um dos papéis.

Tabela 4.3 - Nome das coleções criadas

Consultor	Gestor	Director	Administrador
intranet	intranet	intranet	intranet
guruit	guruit	guruit	guruit
shareConsult	shareGest	shareDirect	shareAdmin

Note-se que os resultados não dependem apenas do perfil do utilizador, pois os privilégios de acesso a determinados conteúdos variam de utilizador para utilizador.

```
or(xrank(generic2:"ColeçãoA",
boost=9000),xrank(generic2:"ColeçãoB",
```

⁵ O Guru-It é uma base de conhecimento interna que permite aos colaboradores da empresa partilharem o seu conhecimento.

```
boost=2000),xrank(generic2:"ColecçãoC", boost=5000))
```

Figura 4.17 - Sintaxe para atribuir pesos diferentes às colecções

Ficou também definido que cada colecção deve ter um peso diferente, isto é, deve ser dado um relevo maior a um resultado de uma colecção do que um resultado doutra. Para atingir este objectivo foi necessário criar uma expressão, que é aplicada como parâmetro estático para as pesquisas, e este encontra-se definido na Figura 4.17.

A expressão da Figura 4.17 aplica um “*boost*” com um determinado peso aos documentos que no campo “*generic2*” tem um determinado valor. Neste caso, o valor é o nome de cada colecção.

4.2.3.2 Resultados

Para uma real avaliação de resultados seria necessário que durante algum tempo, nunca inferior a duas semanas, fosse efectuado um teste de aceitação envolvendo um grupo razoável de pessoas afectas a cada um dos perfis mencionados anteriormente e que estas os seus comentários e reacções para que fosse possível ajustar as configurações durante e após o teste.

Como por motivos de planeamento temporal, entre outros, isso não foi possível, nesta fase ainda se dispõe de qualquer tipo de resultados.

5 Conclusões e Trabalho Futuro

5.1 Satisfação dos Objectivos

O desenvolvimento de um *front-end* de pesquisa para a plataforma seleccionada ficou completo, isto é, foram cumpridos todos os requisitos solicitados no início do projecto. Teve ainda a mais-valia de ficar completamente integrado numa ferramenta já existente que tem como objectivo ser um *dashboard* personalizado para cada um dos colaboradores da empresa. Deste modo, este não tem de aceder a diversos sistemas para encontrarem aquilo de que necessitam. Por exemplo: antes de possuírem o *front-end* desenvolvido, um utilizador que necessitasse de informação do Guru-It teria de aceder ao Guru-It, autenticar-se e usar os seus mecanismos de pesquisa. Nos instantes seguintes, caso sentisse a necessidade de visualizar um documento alojado no Sharepoint da empresa, então teria de aceder ao Sharepoint, autenticar-se, e utilizar o sistema de pesquisa do Sharepoint. Neste momento, o utilizador é capaz de efectuar as duas pesquisas no mesmo aplicativo, e como o aplicativo possui Single Sign On⁶ (SSO) não há necessidade de introduzir as suas credenciais.

No entanto, considera-se que ficou uma parte relevante do trabalho por fazer, que consistia na avaliação da implementação dos perfis de utilizador, que favorecia a obtenção de resultados mais relevantes consoante a função do colaborador na empresa. Ainda não foi possível fazer esta avaliação, porque esta fase do projecto ainda se encontra em ambiente de desenvolvimento. Assim que for feita uma instalação geral da solução, já será possível avaliar o sucesso da implementação da plataforma, considerando principalmente estes dois vectores: a adesão dos colaboradores à sua utilização e a satisfação com os resultados obtidos. Friso que este processo pode ser consideravelmente moroso, dependendo das expectativas dos utilizadores.

Note-se que ao permitir pesquisas sobre o Guru-It (solução de base de conhecimentos) vai permitir que muitas das pesquisas efectuadas em motores de pesquisa *Web* externos à empresa, como é o caso do Google e Bing, sejam agora substituídos por pesquisas internas.

