INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

Explorador integrado de serviços de armazenamento na *Cloud* – Cloud Cockpit

Rodolfo Tiago Gameiro Cardoso

(Licenciado)

Projeto para obtenção do grau de Mestre em Engenharia Informática e de Computadores

Orientador:

Professor Doutor Paulo Manuel Trigo Cândido da Silva

Júri:

Presidente:

Mestre Pedro Alexandre Sela Cunha Ribeiro Pereira

Vogais:

Mestre Carlos Jorge de Sousa Gonçalves

Professor Doutor Paulo Manuel Trigo Cândido da Silva

Setembro de 2013

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

Explorador integrado de serviços de armazenamento na *Cloud* – Cloud Cockpit

Rodolfo Tiago Gameiro Cardoso

(Licenciado)

Projeto para obtenção do grau de Mestre em Engenharia Informática e de Computadores

Orientador:

Professor Doutor Paulo Manuel Trigo Cândido da Silva

Júri:

Presidente:

Mestre Pedro Alexandre Sela Cunha Ribeiro Pereira

Vogais:

Mestre Carlos Jorge de Sousa Gonçalves

Professor Doutor Paulo Manuel Trigo Cândido da Silva

Setembro de 2013

Resumo

O modelo de computação na *Cloud* apresentou, nos últimos anos, um crescimento significativo e evolui para a forma predominante de fornecimento e consumo de recursos computacionais. Este modelo tem como objetivo o fornecimento de sistemas com alta escalabilidade, performance, fiabilidade e recursos configuráveis *on-demand*. Simultaneamente reduz a complexidade de gestão de uma infraestrutura de TI (Tecnologias de Informação) com estas características.

Neste contexto surgem os serviços de armazenamento na *Cloud* que exploram o conceito da computação na *Cloud* para fornecimento de espaço de armazenamento flexível, fiável e *on-demand*. No atual modelo armazenamento na *Cloud* as empresas fornecem tanto o espaço de armazenamento como os serviços ou aplicações de acesso e controlo deste espaço. No entanto este modelo não permite que o utilizador explore o seu espaço de armazenamento de diferentes serviços de forma integrada devido à existência barreiras físicas e lógicas.

Neste projeto é analisado e implementado um sistema que permite explorar diversos serviços de armazenamento na *Cloud* de forma integrada. O sistema, designado por Cloud Cockpit, tem como principais características: (i) integração dos serviços Dropbox e Google Drive (sendo extensível a outros serviços de armazenamento), (ii) pesquisa sobre os conteúdos não estruturados armazenados nos serviços que integra e (iii) interface multiplataforma que abrange computadores, *tablets* e *smartphones*.

O sistema implementado foi testado e avaliado na totalidade das suas funcionalidades. A implementação da interface multiplataforma apresenta-se uma solução viável ao nível de abrangência de dispositivos. A arquitetura do sistema, para além de integrar os serviços Dropbox e Google Drive, é extensível para integração de outros serviços de armazenamento. Os testes revelam que deve ser encontrada uma alternativa ao componente de extração de texto de ficheiros PDF (*Portable Document Format*) devido a problemas de desempenho.

Palavras-Chave

Serviços de armazenamento na *Cloud*, Integração e gestão de dados, Pesquisa em conteúdos não estruturados, Aplicação multiplataforma

Abstract

Over the last few years Cloud Computing had a significant growth and evolves to the predominant model for delivery and consumption of computer resources. This model aims to supply systems with high scalability, performance, reliability and on-demand configuration. Simultaneously reduces the complexity of managing an IT infrastructure with such characteristics.

In this context arise in the Cloud storage services based on the concept of Cloud Computing providing flexible, reliable and on-demand storage space. In the current model the Cloud storage companies provide both storage space and services or applications to access and control this space. However this model does not allow the user to explore their storage space from different services in an integrated way because there are physical and logical barriers.

In this project it is analyzed and implemented a system that provides the capability to explore various Cloud storage services in an integrated manner. The system, referred to as Cloud Cockpit, has as main features: (i) integration of services Dropbox and Google Drive (being extensible to other storage services), (ii) search on the unstructured content stored in the integrated services, (iii) multiplatform interface covering Computers, Tablets and Smartphone's.

The implemented system was tested and evaluated in the totality of its features. The implementation of the multiplatform interface is a viable solution in terms of covered devices. The system architecture integrate services Dropbox and Google Drive and is extensible to integrate other Cloud storage services. Tests show that an alternative must be found to the extracting text from PDF (Portable Document Format) files component due performance issues.

Keywords

Cloud Storage Services, Integration and data management, Unstructured content search, Multiplatform application.

Agradecimentos

O lado bom da vida é encontrarmos pessoas que nos ajudam a atingir os nossos objetivos e concretizar sonhos. Este trabalho marca o final e o início de uma nova etapa, que só se proporciona devido a pessoas especiais que me acompanharam sempre neste percurso e fizeram de mim uma pessoa feliz e realizada. Assim, gostaria de deixar umas palavras de agradecimento:

Em especial, aos meus pais, que foram aqueles que me iniciaram neste percurso e fizeram de mim a pessoa que sou hoje. Sem eles nada seria possível, o meu obrigado por todo o apoio ao longo destes anos.

Á minha melhor amiga e esposa, Marisa Cardoso, pelo apoio incondicional ao longo destes anos e por me apoiar na concretização dos meus objetivos.

Ao meu orientador Professor Doutor Paulo Trigo por todo o empenho, acompanhamento, compreensão, encorajamento e dedicação, bem como o grau de exigência e excelência que demonstrou na busca de uma solução adequada.

Aos meus colegas Pedro Brito, Pedro Januário e Paulo Fagundes que me acompanharam ao longo do Mestrado.

Aos meus colegas e professores e todos aqueles que não referi especificamente, mas que de alguma forma estiveram presentes ao longo de todo este percurso e fizeram parte da minha experiência académica e de vida.

Acrónimos

API	Aplication Programming Interface
ASP	Aplication Service Provider
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
JSON	Javascript Object Notation
PaaS	Platform as a Service
REST	REpresentational State Transfer
SaaS	Software as a Service
SDK	Software Development Kit
TI	Tecnologias de Informação
XML	eXtensible Markup Language

Índice

Índice de Figuras

Índice de Listagens

Capítulo 1

Introdução

O sector das tecnologias de informação (TI) evolui com inovações que alteram substancialmente a forma como se utiliza a tecnologia, como se acede à informação, na forma de gerar valor acrescentado aos negócios e até na forma do utilizador comum interagir com o mundo no seu dia-a-dia. Estas evoluções verificaram-se, por exemplo, com a Internet que permitiu o acesso à informação independentemente da localização geográfica. Ou com o aparecimento do *Application Service Providers* (ASPs) que permitiram interagir com as aplicações diretamente através do *browser* e que são disponibilizadas a múltiplos clientes, ao invés das instalações *standalone* de aplicações. A mais recente evolução deu-se com o surgimento da *Cloud* que, em linguagem comum, se refere a um conjunto de recursos computacionais disponibilizados através de serviços na rede.

O modelo de computação na *Cloud (Cloud Computing)* apresentou, nos últimos anos, um crescimento significativo no sector das Tecnologias de Informação (TI). Este modelo evolui para a forma predominante de fornecimento e consumo de: infraestruturas, tanto a nível de armazenamento com a nível de computação, plataformas e aplicações. O objetivo deste modelo é fornecer sistemas com alta escalabilidade, performance, fiabilidade e recursos configuráveis *on-demand*. Simultaneamente reduz a complexidade da gestão de uma infraestrutura de TI com estas características [1].

Neste contexto surge, naturalmente, o armazenamento de dados que explora as características da *Cloud* para fornecer ao utilizador um espaço de armazenamento flexível, fiável e que potencia a mobilidade do utilizador. Este modelo tem vantagens a

nível financeiro, uma vez que os recursos na *Cloud* são tipicamente mais económicos do que recursos dedicados. Tem também vantagens a nível de segurança dos dados uma vez que a redundância protege o utilizador de falhas de *hardware* ou de eliminação acidental de dados.

Atualmente existem soluções fornecidas pelas grandes companhias de IT como a Google, a Amazon, a Microsoft ou a IBM que oferecem serviços de armazenamento de dados. Existem ainda serviços para o armazenamento e partilha tipos de conteúdo específicos tal como o Youtube [2] ou o Vimeo [3] para vídeos e o Picasa [4] ou Flicker [5] para armazenamento e partilha de fotografias.

1.1 Motivação

No modelo atual de armazenamento na *Cloud* as empresas fornecem tanto o espaço de armazenamento como os serviços ou aplicações de acesso e controlo deste espaço. O problema desta abordagem consiste no facto de os conteúdos de um utilizador armazenados em fornecedores diferentes estarem separados por barreiras físicas e lógicas que não permitem que sejam utilizados de forma integrada.

1.2 Objetivos

O presente trabalho tem como objetivo implementar um sistema que permita a exploração de diferentes serviços de armazenamento na *Cloud* de uma forma integrada. Para tal serão explorados os modelos de interação (i.e. API REST) disponibilizadas por estes serviços com o objetivo de suportar as operações elementares sobre ficheiros assim como a pesquisa de conteúdos não estruturados.

Em síntese, os objetivos do presente trabalho são:

- Disponibilizar as operações elementares sobre ficheiros armazenados nos serviços Dropbox [6] e Google Drive [7];
- Disponibilizar uma interface Web (ASP.NET MVC) utilizando um *Responsive Layout* (HTML5 e CSS3) que se adapte a ecrãs de computadores, *tablets* e *smartphones*;

- Disponibilizar a pesquisa sobre os conteúdos não estruturados (recorrendo ao suporte do Lucene) que permita aos utilizadores pesquisar os seus conteúdos armazenados nos serviços de armazenamento na *Cloud*.

1.3 Organização do documento

Este documento está organizado em onze capítulos. O primeiro capítulo descreve, de forma breve, o problema que se pretende resolver. O segundo capítulo apresenta os conceitos mais importantes do tema desenvolvido. O terceiro capítulo detalha o modelo proposto para a solução do problema. Do quarto ao nono capítulo detalha-se a implementação dos módulos que constituem a solução. O décimo capítulo apresenta os testes efetuados ao sistema no sentido de validar a solução. Por último, no décimo-primeiro capítulo, são discutidos os resultados do trabalho realizado bem como a apresentação de alguns aspetos para a continuidade do projeto.

1.4 Convenções de Escrita

Este documento foi escrito segundo o acordo ortográfico de 1990. São, também, utilizadas as seguintes convenções de escrita:

- As traduções de termos ou expressões originalmente em língua inglesa serão mantidas na sua designação original, sempre que a sua tradução não se justifique devido a perda de significado ou inexistência de tradução adequada sendo também acompanhadas da utilização de texto em itálico;
- O texto em itálico é utilizado em palavras ou expressões relativas a designações de conceitos ou estrangeirismos existentes no texto.

Capítulo 2

Trabalho Relacionado

Neste capitulo são apresentadas algumas noções sobre os temas abordados neste trabalho no sentido de sustentar as abordagens adotadas. São também apresentados alguns desenvolvimentos e trabalhos relacionados com o sistema desenvolvido.

2.1 Modelo de computação na *Cloud*

De acordo com a definição do *National Institute of Standards and Technology* (NIST), [8], a computação na *Cloud* é um modelo que permite o acesso ubíquo e a-pedido (*on-demand)* a um conjunto partilhado e configurável de recursos computacionais, através de serviços na rede. Estes recursos podem ser facilmente aprovisionados ou libertados com um custo mínimo de gestão ou interação com o prestador de serviço.

Este modelo assenta nas seguintes cinco características:

- Recursos a-pedido − O utilizador pode aprovisionar-se de recursos computacionais *on-demand* e sem necessidade de intervenção humana com os provedores do serviço;
- Acesso através da rede − Os recursos são disponibilizados na rede através de mecanismos que promovam uma utilização de clientes heterogéneos;
- Recursos partilhados − O fornecedor do serviço disponibiliza os recursos necessários a vários consumidores utilizando tecnologias como a virtualização e *multitenancy*[1].

[1] Conceito que refere uma instância de software que serve múltiplos clientes e cada pode personalizar alguns aspetos da sua aplicação (virtualizada)

- Escalabilidade rápida − Os recursos podem ser aprovisionados e libertados de uma forma rápida. Para o consumidor não devem existir limites nos recursos que tem disponíveis para alocar.
- Medição de consumo dos serviços – Tem de ser possível efetuar a medição dos serviços consumidos em termos de recursos utilizados, de forma a que seja possível faturar apenas os recursos utilizados.

O modelo de computação na *Cloud* é subdividio em três categorias: *Infrastructure as a Service (IaaS)*, *Platform as a Service (PaaS)* e *Software as a Service (SaaS)*. Estas categorias são definidas pelos diferentes serviços que oferecem conforme se apresenta na Figura 1.

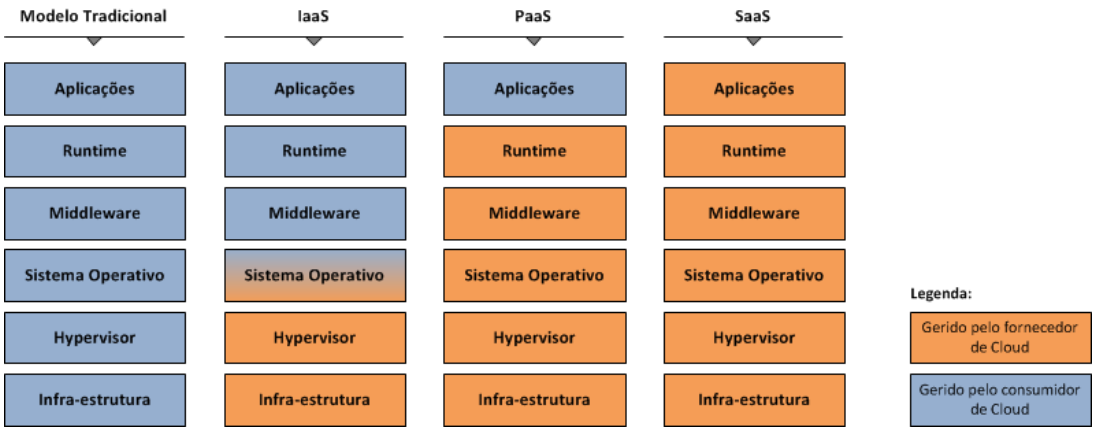

Figura 1 − Cloud Service Models

- *IaaS* − Consiste na disponibilização de unidades físicas tais como servidores, processamento, armazenamento e rede onde o utilizador pode instalar e correr software inclusive o sistema operativo;
- *PaaS* − Consiste na disponibilização de uma plataforma onde o utilizador pode instalar aplicações que se executam em ambientes suportados pelo fornecedor, como por exemplos aplicações baseadas em .Net ou Java. O utilizador apenas controla configurações das aplicações que executa, não tem qualquer responsabilidade de gestão do servidor, sistema operativo, armazenamento ou rede;

- *SaaS* − Consiste na disponibilização de aplicações executadas num ambiente *Cloud*. Estas aplicações são disponibilizadas para clientes tais como *web browsers* ou a interface de um programa instalado na máquina do cliente.