5.2 Trabalho Futuro

Existe muito trabalho futuro nesta área, visto esta apenas estar agora a ser introduzida na empresa. O *front-end* pode incluir muitas mais funcionalidades, principalmente funcionalidades

⁶ O *Single Sign On* (SSO) é definido como um único ponto de autenticação. Um utilizador necessita apenas de apresentar as suas credenciais de acesso uma vez, e desta forma, consegue ter acesso de forma automática a um conjunto de aplicações que suportem este mecanismo.

de pesquisa avançada, para utilizadores mais experientes e com outras necessidades. Tendo sempre em conta que muitos estudos de soluções de pesquisa *Web* apontam para que apenas uma pequena percentagem de utilizadores faz uso destas funcionalidades avançadas. Outra funcionalidade que se considera interessante é a possibilidade de pré-visualização dos documentos, mas que só será possível implementar assim que os repositórios de informação permitirem SSO.

Quanto à plataforma de pesquisa, sendo esta bastante configurável e expansível, existe oportunidade de algum desenvolvimento nesta área. Podem ser desenvolvidos conectores para outros repositórios, conseguindo assim agregar mais informação. O processamento dos documentos é também configurável.

Considera-se que algo interessante seria o próprio utilizador conseguir influenciar os resultados, na medida em que pudesse “votar”, de forma positiva ou negativa, se o resultado obtido foi ou não relevante para ele.

Nota-se também que os dicionários para a nossa língua materna, não são de facto os melhores. Penso que isso se deve ao facto de o número de potenciais clientes que falam Português ser bastante inferior aos que falam por exemplo Inglês ou Chinês.

6 Referências

1. **Singhal, Amit.** Modern Information Retrieval: A Brief Overview. [Online] [Citação: 2 de Novembro de 2009.] <http://singhal.info/ieee2001.pdf>.
2. **Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze.** *Introduction to Information Retrieval*. s.l. : Cambridge University Press, 2008.
3. **Gonçalves, Alexandre.** *Enterprise Search - Technical Benchmark Analysis*. s.l. : PT Sistemas de Informação, 2009.
4. **Autonomy Corporation.** *Autonomy ACI API - Programming Guide*. 2008.
5. **Baptista, Nuno.** Proposta de Dissertação. [Online] [Citação: 24 de Novembro de 2009.] https://www.fe.up.pt/si/estagios_empresas.ver_dados_proposta?p_id=30498.
6. **PT - Sistemas de Informação .** Quem Somos. [Online] [Citação: 4 de Novembro de 2009.] http://www.ptsi.pt/PTSI/quem_somos.html.
7. **Microsoft ES .** FAST ESP Brochure. [Online] [Citação: 23 de Novembro de 2009.] http://download.microsoft.com/download/B/F/D/BFD2EBD0-1100-4821-B0EA-4BAC1535B3A6/FAST_ESP_Brochure.pdf.
8. **Answers.com.** Information retrieval. [Online] [Citação: 26 de Fevereiro de 2010.] <http://www.answers.com/topic/information-retrieval>.
9. **Thaker, Sahil.** Information Retrieval & Search - Basic IR Models. [Online] [Citação: 16 de Janeiro de 2010.] http://blogs.msdn.com/blogfiles/spt/WindowsLiveWriter/InformationRetrieval_14EB0/image_4.png.
10. **Pires, Marina Melo.** Agrupamento Incremental e Hierárquico de Documentos. [Online] [Citação: 9 de Fevereiro de 2010.] http://www.coc.ufrj.br/index.php?option=com_content&task=view&id=3906&Itemid=178.