Por último importa ainda referir que este modelo pode ser disponibilizado em diferentes formas:

- *Cloud* Privada − é disponibilizada exclusivamente para uma organização, ainda que possa abranger várias unidade de negócio. Esta pode ser gerida pela própria organização, por terceiros ou por uma combinação de ambos;
- *Cloud* de comunidade− é disponibilizada para uma comunidade de organizações que possuam algo em comum. Pode ser gerida por uma ou várias organizações da comunidade, por terceiros ou por uma combinação destes;
- *Cloud* pública − é disponibilizada para o público em geral e pode ser gerida por uma organização empresarial, académica, governamental ou uma combinação destes;
- *Cloud* hibrida − é uma composição de duas ou mais modelos de disponibilização apresentados anteriormente.

2.2 Armazenamento de dados na *Cloud*

O armazenamento de dados na *Cloud* oferece ao utilizador um espaço de armazenamento remoto flexível e fiável. Este permite o acesso através da rede a um espaço de armazenamento de grande capacidade, sempre disponível e com a capacidade de crescer à medida das necessidades do utilizador. Comparado com o armazenamento em rede tradicional, o armazenamento na *Cloud* é um sistema complexo de *hardware*, equipamentos de rede, equipamentos de armazenamento, servidores, aplicações, entre outros [9].

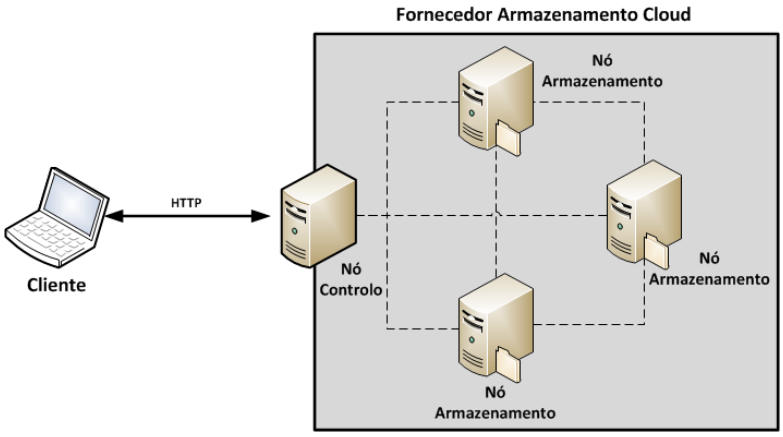

Figura 2 − Arquitetura típica de sistema de armazenamento na *Cloud*

Na Figura 2 é apresentado uma arquitetura típica de um sistema de armazenamento na *Cloud*, em que o armazenamento é efetuado em múltiplos servidores ao invés de num único, como acontece com o armazenamento em rede tradicional. O utilizador vê um servidor virtual, ou seja é criada a ilusão que os seus dados estão a ser armazenados num sitio particular, enquanto que os seus dados podem estar armazenados ou particionados em qualquer um dos servidores que compõem a *Cloud*. Na realidade os dados podem ser movidos entre servidores regularmente à medida que o sistema gere e otimiza o espaço disponível nos seus servidores.

Estes sistemas apresentam vantagens ao nível da gestão, custos, e na prevenção de desastres:

- Facilidade de gestão − Para o utilizador a gestão de um serviço de armazenamento na *Cloud* é mais simples de instalar e manter do que um sistema equivalente instalado *on-permises*[2]. Tipicamente apenas necessita de um *web browser* para aceder e gerir o seu espaço de armazenamento;
- Controlo de custos − O custo das subscrições de armazenamento na *Cloud* são, geralmente, inferiores ao custo hardware e do pessoal técnico para manter uma infraestrutura equivalente;
- Alta disponibilidade e prevenção de desastres − Os sistemas de armazenamento na *Cloud* têm redundância no armazenamento dos dados, o que significa que se existir uma falha no *hardware* o sistema pode continuar a funcionar e não serão perdidos dados;

[2] Conceito que refere um sistema instalado e gerido nas instalações do utilizador

Por outro lado a adoção destes sistemas por parte dos utilizadores também apresentam alguns receios e desafios, tais como:

- Controlo sobre os dados− Uma vez que os dados estão alojados num servidor remoto a perda do controlo dos dados é um receio que existe por parte dos utilizadores;
- Interoperabilidade − Cada fornecedor tem diferentes métodos (API ou aplicações) para acesso ao espaço de armazenamento. A falta de protocolos standard para acesso aos serviços de armazenamento leva a que não exista interoperabilidade entre diferentes sistemas;
- Performance − O acesso aos dados está limitado pelo *throughput* e latência da rede, e apesar dos últimos avanços na velocidade da internet, não é possível acompanhar a performance de um armazenamento tradicional de rede;
- Segurança − Em todos os sistemas apresenta-se como um problema e no armazenamento de dados na *Cloud* não é exceção, qualquer falha no sistema ou na transferência de dados pode comprometer a segurança dos dados.

2.3 Serviços de Armazenamento na *Cloud*

Atualmente existem inúmeras soluções que oferecem serviços de armazenamento na *Cloud*. Estas soluções dividem-se normalmente em dois grupos: as soluções empresarias e as soluções direcionadas para o público em geral.

No contexto das soluções empresariais podem-se encontrar, por exemplo, Amazon Simple Storage Service (Amazon S3) [10], Windows Azure Storage [11], Google Cloud Storage [12], IBM Smart Business Storage Cloud [13], entre outros. Estes têm como objetivo suportar requisitos empresarias como o armazenamento de dados aplicacionais ou o armazenamento de backups. Este sistemas, normalmente, têm uma interface direcionada para recursos com conhecimento em tecnologias de informação.

As soluções como o Dropbox, Google Drive ou SkyDrive [14] são serviços de armazenamento na *Cloud* direcionados para o público em geral. Fornecem um espaço de armazenamento na *Cloud* e geralmente, disponibilizam uma quantidade de espaço de forma gratuita. Apresentam uma interface de acesso *user friendly* em que o utilizador pode fazer a gestão destes sistemas de ficheiros. Por outro lado são também

muito utilizados para outras aplicações guardarem os dados dos utilizadores, desde que estes associem as suas contas. Como por exemplo aplicações para s*martphones* que permitem enviar as fotografias tiradas para uma pasta no Dropbox, ou certas aplicações da loja de aplicações Google Chrome que permitem guardar os dados diretamente no Google Drive.

No contexto das soluções para o público em geral existem também sistemas que enfatizam os aspetos sociais e partilha de conteúdos, mas também podem ser considerados serviços que armazenam conteúdos na *Cloud*. Neste segmento existem soluções tais como o Youtube para armazenamento e partilha de vídeos, o Picasa ou o Flicker para organização e partilha de imagens.

2.4 Integração de serviços de armazenamento na *Cloud*

Numa análise aos sistemas existentes no mercado com objetivos semelhantes aos deste projeto, foram identificados dois sistemas com relevância e que importam referir: Otixo [15] e CloudKafe [16]. Nestes sistemas é possível gerir os ficheiros dos serviços de armazenamento na *Cloud* que este projeto tem como alvo. No entanto ambos apresentam limitações, ao nível do sistema de pesquisa de ficheiros, em que apenas é possível pesquisar por nome de ficheiro. E também apresentam limitações ao nível da interface que está adaptada apenas para computadores, o que faz que utilizadores de *tablets* ou *smartphones* não se deparem com interfaces adaptadas para as suas plataformas.

Capítulo 3

Modelo Proposto

Neste capítulo é apresentado o modelo proposto para a solução e são detalhados os requisitos do sistema. A Figura 3 a presenta um diagrama que ilustra o modelo proposto para a solução do problema. O utilizador interage com o sistema, designado por Cloud Cockpit, que por sua vez executa as operações necessárias nos serviços de armazenamento na *Cloud* do utilizador através das API disponibilizadas pelos mesmos.

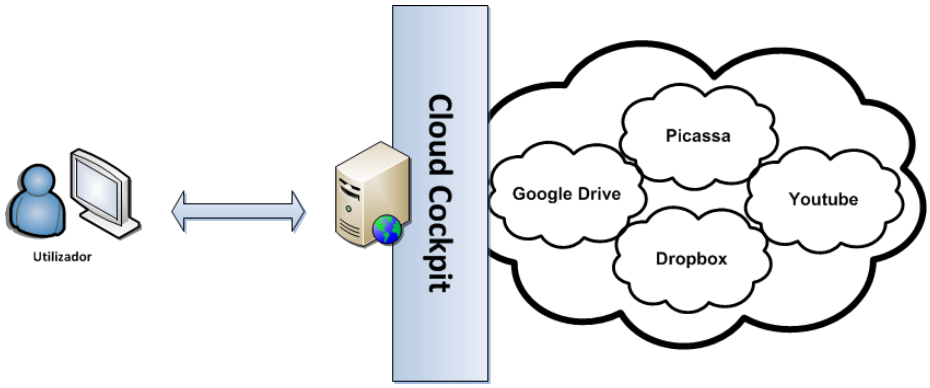

Figura 3 − Arquitetura do modelo proposto

3.1 Requisitos

Uma solução desta natureza apresenta um vasto conjunto de funcionalidades interessantes de disponibilizar, bem como uma série de serviços de armazenamento na *Cloud* a ser integrados. Assim foi necessário definir um conjunto de funcionalidades e de serviços a integrar, que fosse exequível (no tempo deste projeto). Neste contexto definiram-se os seguintes requisitos principais para o sistema, apresentando-se como requisitos mínimos para validação do conceito da solução:

1. Disponibilização das operações de escrita, leitura e cópia de ficheiros nos serviços Dropbox, e Google Drive;
2. Disponibilização de uma interface Web que se adapte a ecrãs de computadores, *tablets* e *smartphones*;
3. Disponibilização de pesquisa sobre os conteúdos não estruturados que se encontrem alojados nos serviços definidos no ponto 1.

Como objetivos secundários, no contexto deste projeto, identificam-se funcionalidades tais como registo, login, login através de *social login providers* e ainda a integração com serviços Youtube e Picasa.

3.2 Modelo de Domínio

A Figura 4 apresenta o modelo do domínio do sistema com a identificação dos atores do mesmo:

- Utilizador - O utilizador do sistema que utilizará um *browser* para interagir com o sistema;
- Social Login Providers - Sistemas externos que fornecem serviços de autenticação dos utilizadores. Especificamente os serviços do Google e Facebook;
- Cloud Storage Services - Sistemas que fornecem serviços de armazenamento na *Cloud*. Estes serviços disponibilizam API's com os quais o sistema Cloud Cockpit interage.

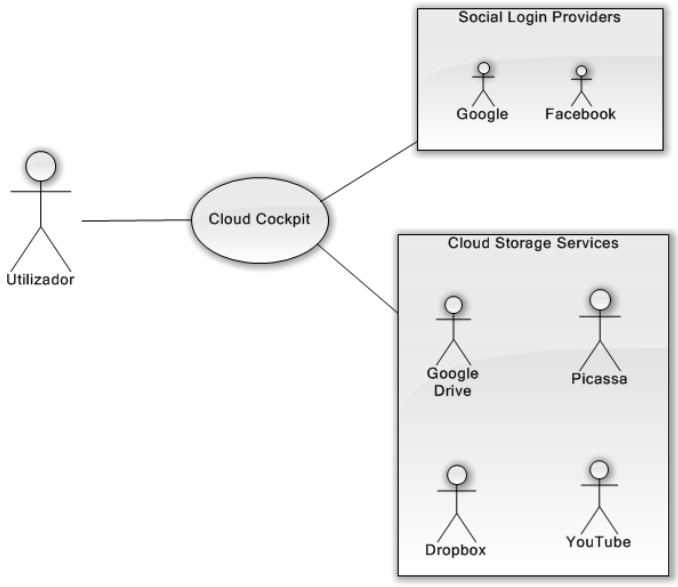

Figura 4 - Modelo de Domínio

3.3 Diagrama de casos de utilização

A Figura 5 apresenta o diagrama de casos de utilização do sistema, identificando as funcionalidades do sistema.

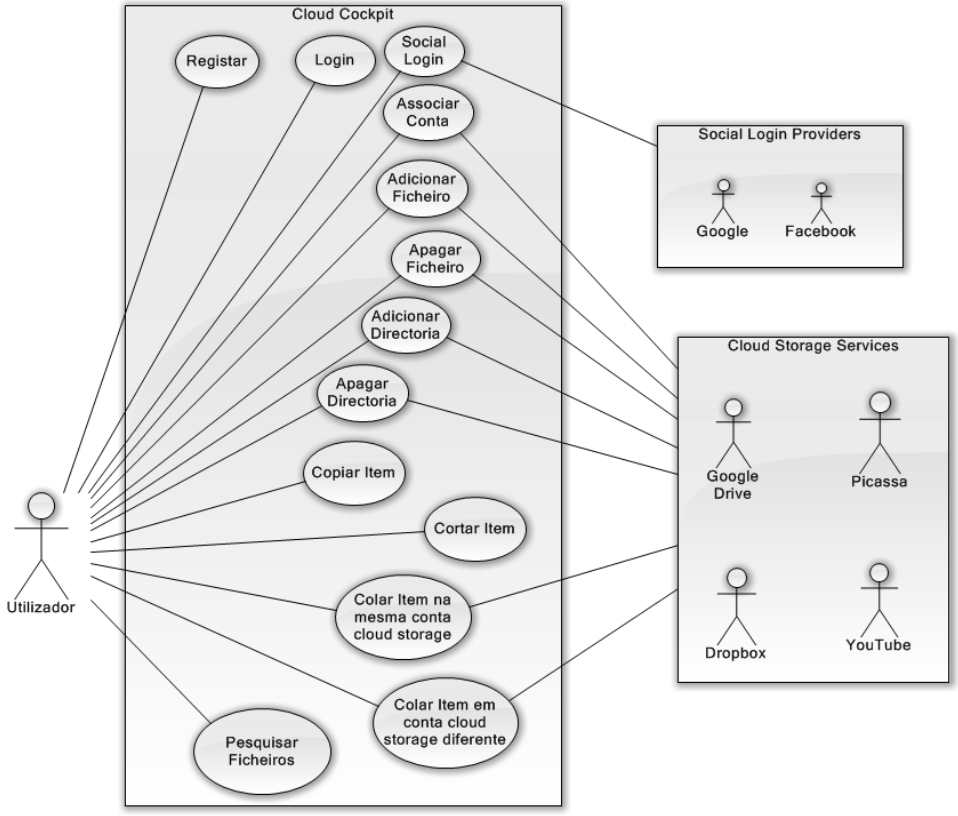

Figura 5 − Diagrama de casos de utilização

Os casos de utilização do sistema podem ser divididos em três grupos: associação de contas, gestão de conteúdos e pesquisa de conteúdos. Da observação do diagrama anterior enumeram-se os casos de utilização:

- UC1 - Registar
- UC2 - Login
- UC3 - Social Login
- UC4 - Associação de contas de *cloud storage*
- UC5 - Adicionar Ficheiro
- UC6 - Apagar Ficheiro
- UC7 - Adicionar Diretoria
- UC8 - Apagar Diretoria
- UC9 - Copiar Item
- UC10 - Cortar Item
- UC11 - Colar Item na mesma conta de *cloud storage*

- UC12 - Colar Item em conta de *cloud storage* diferente
- UC13 - Pesquisar de Ficheiros

3.4 Especificação de casos de utilização

De seguida são detalhados os casos de utilização do sistema que se entendem por mais relevantes e que apresentam um conjunto de operações que caracterizam o essencial da solução. Os restantes casos de utilização do sistema encontram-se em anexo.

A tabela seguinte apresenta a terminologia utilizada na especificação de casos de utilização que se apresenta a seguir.

Termo	**Definição**
Cloud Storage Service	Sistemas que fornecem serviços de armazenamento na *Cloud*
Conta de Cloud Storage Service	Conta de um utilizador num serviço de armazenamento na *Cloud*.
Social Login Provider	Serviço que fornece a operação de autenticação (no contexto deste sistema Google e Facebook)
Ficheiro	Qualquer tipo de ficheiro que possa estar alojado nos serviços de armazenamento na *Cloud*.
Diretoria	Agregador de ficheiros.
Item	Ficheiro ou Diretoria (incluindo todos os ficheiros que estão agregados).

UC4 – Associação de contas de Cloud Storage

Cenário	Associação de uma conta de armazenamento na *Cloud* à área de trabalho do utilizador.
Evento Inicial	Na sua área de trabalho o utilizador escolhe adicionar conta de armazenamento na *Cloud*.
Atores	Utilizador, Cloud Storage Service
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3 (em anexo).
Pós-Condição	Uma nova conta de armazenamento na *Cloud* é adicionada à área de trabalho do utilizador

<table>
<tr><th colspan="3">Fluxo de Eventos</th></tr>
<tr><th>Ator</th><th>Sistema</th><th>Cloud Storage Service</th></tr>
<tr><td>1. Seleciona adicionar nova conta armazenamento na Cloud;
2. Escolhe um dos serviços disponíveis.</td><td></td><td></td></tr>
<tr><td></td><td>3. Faz um pedido de ao serviço com indicação do que pretende controlar;
4. Redireciona o utilizador para a página do serviço para dar permissões de acesso.</td><td></td></tr>
<tr><td></td><td></td><td>5. Apresenta a página para o utilizador se autenticar.</td></tr>
<tr><td>6. Autentica-se no serviço;
7. Aceita que o sistema Cloud Cockpit controle a sua conta.</td><td></td><td></td></tr>
<tr><td></td><td></td><td>8. Retorna a informação necessária para que o sistema aceda ao serviço em nome do utilizador.</td></tr>
<tr><td></td><td>9. Guarda os dados necessários para aceder ao serviço em nome do utilizador;
10. Associa uma nova conta na área de trabalho do utilizador.</td><td></td></tr>
<tr><th colspan="3">Exceções</th></tr>
<tr><td colspan="3">• No passo 5, se o utilizador não se autenticar o caso de utilização termina com uma mensagem de falha ao associar;
• No passo 6, se o utilizador não aceitar o caso de utilização termina com uma mensagem de falha ao associar;</td></tr>
</table>

UC5 – Adicionar Ficheiro

Cenário	Adicionar ficheiro numa conta de um serviço de armazenamento na *Cloud*.
Evento Inicial	O utilizador seleciona um ficheiro para adicionar e um destino numa conta de armazenamento na *Cloud*.
Atores	Utilizador, Cloud Storage Service
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de UC2 ou UC3(em anexo). O utilizador associou a conta anteriormente através do caso de utilização UC4.
Pós-Condição	O ficheiro selecionado é adicionado à conta de armazenamento na *Cloud*.

Fluxo de Eventos		
Ator	**Sistema**	**Cloud Storage Service**
2. Seleciona um ficheiro e um destino para o ficheiro.		
	1. Valida que o tipo de ficheiro a carregar é suportado pelo Cloud Storage Service de destino; 2. Faz o pedido de carregamento do ficheiro ao Cloud Storage Service de destino; 3. Enquanto o carregamento não terminar notifica o utilizador do progresso desta operação.	
		4. Carrega o ficheiro; 5. Confirma que o ficheiro foi adicionado com sucesso.
	6. Notifica o utilizador de carregamento efetuado com sucesso.	
Exceções		

• Se o carregamento do ficheiro falhar o utilizador é notificado e o caso de utilização termina.

UC12 – Colar Item em conta de Cloud Storage diferente

Cenário	Colar um ficheiro ou diretoria da área de transferência para um destino numa conta de um serviço armazenamento na *Cloud* diferente da origem.
Evento Inicial	O utilizador escolhe o destino do item e a opção “Colar”.
Atores	Utilizador, Cloud Storage Service de Origem, Cloud Storage Service de Destino
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3 (em anexo). O utilizador associou a conta anteriormente através do caso de utilização UC4. O utilizador colocou um item na área de transferência do sistema através do caso de utilização UC9 ou UC10 (em anexo).
Pós-Condição	O item é colado no destino. Fica alojado apenas no destino se estava marcado para cortar. Se estava marcado para copiar fica alojado fisicamente na origem e no destino.

Fluxo de Eventos			
Ator	**Sistema**	**Cloud Storage Service Origem**	**Cloud Storage Service Destino**
1. O utilizador escolhe o destino do item e a opção “Colar”.			
	2. Verifica se a origem e o destino se encontram em contas de armazenamento na *Cloud* diferentes; 3. Verifica se o tipo do item é suportado na		

	localização de destino 4. Descarrega o item da localização de origem.		
		5. Envia o item pedido	
	6. Faz o pedido à localização de destino para carregar o item.		
			7. Carrega o item; 8. Confirma que o item foi adicionado com sucesso.
	9. Faz o pedido à localização de origem para apagar o item;		
		10. Apaga o item de origem; 11. Confirma que o item foi apagado com sucesso.	
	12. Retira o item da área de transferência; 13. Notifica o utilizador que o item foi colado com sucesso.		

Cenário alternativo 1
• No passo 4, se o item estiver marcado para copiar, o caso de utilização executa o passo 8 e de seguida continua o passo 12.
Exceções
• No passo 4 se a localização de origem for é na mesma conta de armazenamento na *Cloud* da localização de destino, é invocado o caso de utilização UC11 (em anexo) no passo 4; • No passo 4 se o tipo de item não é suportado no serviço de armazenamento na *Cloud* de

destino, o utilizador é informado e o caso de utilização termina;

- No passo 11 se não for possível apagar o item, o utilizador deve ser informada que a cópia foi realizada, mas o item não foi apagado da origem;
- Se o colar do item falhar o utilizador é notificado e o caso de utilização termina.

3.5 Abordagem

A arquitectura que se encontra representada no diagrama de blocos da Figura 6 decompõe o sistema em módulos coesos e desacopolados, com o intuito de separar as caracteristicas essenciais e consequentemente garantir o nivel de abstracção entre módulos, confinando-lhe funcionalidades concretas e facilitando a sua integração.

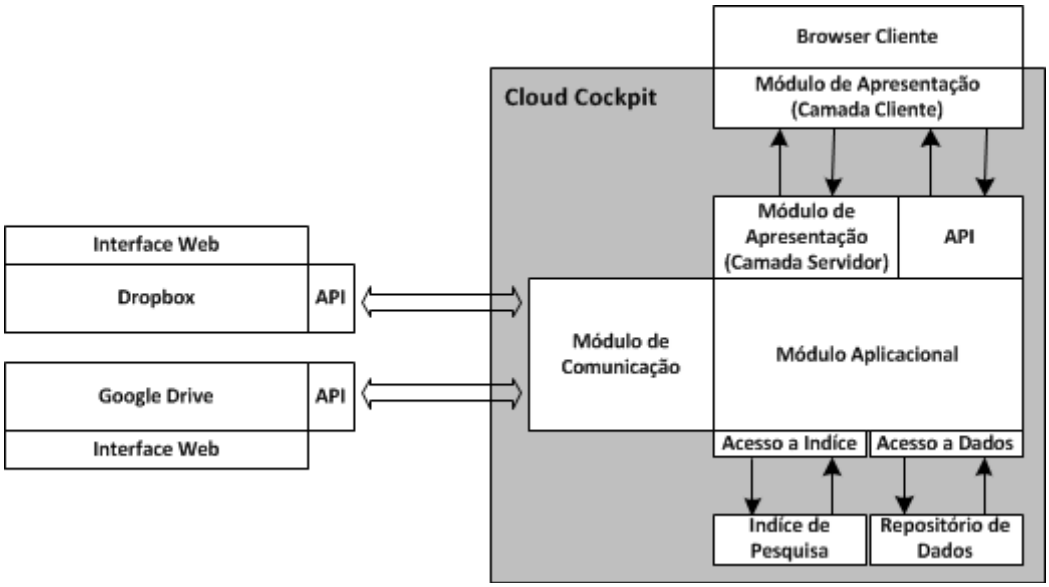

Figura 6 – Diagrama de blocos do sistema

O sistema armazena, no repositório de dados, as informações dos utilizadores do sistema. Estes dados contemplam a informação necessária para fazer a ligação com as contas de armazenamento na *Cloud* que cada utilizador associa ao sistema. Note-se que a informação armazenada é a estritamente necessária para efetuar as ligações. O sistema não armazena nomes de utilizador ou palavras chaves para acesso aos referidos serviços.

No contexto da recuperação de informação o sistema utiliza índices que armazenam a informação dos conteúdos não estruturados. Para tal é necessário um processo que

permita o *crawling* e indexação dos conteúdos alojados nos serviços de armazenamento na *Cloud*. Os índices de utilizadores diferentes estão fisicamente separados para garantir o isolamento da informação do utilizador.

O módulo aplicacional efetua a integração entre os diferentes componentes do sistema, bem como a orquestração das operações nos serviços de armazenamento na *Cloud*.

Para efetuar a comunicação com os serviços de armazenamento na *Cloud*, o sistema recorre ao módulo de comunicação. Este contempla a implementação especifica para ligação às API disponibilizadas pelos serviços. Importa salientar que cada serviço, para além de se apresentar em *endpoints* diferentes, também utiliza representações de dados diferentes. Pelo que neste módulo torna-se importante garantir a abstração destes aspetos ao módulo aplicacional. Em suma, o módulo de comunicação lida com a comunicação, autenticação e mapeamento de dados de cada serviço de modo a facultar ao módulo aplicacional uma imagem homogénea de todos os serviços.

A interface com o utilizador encontra-se limitada ao módulo de apresentação, que tem como alvo um navegador Web. Esta camada encontra-se tanto no servidor para servir todos os recursos necessários ao funcionamento de uma aplicação Web, como também no cliente de forma a potenciar a experiência de utilização e usabilidade da aplicação.

Por último destaca-se também uma camada de API. Esta disponibiliza o conjunto de operações necessárias para implementar os requisitos do sistema, e é utilizada pela camada de apresentação no cliente. O objetivo desta API é permitir que o núcleo do sistema seja reutilizado por outras aplicações futuramente. Como por exemplo por numa aplicação para dispositivos móveis, ou mesmo por outras aplicações que necessitem de se abstrair da pluralidade de ofertas de serviços de armazenamento na *Cloud* (embora este último cenário se encontre fora do âmbito deste projeto).

Capítulo 4

Implementação do Modelo de dados

O modelo de dados do sistema visa dar suporte à informação dos utilizadores do sistema, a forma de acesso e as contas de serviços armazenamento na *Cloud* que cada utilizador tem associadas, incluindo a informação necessária para efetuar a ligação a essas mesmas contas.

A Figura 7 apresenta o modelo entidade associção do sistema que foi elaborado de forma a dar suporte aos requisitos. O modelo de dados é composto pelas quatro tabelas que se descrevem de seguida:

- *CloudStorageAccount* – Tabela que guarda a informação de contas de armazenamento na Cloud que foram adicionadas ao sistema. Cada conta tem um utilizador associado, um nome definido pelo utlizador e um conjunto de informação relacionada com o protocolo de ligação (OAuth) às API dos serviços. Este componente é apresentado no Capítulo 6;
- *UserAccount* – Tabela que armazena a informação dos utilizadores do sistema. Cada utilizador tem um nome de utilizador, e uma forma de efectuar login, seja por nome de utilizador e palavra-chave, ou seja através de um serviço de Social Login;
- *CloudStorageAccountType* – Tabela para parametrizar os tipos de contas que podem ser adicionadas ao sistema;
- *LoginType* – Tabela para parametrizar os tipos de *logins* que são aceites pelo sistema.

No diagrama do modelo entidade associação e no modelo lógico são utilizadas as seguintes convenções:

- Chaves primárias a sublinhado;
- Chaves candidatas com o prefixo <ccN>, em que N representa o número da chave candidata.

No modelo lógico é também utilizado o prefixo [FK] para identificação das chaves estrangeiras.