11. **Chaves, Marcirio Silveira.** Um Estudo e Apreciação sobre Algoritmos de Stemming. [Online] [Citação: 9 de Fevereiro de 2010.] <http://mchaves.wdfiles.com/local--files/publicacoes/stemming-jii2003>.
12. **Paolo Rosso, Edgardo Ferretti, Daniel Jiménez e Vicente Vidal.** Text Categorization and Information Retrieval Using WordNet Senses. [Online] [Citação: 19 de Fevereiro de 2010.] <http://www.fi.muni.cz/gwc2004/proc/110.pdf>.
13. **Ruch, Patrick.** Using contextual spelling correction to improve retrieval effectiveness in degraded text collections. [Online] [Citação: 19 de Fevereiro de 2010.] <http://www.aclweb.org/anthology/C/C02/C02-1109.pdf>.
14. **Microsoft ES.** FAST ESP Datasheet. [Online] [Citação: 23 de Novembro de 2009.] http://download.microsoft.com/download/7/3/A/73AB39D7-52FE-4EB0-8941-AFF78BDD35F0/FAST_ESP_Data_Sheet.pdf.
15. **Autonomy.** Enterprise Search. *Enterprise Search from Autonomy*. [Online] [Citação: 26 de Novembro de 2009.] <http://www.autonomy.com/content/Technology/evolution/evolution-of-search-pan-enterprise-search/index.en.html>.
16. **FAST.** *Deployment Planning Guide*. s.l. : Fast Search & Transfer ASA, 2009.
17. **Autonomy.** Scalability and Performance. [Online] [Citação: 26 de Novembro de 2009.] <http://www.autonomy.com/content/Products/products-idol-server-scalability/index.en.html>.
18. —. *Performance and Scalability White Paper*.
19. —. IDOL functionality. [Online] [Citação: 26 de Novembro de 2009.] <http://www.autonomy.com/content/Functionality/idol-functionality-conceptual-search/index.en.html>.
20. **FAST ESP.** *Query Language and Parameters Guide*. 2009.
21. **Gonçalves, Alexandre.** *Enterprise Search - Technical Benchmark Analysis*.
22. **Geoxplorer.** Vantagens & Desvantagens: Adobe FLEX vs. Microsoft SILVERLIGHT. [Online] [Citação: 10 de Dezembro de 2009.] http://geoxplorer.files.wordpress.com/2008/03/flex_vs_silverlight.pdf.
23. **Smith, Josh.** WPF Apps With The Model-View-ViewModel Design Pattern. [Online] [Citação: 9 de Fevereiro de 2010.] MVVM is targeted at modern UI development platforms (WPF and Silverlight) in which there is a UX developer who has different requirements than a more “traditional” developer (i.e. oriented toward business logic and back end development).
24. **Microsoft.** What is Silverlight. *Microsoft Silverlight*. [Online] 2009. [Citação: 18 de Dezembro de 2009.] <http://www.microsoft.com/silverlight/overview/default.aspx>.
25. **FAST.** *Query Integration Guide*.
26. —. *ESP 5 System Overview*.

27. **Microsoft.** Microsoft Visual Studio. [Online] [Citação: 8 de Fevereiro de 2010.] O Microsoft Visual Studio é um pacote de programas da Microsoft, para desenvolvimento de software, especialmente dedicado, ao framework .NET e às linguagens Visual Basic (VB), C, C++, C# (C Sharp) e J# (J Sharp). Também é um grande produto de desenvolvim.
28. —. Web Services Description Language Tool (Wsdll.exe). *.NET Framework Developer Center*. [Online] [Citação: 18 de Dezembro de 2009.] <http://msdn.microsoft.com/en-us/library/7h3ystb6%28VS.71%29.aspx>.
29. **Eviware.** soapUI; the Web Services Testing tool. [Online] [Citação: 18 de Fevereiro de 2010.] <http://www.soapui.org/>.
30. **Wireshark Foundation.** Wireshark - Go Deep. [Online] [Citação: 18 de Fevereiro de 2010.] <http://www.wireshark.org/>.
31. **FAST, Search Best Practices.** *Relevancy and search - A business white paper*.
32. **FAST.** *Search Business Center Guide*.
33. **Computer Business Review.** 2008.
34. **Google.** *Google Search Appliance 6 - Datasheet*.
35. **Microsoft.** XML Schema Definition Tool (Xsd.exe) . *.NET Framework Developer Center*. [Online] [Citação: 18 de Dezembro de 2009.] <http://msdn.microsoft.com/en-us/library/x6c1kb0s%28VS.80%29.aspx>.
36. **Sherman, Chris.** Are Search Engines Biased? [Online] [Citação: 26 de Fevereiro de 2010.] <http://searchenginewatch.com/2159431>.

Anexo A: *Schema* do resultado devolvido pelo webservice SearchWS

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!--W3C Schema generated by XMLSpy v2010 (http://www.altova.com)-->
<!--Please add namespace attributes, a targetNamespace attribute and import elements
according to your requirements-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
  <xs:complexType name="SEGMENTS">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="SEGMENT"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="SEGMENTS" type="SEGMENTS"/>
  <xs:complexType name="SEGMENT">
    <xs:sequence>
      <xs:element ref="RESULTPAGE"/>
    </xs:sequence>
    <xs:attribute name="NAME" type="xs:anySimpleType" use="required"/>
  </xs:complexType>
  <xs:element name="SEGMENT" type="SEGMENT"/>
  <xs:complexType name="RESULTPAGE">
    <xs:choice>
      <xs:sequence>
        <xs:element ref="QUERYTRANSFORMS"/>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="CATEGORYLIST"/>
        </xs:choice>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element ref="NAVIGATION"/>
        </xs:choice>
        <xs:element ref="CLUSTERS"/>
        <xs:choice>
          <xs:sequence>
            <xs:element ref="RESULTSET"/>
            <xs:element ref="PAGENAVIGATION"/>
          </xs:sequence>
          <xs:element ref="EMPTYRESULTSET"/>
        </xs:choice>
      </xs:sequence>
      <xs:element ref="ERROR"/>
    </xs:choice>
  </xs:complexType>
  <xs:element name="RESULTPAGE" type="RESULTPAGE"/>
  <xs:complexType name="QUERYTRANSFORMS">
    <xs:sequence maxOccurs="unbounded">
      <xs:element ref="QUERYTRANSFORM"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="QUERYTRANSFORMS" type="QUERYTRANSFORMS"/>
  <xs:complexType name="QUERYTRANSFORM">
    <xs:attribute name="NAME" type="xs:anySimpleType" use="required"/>
    <xs:attribute name="ACTION" type="xs:anySimpleType" use="required"/>
    <xs:attribute name="QUERY" type="xs:anySimpleType" use="required"/>
    <xs:attribute name="CUSTOM" type="xs:anySimpleType" use="required"/>
    <xs:attribute name="MESSAGE" type="xs:anySimpleType" use="required"/>
    <xs:attribute name="MESSAGEID" type="xs:anySimpleType" use="required"/>
  </xs:complexType>
  <xs:element name="QUERYTRANSFORM" type="QUERYTRANSFORM"/>
  <xs:complexType name="CATEGORYLIST" mixed="true"/>
  <xs:element name="CATEGORYLIST" type="CATEGORYLIST"/>

```

```

<xs:complexType name="NAVIGATION">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="NAVIGATIONENTRY"/>
  </xs:sequence>
  <xs:attribute name="HITS" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="ENTRIES" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="NAVIGATION" type="NAVIGATION"/>
<xs:complexType name="NAVIGATIONENTRY">
  <xs:sequence>
    <xs:element ref="NAVIGATIONELEMENTS"/>
  </xs:sequence>
  <xs:attribute name="NAME" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="DISPLAYNAME" type="xs:anySimpleType"
use="required"/>
  <xs:attribute name="TYPE" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="UNIT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="MODIFIER" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="SCORE" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="SAMPLECOUNT" type="xs:anySimpleType"
use="required"/>
  <xs:attribute name="HITCOUNT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="RATIO" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="MIN" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="MAX" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="MEAN" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="ENTROPY" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="NAVIGATIONENTRY" type="NAVIGATIONENTRY"/>
<xs:complexType name="NAVIGATIONELEMENTS">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="NAVIGATIONELEMENT"/>
  </xs:sequence>
  <xs:attribute name="COUNT" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="NAVIGATIONELEMENTS" type="NAVIGATIONELEMENTS"/>
<xs:complexType name="NAVIGATIONELEMENT">
  <xs:attribute name="NAME" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="MODIFIER" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="COUNT" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="NAVIGATIONELEMENT" type="NAVIGATIONELEMENT"/>
<xs:complexType name="CLUSTERS">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="CLUSTER"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="CLUSTERS" type="CLUSTERS"/>
<xs:complexType name="CLUSTER">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="NODE"/>
  </xs:sequence>
  <xs:attribute name="TYPE" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="CLUSTER" type="CLUSTER"/>
<xs:complexType name="NODE">
  <xs:sequence>
    <xs:element ref="LABELS" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="MEMBERS" minOccurs="0"
maxOccurs="unbounded"/>

```