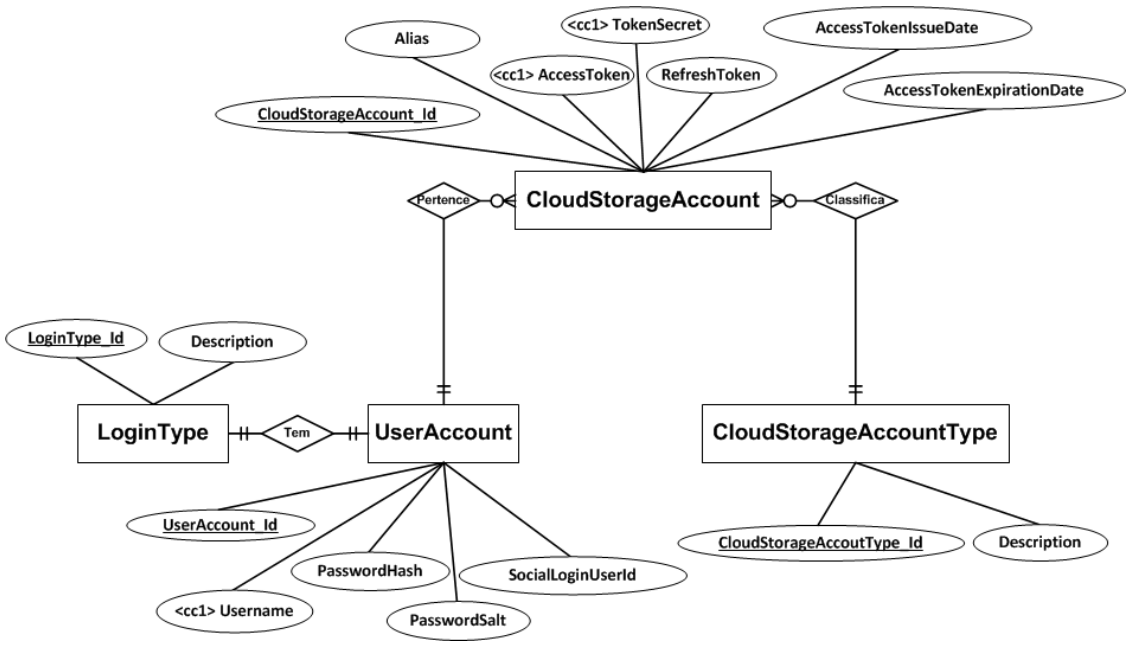

Figura 7 − Modelo Entidade Associação

O modelo de dados é suportado por uma base de dados relacional cuja definição do modelo lógico que se apresenta a seguir.

```
 CloudStorageAccount( CloudStorageAccount_Id, <cc1> CloudStorageAccountType_Id[FK](1)
       UserAccount_Id[FK](2), Alias, <cc1> AccessToken, <cc1> TokenSecret,
       RefreshToken,AccessTokenIssueDate,   AccessTokenExpirationDate )

UserAccount( UserAccount_Id, LoginType_Id[FK](3), <cc1> Username, PasswordHash,
       PasswordSalt,  SocialLoginUserId )

CloudStorageAccountType( CloudStorageAccountType_Id, Description )

LoginType( LoginType_Id, Description )

(1)CloudStorageAccountType_Id referencia CloudStorageAccountType.CloudStorageAccountType

(2)UserAccount_Id referencia UserAccount.UserAccount_Id

(3)LoginType_Id referencia LoginType.LoginType_Id
```

Listagem 1− Modelo Lógico

Para a concretização do repositório de dados foi desenvolvida uma base de dados suportada pelo Sistema de Gestão de Base de Dados (SGBD) *SQL Server 2008*. A escolha deste SGBD está relacionada com a abordagem deste projecto quanto à utilização de tecnologias Microsoft (plataforma .NET). A maior limitação desta decisão poderia prende-se com a necessidade de aquisição (e manutenção) de licenças para utilização. No entanto existem versões designadas por *express* que, embora com algumas restrições, permitem utilizar este SGBD de forma gratuita. Outros produtos poderiam ter sido utilizados, como Oracle ou MySql, mas o nível de integração do SQL Server com a plataforma .NET é superior quando comparado com qualquer outra tecnologia, nomeadamente a integração com a tecnologia de acesso a dados ADO.NET Entity Framework recomendada pela Microsoft [17].

Capítulo 5

Implementação do Módulo Aplicacional

Neste capitulo são apresentados os detalhes de implementação da camada de acesso a dados, o acesso ao índice de pesquisa e a execução de operações nas contas de armazenamento na *Cloud*.

5.1 Acesso a dados

Para implementação da camada de acesso a dados foi utilizada à tecnologia ADO.NET Entity Framenwork 5.0. Esta tecnologia permite efetuar o mapeamento entre Objetos e o Modelo Relacional (ORM). A escolha desta tecnologia justifica-se com a necessidade de aumentar a produtividade de implementação desta camada bem como ser esta a tecnologia recomendada pela Microsoft.

Para implementação desta camada foi necessário mapear cada uma das tabelas em objetos através da implementação de classes de mapeamento da Entity Framework. Definiu-se o conjunto de operações de manipulação de dados necessário para cada tabela, e implementou-se um padrão de repositório genérico [18] que implementa as operações base para manipulação de todas as tabelas. Cada objeto que estende este repositório pode implementar operações especificas de manipulação da tabela respetiva. O diagrama seguinte representa um exemplo de utilização com o repositório de acesso à tabela *CloudStorageAccount*.

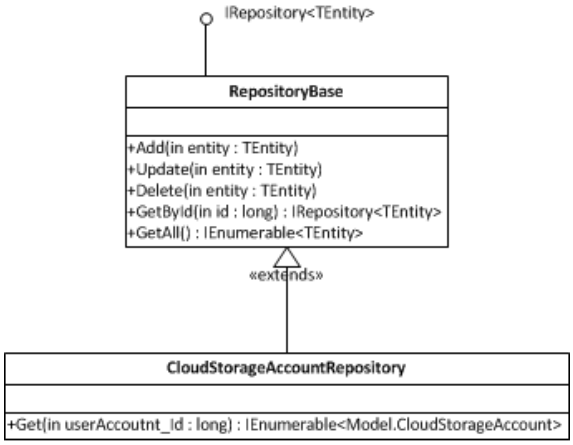

Figura 8 – Implementação do padrão repositório em CloudStorageAccount

5.2 Acesso ao índice de pesquisa

O acesso ao índice de pesquisa é realiza-se através de uma interface que foi definida de forma a dar suporte a todas as operações necessárias. A figura seguinte apresenta esta interface de acesso, bem como a implementação do padrão *factory method* [19] que foi utilizado para desacoplar a camada aplicacional da implementação especifica do índice de conteúdos não estruturados. O detalhe de implementação do índice e algoritmos de *crawling* é apresentado no Capítulo 7.

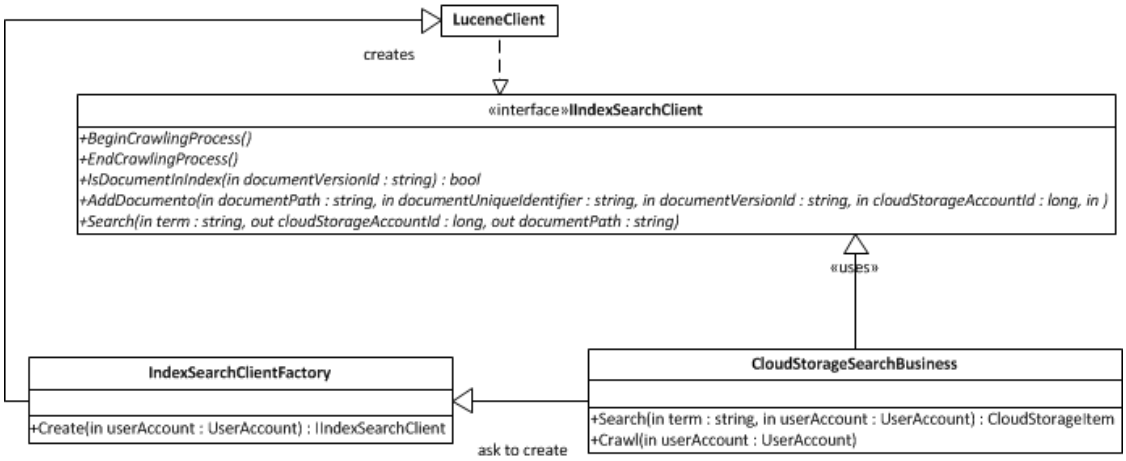

Figura 9 – Camada de acesso ao índice

A classe de efetua a lógica de negócio da pesquisa *CloudStorageSearchBusiness* utiliza a classe *IndexSearchFactory* para criar objetos com a interface *IIndexSearchClient* que contém todos os métodos necessários para acesso ao índice de pesquisa.

5.3 Execução de operações nos serviços de armazenamento

O módulo aplicacional é responsável por coordenar as operações que são efetuadas nos serviços de armazenamento na *Cloud*. Com a finalidade de desacoplar o módulo aplicacional dos detalhes de implementação da ligação com cada serviço de armazenamento na *Cloud*, este módulo apenas depende da interface *ICloudServiceClient*. Esta interface disponibiliza todos os métodos necessários para que o módulo aplicacional efetue as operações que necessita. A implementação de cada cliente de ligação aos serviços específicos é delegada no módulo de comunicação, apresentado no Capítulo 6.

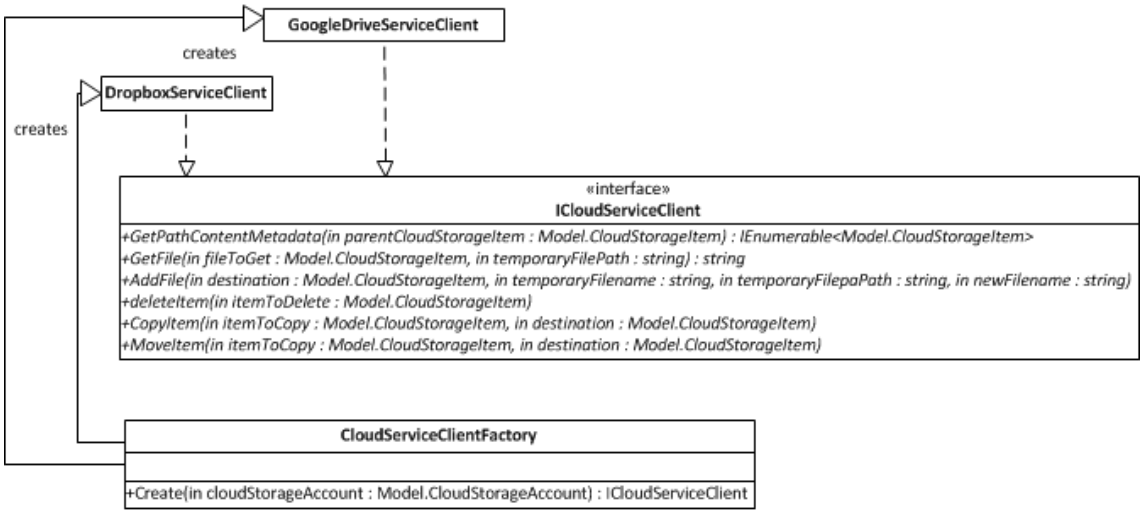

Figura 10 – Diagrama da interface ICloudServiceClient

A Figura 10 apresenta a referida interface, esta contém métodos para navegar em pastas, descarregar ficheiros, adicionar ficheiros, remover itens, copiar e mover itens. A camada aplicacional obtém os objetos de ligação aos serviços de armazenamento através da invocação de um método *factory* na classe *CloudServiceFactory*. Este, dependendo do tipo de conta, retorna um objeto para gerir a ligação a um serviço especifico (*DropboxServiceClient* ou *GoogleDriveServiceClient*).

Esta abordagem permite que a utilização de novos serviços de armazenamento com o módulo aplicacional seja transparente, bastando apenas implementar uma classe que implemente *ICloudServiceClient* e a alteração do método *factory* para criação de clientes para o novo serviço.

Nas seguintes duas imagens são apresentados dois diagramas de sequência com o fluxo de operações para exemplificar a coordenação efetuada pelo módulo aplicacional.

A Figura 11 representa o fluxo para adicionar um ficheiro num serviço de armazenamento na *Cloud*, que se passa a descrever:

1. O processo é iniciado quando o método de adicionar ficheiro do módulo aplicacional é invocado;
2. O ficheiro é guardado no servidor do sistema numa pasta temporária;
3. O cliente de acesso ao serviço é criado através da classe *CloudServiceFactory*;
4. O ficheiro é adicionado através da invocação do método *AddFile*, da interface *ICloudServiceClient*;
5. O último passo consiste em eliminar o ficheiro da pasta temporária.

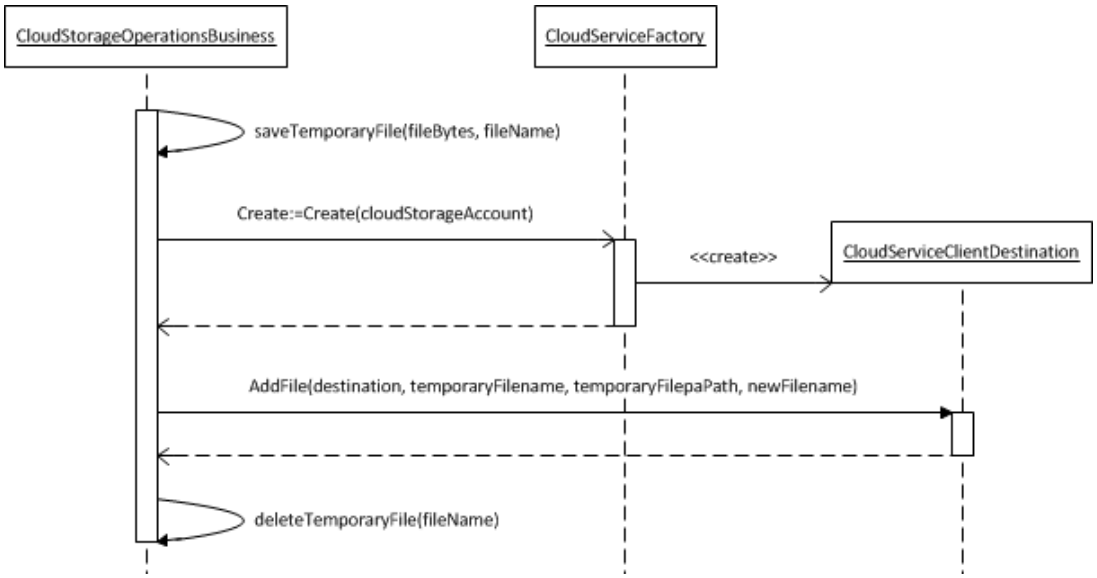

Figura 11 – Diagrama de sequência da operação de adição de ficheiro

Na Figura 12 é apresentado o fluxo para mover um ficheiro entre duas contas de dois serviços de armazenamento na *Cloud*. Note-se que se a operação fosse de mover um ficheiro dentro da mesma conta de armazenamento, este processo não seria necessário, uma vez que os serviços de armazenamento disponibilizam operações com este objetivo. Assim a operação de mover um ficheiro consiste em:

1. Os clientes de acesso aos serviços, origem e destino, são criados através da classe *CloudServiceFactory*;
2. O ficheiro é descarregado do serviço de origem e guardado numa pasta temporária do servidor;
3. O ficheiro é adicionado ao serviço de destino, invocando o método *AddFile* da interface *ICloudServiceClient*;
4. O ficheiro é eliminado do serviço de origem, através da invocação do método *deleteItem* da interface *ICloudServiceClient*;
5. Por último o ficheiro é eliminado da pasta temporária.