```

        </xs:sequence>
        <xs:attribute name="SUBMEMCNT" type="xs:anySimpleType"
use="required"/>
        <xs:attribute name="EXTHITCNT" type="xs:anySimpleType"/>
    </xs:complexType>
    <xs:element name="NODE" type="NODE"/>
    <xs:complexType name="LABELS">
        <xs:sequence maxOccurs="unbounded">
            <xs:element ref="LABEL"/>
        </xs:sequence>
        <xs:attribute name="COUNT" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="LABELS" type="LABELS"/>
    <xs:complexType name="LABEL" mixed="true"/>
    <xs:element name="LABEL" type="LABEL"/>
    <xs:complexType name="MEMBERS">
        <xs:sequence maxOccurs="unbounded">
            <xs:element ref="MEMBER"/>
        </xs:sequence>
        <xs:attribute name="COUNT" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="MEMBERS" type="MEMBERS"/>
    <xs:complexType name="MEMBER">
        <xs:attribute name="OFFSET" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="MEMBER" type="MEMBER"/>
    <xs:complexType name="RESULTSET">
        <xs:choice maxOccurs="unbounded">
            <xs:element ref="HIT"/>
            <xs:element ref="BADHIT"/>
        </xs:choice>
        <xs:attribute name="FIRSTHIT" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="LASTHIT" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="HITS" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="TOTALHITS" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="MAXRANK" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="TIME" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="RESULTSET" type="RESULTSET"/>
    <xs:complexType name="HIT">
        <xs:sequence maxOccurs="unbounded">
            <xs:element ref="FIELD"/>
        </xs:sequence>
        <xs:attribute name="NO" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="RANK" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="SITEID" type="xs:anySimpleType" use="required"/>
        <xs:attribute name="MOREHITS" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="HIT" type="HIT"/>
    <xs:complexType name="FIELD" mixed="true">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element ref="sep"/>
            <xs:element ref="key"/>
        </xs:choice>
        <xs:attribute name="NAME" type="xs:anySimpleType" use="required"/>
    </xs:complexType>
    <xs:element name="FIELD" type="FIELD"/>
    <xs:complexType name="key" mixed="true"/>
    <xs:element name="key" type="key"/>
    <xs:complexType name="sep"/>

```

```

<xs:element name="sep" type="sep"/>
<xs:complexType name="BADHIT"/>
<xs:element name="BADHIT" type="BADHIT"/>
<xs:complexType name="PAGENAVIGATION">
  <xs:sequence>
    <xs:element ref="PREVPAGE" minOccurs="0"/>
    <xs:element ref="NECTPAGE" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="PAGENAVIGATION" type="PAGENAVIGATION"/>
<xs:complexType name="PREVPAGE">
  <xs:attribute name="FIRSTHIT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="LASTHIT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="URL" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="PREVPAGE" type="PREVPAGE"/>
<xs:complexType name="NECTPAGE">
  <xs:attribute name="FIRSTHIT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="LASTHIT" type="xs:anySimpleType" use="required"/>
  <xs:attribute name="URL" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="NECTPAGE" type="NECTPAGE"/>
<xs:complexType name="ERROR" mixed="true">
  <xs:attribute name="CODE" type="xs:anySimpleType" use="required"/>
</xs:complexType>
<xs:element name="ERROR" type="ERROR"/>
<xs:complexType name="EMPTYRESULTSET"/>
<xs:element name="EMPTYRESULTSET" type="EMPTYRESULTSET"/>
</xs:schema>

```


Anexo B: Visão geral da arquitectura FAST ESP

V1.1 Delivery Efficiency October 2007

Queries In

Results Out

NOTE:
All port numbers are given relative to a base port of 13000.
For other base ports, please subtract 13000 from port number given and then add your base port to arrive at corresponding port number for your installation.