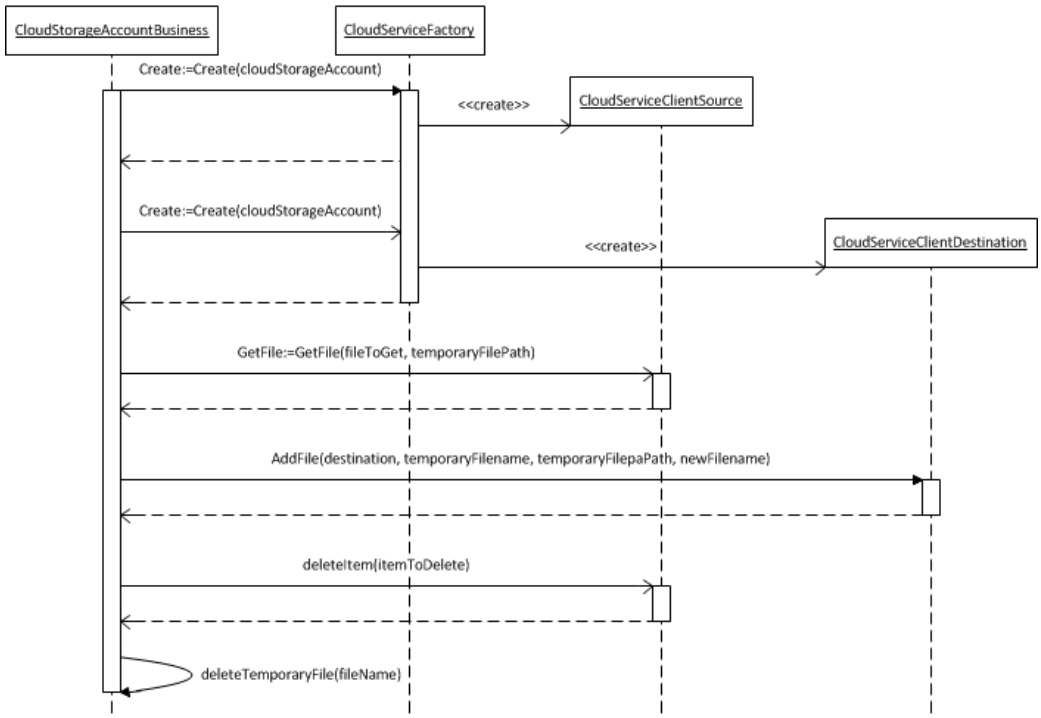

Figura 12 – Diagrama de sequência da operação de mover um ficheiro

Note-se que todas as operações em que é necessário carregar ou descarregar ficheiros implicam que estes sejam guardados numa pasta temporária do servidor. Isto é possível pois o sistema Cloud Cockpit tem permissão para aceder às contas dos utilizadores (ver ponto 6.1). Não é possível transferir os ficheiros diretamente entre os dois serviços porque não é possível fazer com que estes comuniquem entre si.

Capítulo 6

Implementação do Módulo de Comunicação

Neste capitulo são apresentados os detalhes do modulo de comunicação, tais como a forma de associação de contas de armazenamento na *Cloud* com o sistema, a autenticação e comunicação com os serviços de armazenamento.

6.1 Associação de Contas de armazenamento na *Cloud*

O sistema interage com os serviços de armazenamento na *Cloud* através de chamadas às API (serviços HTTP) disponibilizadas. Para tal, o sistema autentica-se nos serviços em nome dos proprietários das contas, ainda que para este processo de autenticação o sistema não necessite de consultar nem armazenar os dados de autenticação dos utilizadores.

O sistema faz uso dos mecanismos de autenticação suportados pelos serviços de armazenamento na *Cloud*, que autenticam aplicações de terceiros para executar operações através da API. Estes mecanismos de autenticação são baseados nos standards OAuth [20] e OAuth 2.0 [21].

O OAuth define-se como um protocolo, e o OAuth 2.0 como uma *framework*. Ambos fornecem um método de acesso a recursos por parte de aplicações de terceiros em nome do proprietário, sem necessidade de partilha das credenciais de autenticação. O OAuth 2.0 é uma evolução do OAuth tendo como principais objetivos: melhorar o suporte a aplicações que não sejam executadas num browser, remover a necessidade de criptografia dos clientes, reduzir a complexidade das assinaturas das mensagens e a separação clara entre o servidor que executa a autenticação e o servidor que responsável pelo tratamento dos pedidos OAuth [22].

A Figura 13 apresenta um fluxo exemplificativo do processo de associação de uma conta de armazenamento na *Cloud* com o sistema. Note-se que a implementação detalhada do protocolo de autenticação (OAuth ou OAuth 2.0) não é o objetivo deste diagrama.

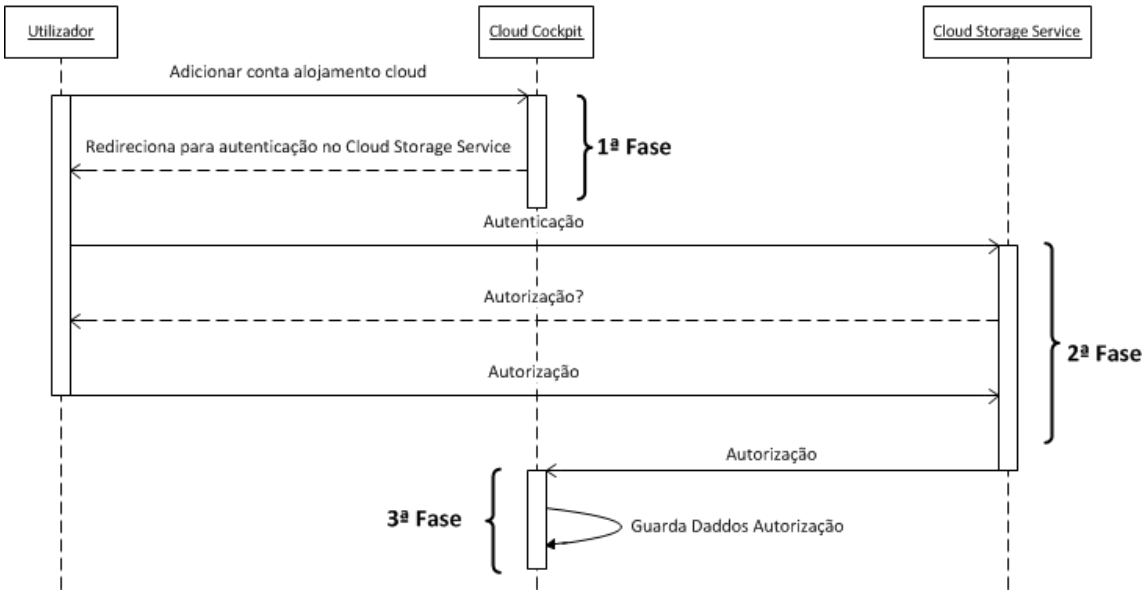

Figura 13 – Diagrama de fluxo de associação de contas de armazenamento

Na primeira fase o sistema recebe a informação que o utilizador deseja adicionar uma conta de armazenamento na *Cloud*, e dá a indicação ao browser cliente para redirecionar para a página de autenticação do serviços de armazenamento (Dropbox ou Google Drive).

Na segunda fase é executado o protocolo de autenticação, o serviço autentica o utilizador através das suas credências de autenticação. De seguida apresenta a informação ao utilizador que o sistema Cloud Cockpit deseja aceder à API e que pretende ter permissões para aceder a todo o seu espaço de armazenamento. O utilizador tem a opção de aceitar ou de rejeitar. É nesta segunda fase que se executa o fluxo específico OAuth ou OAuth 2.0.

A terceira fase é despoletada após o utilizador aceitar ou rejeitar o pedido de permissão. Caso aceite o sistema guarda os dados necessários para que possa aceder à API em nome do proprietário da conta.

É ainda de referir que o utilizador apenas pode adicionar contas de armazenamento após efetuar login na aplicação.

6.1.1 Autenticação Dropbox

A API Dropbox utiliza como protocolo de autenticação o OAuth. Para que um sistema cliente (Cloud Cockpit) possa enviar um pedidos OAuth ao servidor (Dropbox) foi necessário um registo prévio na consola de aplicações Dropbox. Este registo faculta dois *tokens* essenciais para o protocolo OAuth o *ClientKey* e *ClientSecret.* Ressalve-se que este processo apenas é necessário efetuar pelo individuo que disponibiliza a aplicação para os utilizadores. Os utilizadores não necessitam de nenhum registo prévio na consola de aplicações do Dropbox. As chaves disponibilizadas são utilizadas para a API Dropbox reconhecer os pedidos vindos do sistema Cloud Cockpit.

O diagrama seguinte apresenta a estratégia utilizada para separar os detalhes de implementação do OAuth dos detalhes necessários para ligar a um serviço específico. A classe *OAuth1* implementa os métodos *RequestAuthorization* e *RequestAuthorizationCallback* que são necessários para associação de contas de armazenamento na *Cloud*. O método *GetAuthenticatedRequest* autentica os pedidos HTTP, com base nas chaves *AccessToken* e *TokenSecret* presentes no objeto *CloudStorageAccount* recebido por parâmetro e que foram obtidas no processo de associação da conta de armazenamento.

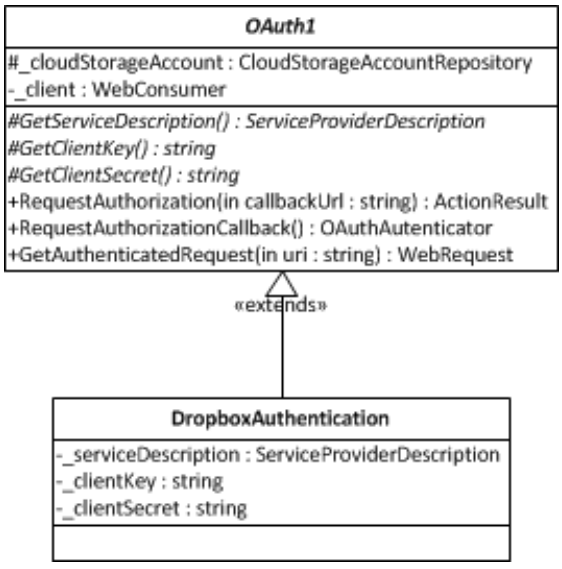

Figura 14 – Diagrama de classes OAuth1

A classe *DropboxAuthentication* fornece os *tokens* (obtidos através da consola de aplicações Dropbox) e respetivo endereço do serviço de autenticação. Deste modo é possível que o sistema suporte novos serviços baseados em OAuth, em que apenas é necessário implementar uma classe que derive de OAuth1 e implemente os métodos abstratos da mesma.

O fluxo de associação de conta especifico do protocolo OAuth foi implementado com recurso à *framework open source* DotNetOpenAuth [23].

6.1.2 Autenticação Google Drive

A API do Google Drive utiliza o mecanismo de OAuth 2.0. O processo para permitir efetuar pedidos OAuth 2.0 ao servidor (Google Drive), é semelhante ao anterior. Foi necessário o registo prévio na consola de aplicações da Google e em que se obtiveram dois *tokens* o *ClientIdentifier* e *ClientSecret*. Estes são utilizados pelo OAuth 2.0 para identificar o sistema Cloud Cockpit perante a API Google Drive.

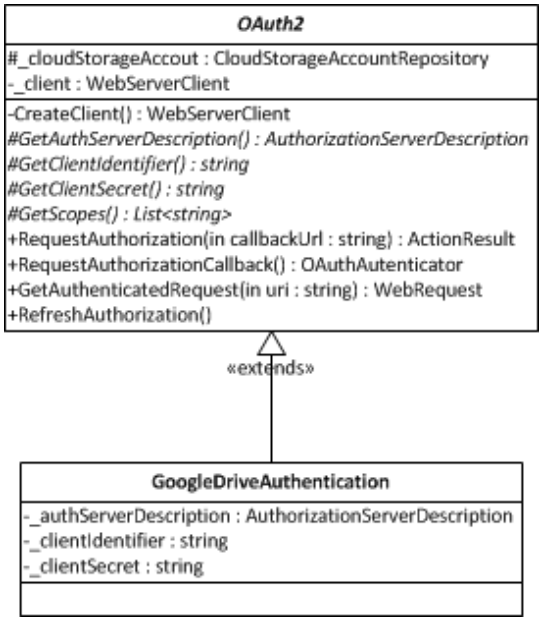

Figura 15 – Diagrama de classes OAuth2

A Figura 15 apresenta o diagrama de classes da estratégia utilizada para separar os detalhes de implementação do Auth 2.0 e dos detalhes de ligação com um serviço específico.

A classe *OAuth2* implementa os métodos *RequestAuthorization* e *RequestAuthorizationCallback* que são utilizados no processo de associação de contas Google Drive com o sistema. O método *GetAuthenticatedRequest* que permite autenticar pedidos HTTP, com base nas chaves obtidas no processo de registo com a consola de aplicações Google Drive. E por último o método *RefreshAuthorization* que permite ao sistema refrescar os *tokens* de autorização, uma vez que o protocolo *Auth 2.0* funciona com *tokens* de curta duração.

A classe *GoogleDriveAuthentication* fornece os *tokens* (obtidos através da consola de aplicações Google Drive) e o descritor do serviço de autenticação.

Este modelo também permite a adição de novos serviços baseados em *OAuth 2.0* sendo apenas necessário implementar uma classe que derive de *OAuth2*.

O fluxo de associação de conta específico do protocolo OAuth 2.0 foi implementado com recurso à *framework open source* DotNetOpenAuth.

6.2 Homogeneização de dados

Cada serviço de armazenamento de dados na *Cloud* representa um item (ficheiro ou pasta) de forma diferente. O sistema tem a necessidade de se abstrair desta questão, e por este motivo foi definida a classe *CloudStorageItem,* apresentada na Figura 16. A classe apresenta os atributos necessários para o funcionamento da aplicação. Cada cliente para os diferentes serviços (*DropboxServiceClient* e *GoogleDriveServiceClient*) utiliza os metadados fornecidos pelos serviços e transforma-os na classe apresentada, de forma a ser utilizada pela restante aplicação.

Esta classe contém atributos para descrição dos item tais como, o nome, o caminho, se é uma pasta, o tipo de conteúdo dos ficheiros e o tamanho. Contém também atributos direcionados para a pesquisa tal como o identificador da versão do item.

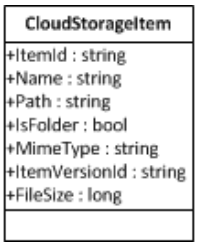

Figura 16 – Classe CloudStorageItem

Note-se que esta classe tem apenas o intuito de descrever os metadados do item. No caso de um ficheiro este será sempre tratado com um *array* de *bytes* e é apenas descarregado quando pedido. Pelo que a navegação nas pastas dos serviços de armazenamento na *Cloud* apenas descarrega os metadados e nunca o conteúdo dos ficheiros que, pode ser, de grande dimensão.

As definição de item do serviço Dropbox e do serviço Google Drive são diferentes, ainda que, se consiga em ambos retirar as informações base de ficheiro tal como nome, tamanho e tipo. Importa ainda referir que o a definição de item do Google Drive dá bastante enfâse a campos para ferramentas colaborativas, enquanto o Dropbox se apresenta como um gestor de ficheiros puro.

6.3 Comunicação com as API

A API disponibilizadas pelos serviços são baseadas na arquitetura REST que expõe os serviços através de pedidos HTTP [24]. Os dados são obtidos dos serviços no formato XML ou JSON. Para a comunicação é utilizada a classe *WebRequest* do *namespace System.Web* e a *framework Newtonsoft.Json* para serialização e leitura de dados no formato JSON.

Os *endpoints* utilizados para o Dropbox, bem como a definição dos parâmetros está disponível online na documentação da API [25].

Os *endpoints* utilizados para o Google Drive, e a definição dos parâmetros está disponível online na documentação da API [26].

De referir que o serviço Google Drive disponibiliza um SDK para C# enquanto que o serviço Dropbox não. A decisão de implementar o módulo de comunicação de raiz para todos os serviços, ao invés de utilizar o SDK nos serviços que estivesse

disponível, prende-se com o facto da idealização uma plataforma estruturada de acordo com os requisitos do sistema, sem a necessidade a dependência de SDK que para além de terem funcionalidades que não seriam usadas no sistema, são sempre código de terceiros em que não existe controlo. Esta abordagem permite também ter uma aplicação estruturada e modular que com, relativa, facilidade se integram novos serviços de armazenamento na *Cloud*.

Capítulo 7

Implementação da Pesquisa

Na implementação da pesquisa foi necessário perceber quais as soluções de recuperação de informação existentes que poderiam ser integradas com o sistema. Após uma pesquisa chegou-se à conclusão que as soluções para integração na plataforma .NET não são muitas. Uma alternativa é recorrer a uma solução como o Apache Solr [27] que é uma plataforma de pesquisa empresarial que disponibiliza um conjunto de funcionalidades relacionadas com recuperação de informação, entre elas a indexação e pesquisa. No entanto o Apache Solr necessita de ser instalado num servidor Web e é manipulada através de pedidos HTTP. Outra alternativa consiste na utilização de uma biblioteca de recuperação de informação que permite ser integrada no núcleo do sistema desenvolvido. Neste contexto foram analisados os ambientes Lucene.NET [28] (*port* para .NET do Apache Lucene implementado em Java) e o ambiente Xapian [29]. Na escolha da abordagem prezou-se a implementação do motor de *crawling* e indexação integrado na aplicação. Perante as bibliotecas analisadas optou-se pela utilização do Lucene uma vez que é uma solução mais sólida e adotada inclusivamente por outras soluções tais como o Apache Solr.

A solução de pesquisa implicou a implementação de um processo de *crawling* e indexação para a criação de um índice que posteriormente pode ser pesquisado.

Nos pontos seguintes são apresentados os detalhes relacionados com a indexação de ficheiros e com o algoritmo de *crawling*.

7.1 Indexação

O processo de indexação passa por inserir no índice de pesquisa um documento que é caracterizado por um conjunto de campos. Para este sistema foram definidos os seguintes campos para caracterizar cada documento:

- *DocumentVersionId* – identificador de versão do documento;
- *CloudStorageAccountID* – identificador da conta de armazenamento na *Cloud* à qual o documento pertence;
- *Content* – o conteúdo do ficheiro, onde será efetuada a pesquisa;
- *NoDelete* – campo para auxiliar o processo de *crawling*.

O campo *Content* é o único campo pesquisável do sistema, logo é necessário que no momento da indexação seja guardado o vetor de termos de forma a que possa ser executada uma pesquisa vetorial ao invés de um simples pesquisa booleana. A pesquisa vetorial permite ordenar os resultados em termos de relevância para a pesquisa efetuada, enquanto a pesquisa booleana apenas devolve os documentos que contêm o termo, não sendo possível executar qualquer tipo de ordenação de resultados.

É de referir que foi utilizado o analisador standard do Lucene, pelo que a lista de *steaming* e *stop words* utilizada é a standard, pelo que existe ainda espaço para posterior refinamento na obtenção de resultados especificamente para a língua Portuguesa.

O processo de extração de texto dos ficheiros foi implementado com recurso à interface *IFilter* [30]. Esta interface é utilizada pelo sistema operativo Windows no contexto do seu serviço de pesquisa de ficheiros ou Windows Indexing Service. Como a aplicação é executada num ambiente Windows esta é a única opção para extração de dados dos ficheiros. O processo de extração é executado através da utilização da função *LoadIFilter* presente em *query.dll*. Esta função devolve um objeto COM [31] que implementa a interface *IFilter* para o tipo de ficheiro passado por parâmetro. Esta interface contém os métodos necessários para extração de texto. No ambiente de desenvolvimento do sistema é possível extrair texto de ficheiros em formato TXT, em

todos os formatos Microsoft Office e também de ficheiros PDF através do *Adobe PDF iFilter 9* [32].

7.2 Algoritmo de *Crawling*

O algoritmo de *crawling* tem como principal função percorrer todos os ficheiros armazenados nas contas de cada utilizador para que os dados destes ficheiros possam ser extraídos para o índice. Este é um processo que deve executar com a maior frequência possível de modo a que os dados do índice se encontrem o mais atualizados possível. Note-se que os utilizadores podem fazer alterações aos seus ficheiros sem utilizar o sistema Cloud Cockpit. Por exemplo podem usar as aplicações instaladas nas máquinas clientes que funcionam integradas com o explorador do Windows, ou a própria interface Web de cada serviço. Este facto dificulta o processo porque o processo de adição, remoção ou atualização dos dados sai do domínio do sistema Cloud Cockpit.

O desafio que se coloca é minimizar a necessidade de descarregar, e consequentemente indexar, ficheiros para que o processo seja o mais rápido possível. Para tal é necessário detetar quais os ficheiros adicionados ou alterados para adicionar ao índice e quais os ficheiros removidos, para remover do índice. O desafio de detetar os ficheiros inseridos ou atualizados é abordado mantendo no índice gerado pelo Lucene um identificador único de versão de ficheiro (campo *DocumentVersionId*). Este identificador é um atributo gerado pelo serviço que armazena o ficheiro e está disponível nos metadados do ficheiro. Para detetar os ficheiros removidos é utilizado o campo *NoDelete*.

O algoritmo para resolver aquele desafio organiza-se nas seguintes três fases:

1. Todos os ficheiros presentes no índice do utilizador são marcados para eliminar, através da atualização do campo *NoDelete* com o valor zero (*False*);
2. O modulo aplicacional percorre todas as pastas em cada conta de armazenamento na *Cloud* do utilizador. Por cada ficheiro é verificado se existe no índice algum documento com o *ItemVersionId* do ficheiro. Se existir é marcado o campo *NoDelete* com o valor um (*True*). Caso contrário o

conteúdo é descarregado para uma pasta temporária, é indexado e por último é eliminado da pasta temporária;

3. O processo final passa por eliminar todos os ficheiros do índice que apresentem o campo *NoDelete* com o valor zero (*False*) pois isso quer dizer, não foram detetados pelo processo de *crawling*. Isto garante que todos os ficheiros que já não se encontram nas contas do utilizador são eliminados do índice (quer por terem sido eliminados ou por terem sido atualizados).

Note-se que se as operações de adição ou remoção de ficheiros forem executadas pelo sistema Cloud Cockpit são executadas as correspondentes operações no índice de pesquisa. Com isto pretende-se que o índice esteja o mais atualizado possível e minimizar o tempo de *crawling* e indexação no momento em que for executado. Este algoritmo é importante no sentido de garantir que o utilizador pode sempre utilizar a pesquisa do sistema, mesmo que utilize as interfaces específicas dos serviços de armazenamento para manipular os ficheiros.

Por último é de referir que este algoritmo está implementado no módulo aplicacional que comunica com os serviços de armazenamento na *Cloud* através da interface *IServiceClient* apresentada no ponto 5.3 e com o índice através da interface *ISearchClient* apresentada no ponto 5.2. Isto garante que se forem integrados novos serviços no sistema Cloud Cockpit, estão aptos a serem indexados.

Capítulo 8

Implementação da API

A camada de API define-se como a interface pública do módulo aplicacional para o código cliente. Atualmente o código cliente que utiliza esta camada é o código Javascript que é executado no *browser* do cliente. O objetivo desta camada é que futuramente possa ser explorada por outros clientes, tais como aplicações de dispositivos móveis.

A implementação desta camada foi efetuada através da utilização da tecnologia ASP.NET Web API [33]. Esta é uma *framework* que permite disponibilizar serviços HTTP para desenvolver aplicações REST sobre a plataforma .NET. As chamadas à API são efetuadas através de pedidos HTTP, e o retorno é no formato JSON ou XML, dependendo do formato que o cliente optar.

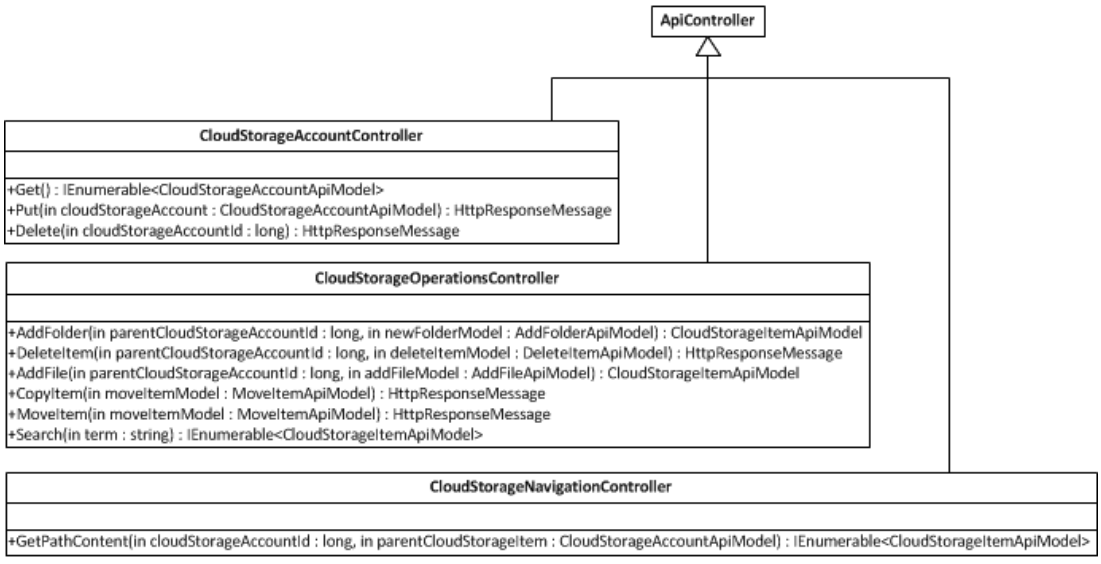

Figura 17 – Classes de implementação da API

A Figura 17 apresenta as classes utilizadas para implementação da API, todas derivam de *ApiController* que é a classe base dos *endpoints* na ASP.NET Web API. A classe *CloudStorageAccountController* contém métodos para selecionar, inserir ou eliminar contas de armazenamento na *Cloud*. A classe *CloudStorageOperationsController* contém os métodos para execução de operações sobre as contas de armazenamento, como por exemplo adicionar uma pasta ou um ficheiro. A classe *CloudStorageAccountController* contém os métodos necessários para navegação nas contas de armazenamento.

Para aceder a estes métodos o cliente necessita de estar previamente autenticado na aplicação Cloud Cockpit. Se este requisito não se verificar é-lhe negado o acesso.

Os modelos utilizados pela API são distintos dos modelos utilizados no módulo aplicacional, sendo, no momento da invocação, convertidos. Isto permite que os modelos da API sejam especializados nas operações que são disponibilizadas na API e ao mesmo tempo não são restringidos pela modelação e arquitetura do módulo aplicacional.

Capítulo 9

Implementação do Módulo de Apresentação

As aplicações Web apresentam-se como um mais valia no contexto das aplicações multiplataforma pois são construídas com base em standards como HTML, CSS e Javacript que são suportados pela generalidade dos dispositivos atuais. Por esta razão a interface com o utilizador está implementada com as tecnologias HTML5, CSS3 e Javascript.

A interface do sistema encontra-se otimizada para computadores, *tablets* e *smartphones*. Para a implementação deste requisito foi utilizada a técnica de *Responsive Layout* que permite adaptar a interface ao dispositivo, através da utilização de CSS3, ao invés da criação de uma interface específica para cada tipo de dispositivo alvo. A desvantagem desta abordagem é que é necessário ter em conta que o utilizador pode estar a aceder à aplicação a partir de um ecrã relativamente pequeno. Este ponto é importante quando se decide a forma como as funcionalidades são apresentadas ao utilizador. Por exemplo, no caso deste sistema não foi dada a possibilidade de explorar duas contas de armazenamento na *Cloud* "lado-a-lado" porque esta funcionalidade não faz sentido num ecrã de um *smartphone*.

A implementação do módulo de apresentação foi baseada no *framework* ASP.NET MVC 4. Esta possibilita a implementação de aplicações Web baseada no padrão *Model View Controller* (MVC). Na vertente cliente foi utilizado *KnockoutJS* que é uma biblioteca Javascript baseada no padrão *Model View View Model* (MVVM), esta visa o auxilio à criação de interfaces ricas e responsivas. O padrão MVVM é uma variante do padrão MVC que foi desenvolvido para dar suporte a interfaces que são desenvolvidas

de forma declarativa. Tem como objetivo abstrair totalmente o desenvolvimento da interface da lógica de negócio através da utilização de *bindings* declarativos. [34]

9.1 Interface adaptativa

A implementação de uma interface adaptativa, ou *responsive layout* em inglês, assenta fundamentalmente em dois conceitos: a implementação de um *layout* em grelha e a utilização de CSS *media queries* [35].

Um layout em grelha é uma forma de organizar a estrutura base da aplicação em colunas. Estas colunas definem o espaço que cada zona irá utilizar. O *layout* foi dividido em quatro colunas, de modo a que o espaço mínimo utilizado por uma zona seja de 25% do espaço disponível.

Na Figura 18 é apresentado o layout da aplicação quando visto num resolução de computador ou *tablet.* A zona esquerda (a cor de laranja) que ocupa uma coluna, ou 25% do espaço disponível. A zona direita (a verde) ocupa três colunas, 75% do espaço disponível. A barra de topo com o logotipo ocupa as quatro colunas.

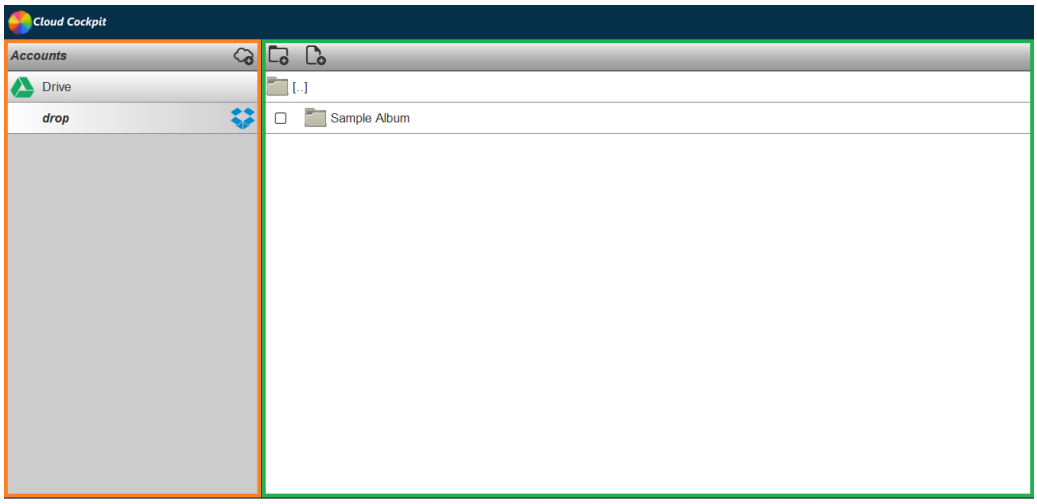

Figura 18 – Layout da aplicação (Computador e *Tablet*)

As *media queries* são uma especificação do CSS3 [36] para permitir definir estilos no âmbito das capacidades especificas do dispositivo, tais como largura, altura, orientação, resolução, entre outros. A Figura 19 apresenta o *layout* da aplicação num s*martphone* com uma resolução horizontal máxima de 320px.

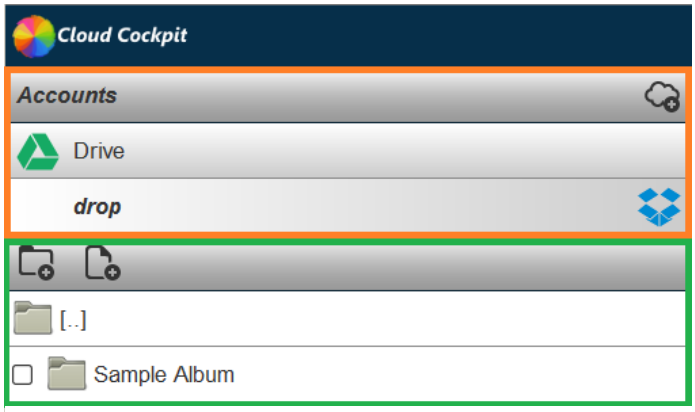

Figura 19 – Layout da aplicação (*Smartphone*)

Para obter este comportamento definiu-se que as colunas ocupam 100% do espaço disponível. E definiu-se uma *media query* para que todos os dipositivos com resolução horizontal superior a 768px as colunas passem a ocupar 25% do espaço, ver listagem abaixo.

Note-se que foi seguida uma abordagem *mobile first* em que, se o dispositivo não suportar *media queries* é-lhe apresentado a versão *smartphone*. Foi seguida esta abordagem porque na generalidade os dispositivos recentes e navegadores Web de computadores suportam *media queries* [37]. Apenas dispositivos mais antigos podem não suportar, mas mesmo nesses garante-se que a aplicação é apresentada corretamente.

```
.g1,.g2,.g3,.g4{display:block; position: relative; margin:0}
.g1,.g2,.g3,.g4{width:100.0%}
@media only screen and (min-width: 768px) {
        .g1,.g2,.g3,.g4 {display:inline; float: left}
        .g1 {width:25%}
        .g2 {width:50%;}
        .g3 {width:75.0%}
        .g4 {width:100.0%}
}
```

Listagem 2 – Código CSS com definição de colunas

9.2 Estrutura de páginas da aplicação

A aplicação é composta por duas páginas HTML. A página de Login que é a página de entrada na aplicação e permite ao utilizador autenticar-se no sistema e a página do Explorador, que é o espaço onde o utilizador pode adicionar e explorar as suas contas de armazenamento na *Cloud*.

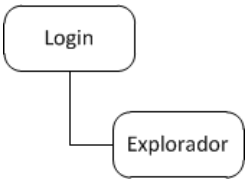

Figura 20 – Estrutura de páginas da aplicação

9.3 Utilização de KnockoutJS

A biblioteca Javascript KnouckoutJS é utilizada na página do Explorador e tem como objetivo auxiliar na construção de um cliente rico. As principais características desta biblioteca são as seguintes:

- Controlo de dependências: refrescamento automático da interface com o utilizador sempre que os dados do modelo (Javascript) sejam alterados;
- *Bindings* declarativos: *binding* simplificado através da utilização de atributos no HTML para interligar a interface com o modelo.

A listagem seguinte apresenta o exemplo de utilização do KnockoutJS na lista de contas de armazenamento apresentada na interface. Esta tem como objetivo controlar a lista de contas de armazenamento na *Cloud* do utilizador.

```
<ul data-bind="foreach: cloudStorageAccounts">
       <li data-bind="click: $parent.navigateToCloudStorageAccount, css: { 'active' : 
$parent.currentCloudStorageAccount() == $data}">
               <span data-bind="css: $data.cssTypeClass()" class="icon"></span>
               <!--ko text: alias--><!--/ko-->
       </li>
</ul>
```

Listagem 3 – Exemplo de utilização do KnockoutJS

Na listagem apresentada estão visíveis as características da biblioteca apresentadas anteriormente:

- *Bindings* declarativos: os *bindings* são efetuados através do atributo *data-bind* nos elementos;
- Controlo de dependências: é criado um elemento <*li*> por cada conta de armazenamento presente no objeto do modelo *cloudStorageAccounts*. Mas inicialmente este objeto não contém dados, estes são carregados através de uma chamada AJAX [38] no carregamento da página. Após esta chamada é executado automaticamente o código de criação dos elementos <*li*>.

Capítulo 10

Validação e Testes

A arquitetura modular do sistema permitiu testar módulo-a-módulo à medida da evolução do desenvolvimento. Não foram utilizados testes unitários pois a natureza da solução não facilita a verificação automática do resultado das operações nas contas dos serviços de armazenamento na *Cloud*.

Os testes às operações em serviços de armazenamento na *Cloud* foram efetuados com recurso a uma conta no serviço Dropbox e outra do serviço Google Drive. Estas duas amostras permitiram validar a correta implementação das operações de: criação e remoção de pastas, carregamento e remoção de ficheiros, movimentação de ficheiros na mesma conta de armazenamento na *Cloud* e a movimentação de ficheiros entre duas contas diferentes.

Em relação à pesquisa de ficheiros decidiu-se analisar o tempo de *crawling* e indexação de cada serviço em relação à dimensão total dos ficheiros a descarregar. A necessidade desta análise surgiu devido ao facto de não existir a noção da complexidade temporal deste processamento sendo que essa informação é essencial para avaliar a frequência com que o processo de *crawling* e indexação pode executar-se.

Foram efetuados três tipos de análise: (i) Tempo de *crawling* e indexação em função do dimensão dos conteúdos a descarregar e indexar (por tipo de serviço de armazenamento), (ii) Tempo de *download* de ficheiros em função da dimensão de ficheiro (por tipo de serviço de armazenamento), e (iii) Tempo de extração de conteúdos de ficheiro em função da dimensão de ficheiro (por tipo de ficheiro).

A Figura 21 apresenta o gráfico que relaciona o tempo, em segundos, de uma operação de *crawling* e indexação com a dimensão total dos ficheiros descarregados, em *megabytes*. O gráfico apresenta os dados com base em vinte e um processos de *crawling* e indexação em cada serviço. Esta análise está dividida por tipo de serviço, a azul encontram-se os dados relacionados com o Dropbox, e a verde os dados relacionados com o Google Drive.

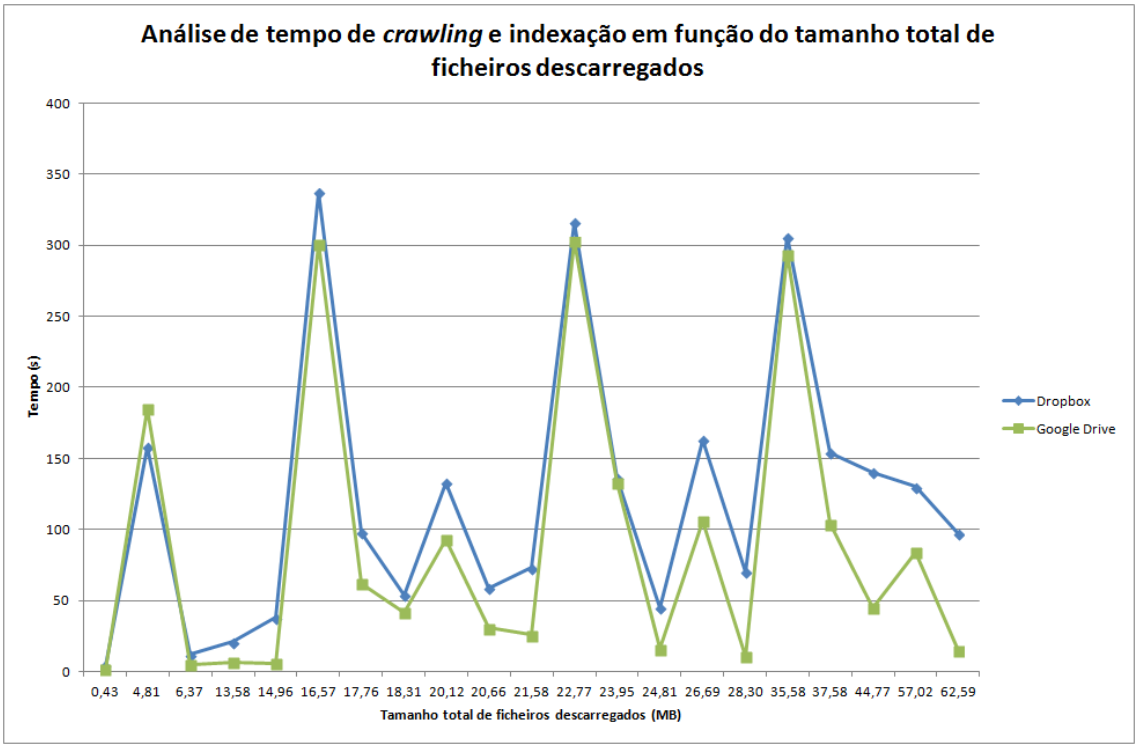

Figura 21 – Análise de tempo de *crawling* e indexação

Da observação do gráfico pode-se constatar que o tempo de *crawling* não está diretamente relacionado com o volume de dados a processar. É também notório que, na generalidade, o tempo de *crawling* de documentos no serviço Google Drive é mais rápido do que no serviço Dropbox.

Com as duas análises seguintes pretende-se verificar a razão dos tempos de *crawling* não estarem diretamente relacionados com o volume de dados, bem como a razão do tempo de *crawling* e indexação ser inferior no serviço Google Drive.

A Figura 22 apresenta a análise de tempos de descarregamento de ficheiros individuais em função da dimensão dos mesmos. É notório que o tempo de descarregamento de ficheiros do serviço Dropbox é maior, e apresenta um

crescimento exponencial com o aumento da dimensão do ficheiro, em relação aos ficheiros armazenados no Google Drive. Este facto explica a razão pela qual o tempo de *crawling* e indexação do serviço Dropbox ser, na generalidade, superior ao serviço Google Drive.

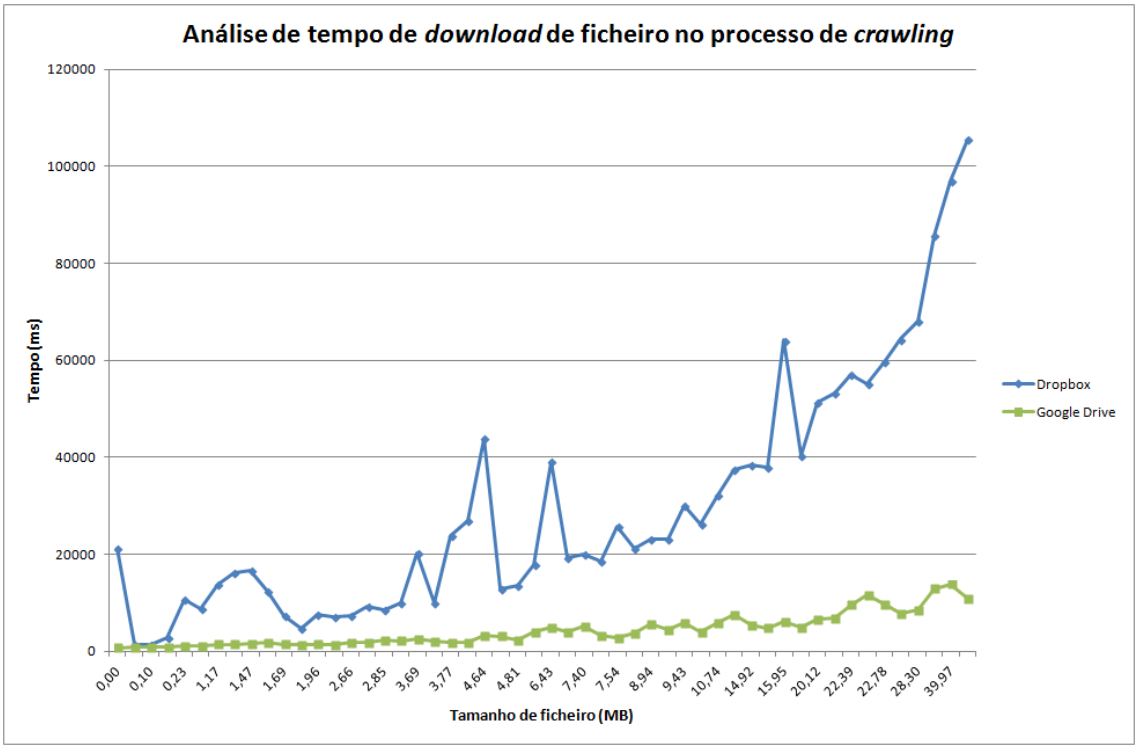

Figura 22 – Análise de tempo de *download* de ficheiro

É ainda de notar que, com os dados obtidos, a média de tempos de descarregamento de ficheiros (de ambos os serviços) é de 15 segundos, enquanto que a média de *crawling* e indexação é de 105 segundos. Pelo que se pode constatar que o fator que tem maior influência no tempo de *crawling* e indexação não é a dimensão total de ficheiros descarregados.

Os gráficos seguintes apresentam o tempo de extração de conteúdo de diferentes tipos de ficheiros, em função da dimensão dos mesmos: ficheiros texto (extensão TXT), documentos *Word* (extensão DOCX), folhas de cálculo (extensão XLSX), diapositivos (extensão PPTX) e P*ortable Document Format* (extensão PDF). Entenda-se por tempo de extração de conteúdo o tempo que o *IFilter* respetivo demorar a extrair o texto do ficheiro. Os gráficos foram agrupados por ordem de grandeza do eixo temporal de forma a que os dados possam ser analisado com clareza.

A Figura 23 apresenta a análise relativa a ficheiros DOCX e TXT. A extração de conteúdo de ficheiros TXT está diretamente relacionada com a dimensão dos mesmos. O tempo de extração de conteúdo de ficheiros DOCX não está relacionado com a dimensão, sendo que poderá ter a ver com o tipo de conteúdos do documento.

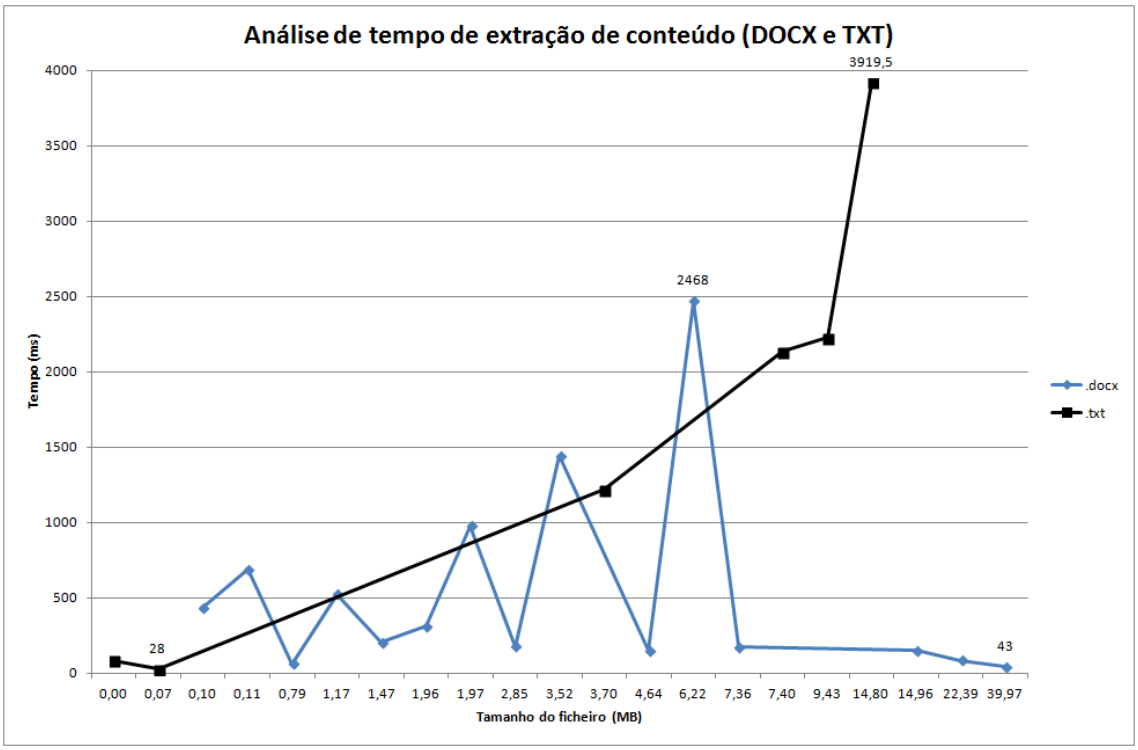

Figura 23 – Análise de tempo de extração de conteúdo de ficheiros (DOCX e TXT)

Os tempos da análise variam entre, aproximadamente, 0,028 e 4 segundos para ficheiros TXT e entre os 0,045 e 2,5 segundos nos ficheiros DOCX.

A Figura 24 contém a análise dos ficheiros PPTX e XSLX. Enquanto o tempo dos ficheiros XLSX aparentam estar relacionados com a dimensão dos ficheiros, nos ficheiros PPTX essa relação não é completamente evidente, o que poderá indicar que o tempo de extração está relacionado com o conteúdo dos documentos. A ordem de grandeza dos tempos é superior à analise anterior. Os ficheiros XLSX apresentam tempos, aproximadamente, entre 20 e 100 segundos e os ficheiros PPTX estão entre os 0,173 e 34 segundos.

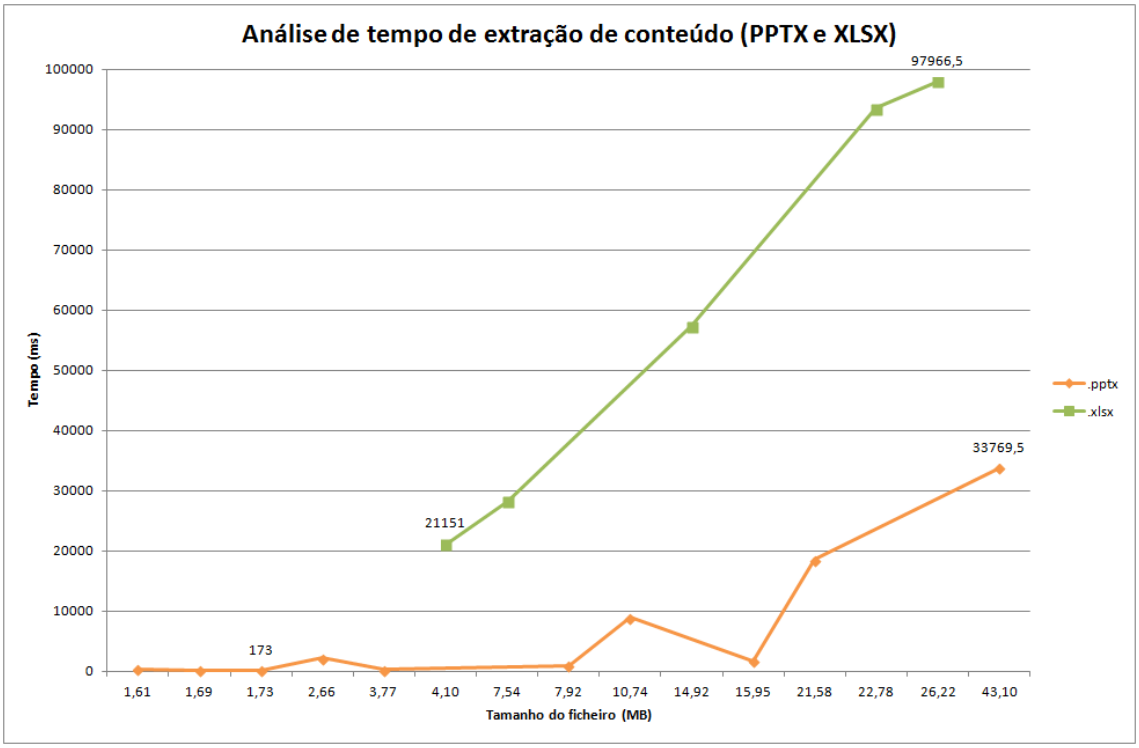

Figura 24 – Análise de tempo de extração de conteúdo de ficheiros (PPTX e XLSX)

A Figura 25 apresenta a análise de tempos de ficheiros PDF. Como se pode verificar a distribuição de tempos é irregular pelo que parece que está dependente do conteúdo do documento.

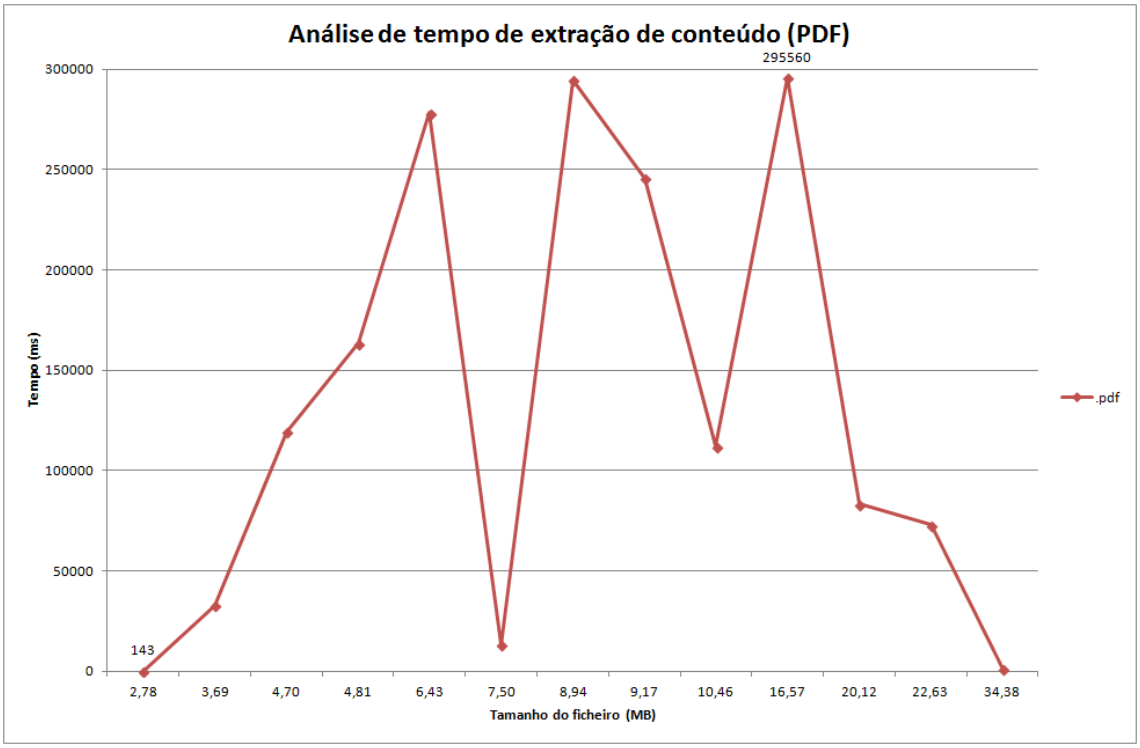

Figura 25 – Análise de tempo de extração de conteúdos de ficheiros (PDF)

É de salientar que os tempos estão, aproximadamente, entre os 0,143 e 300 segundos. Valores que se apresentam muito elevados quando comparados com as restantes análise.

Com base nas análises efetuadas conclui-se que o fatores com maior relevância no tempo do *crawling* e indexação são o tipo de ficheiros e o conteúdo dos mesmos. Ao invés da dimensão dos ficheiros, como era inicialmente previsto. Estes fatores estão diretamente relacionados com o desempenho da extração do texto por parte do *IFilter* respetivo. Os ficheiros PDF são os que apresentam pior desempenho pelo que se conclui deve de ser encontrada uma alternativa ao *IFilter* fornecido pela Adobe no sentido de diminuir o tempo de extração de conteúdo de ficheiros PDF e consequentemente o tempo de *crawling* e indexação de ambos os serviços suportados.

Não é possível determinar a frequência com que o processo de *crawling* e indexação deve de ser executado uma vez que o tempo de um processo de *crawling* e indexação está dependente do tipo de conteúdo dos ficheiros e não da dimensão dos mesmo, fator no qual o sistema não tem controlo. Assim conclui-se que este processo deve ser executado com a maior frequência possível para minimizar o número de ficheiros a indexar. Tendo em consideração que este processo não pode degradar a capacidade de resposta do sistema.

A recolha das métricas apresentadas foram efetuadas sob as mesmas condições de largura de banda e carga no servidor de desenvolvimento do sistema.

Capítulo 11

Conclusões

O projeto Cloud Cockpit surge como resposta a um desafio pessoal de criar um sistema que, atualmente, possa ser explorado por utilizadores no sentido de solucionar o problema da dispersão de conteúdos pelos sistemas de armazenamento na *Cloud*. A oportunidade de idealizar e desenvolver um sistema não foi encarada apenas com o intuito de desenvolver um projeto académico, mas também com a ideia de desenvolver um produto que possa ser relevante no contexto dos problemas atuais.

Dos objetivos inicialmente traçados verifica-se a implementação com sucesso de uma interface multiplataforma que permite utilizar o sistema com *smartphones*, *tablets* e computadores. É de realçar que esta abordagem é a ideal se tivermos em conta a produtividade de desenvolvimento *versus* a disponibilidade da aplicação no contexto de dispositivos abrangidos. A abordagem referida revela-se, ao mesmo tempo, algo limitadora uma vez que as funcionalidades disponibilizadas têm de fazer sentido em todos os tipos de dispositivos, o que nem sempre se verifica. Para colmatar esta lacuna, a solução de disponibilizar uma API, que é utilizada pela atual interface, verifica-se promissora uma vez que é possível desenvolver novas interfaces para o sistema reaproveitando toda a lógica aplicacional.

No âmbito das operações sobre ficheiros e integração entre serviços de armazenamento constata-se que a adição, remoção movimentação e cópia de ficheiros ou pastas em todos nos serviços foi implementada com sucesso. Tanto dentro da mesma conta de armazenamento como em contas diferentes. A movimentação e cópia de pastas entre contas diferentes não foi implementada uma vez que seria necessário validar o tamanho e complexidade da estrutura a copiar. O

que poderia levar a tempos excessivos de execução ou mesmo diminuir a capacidade de resposta do sistema.

É de referir que a arquitetura da solução foi desenhada de forma a que a integração com outros serviços de armazenamento seja transparente ao módulo aplicacional, o que permite diminuir o custo de desenvolvimento e complexidade de integração.

Por último a implementação da pesquisa sobre conteúdos não estruturados armazenados nas contas dos utilizadores encontra-se funcional podendo, no entanto, a indexação ter que ser refinada para ajustar o *steaming* e remoção de *stop words* de acordo com o idioma dos conteúdos. É ainda de referir que o *IFilter* disponibilizado pela Adobe, para extração de texto de ficheiros PDF, tem alguns problemas de performance. Esta situação leva à necessidade de encontrar outra implementação para a extração de texto de ficheiros PDF no sentido de aumentar a performance do sistema na vertente de *crawling* e indexação.

11.1 Trabalho Futuro

É de referir que, não sendo um objetivo principal, o *login* e registo do utilizador não está implementado. Pelo que, para o sistema ser disponibilizado para utilização, estas duas funcionalidade devem ser implementadas. Para além destas funcionalidades o âmbito do projeto deixa em aberto um conjunto de funcionalidade que seriam interessantes de explorar.

Integração de outro serviços de armazenamento na *Cloud*, como por exemplo Box, Skydrive, SugarSync [39], entre outros. Ou ainda com serviços de componente de partilha social tais como Youtube, Vimeo, Picasa ou Flicker .

Uma outra funcionalidade com potencial para ser implementada, é a possibilidade de outras aplicações utilizarem a API do sistema para armazenar ficheiros de uma forma independente do serviço de armazenamento especifico.

Para além destas, também poderão ser implementadas funcionalidade de gestão e otimização de espaço disponível nas contas de armazenamento.

Bibliografia

[1] J. Wu, L. Ping, X. Ge, Y. Wang e J. Fu, "Cloud Storage as the Infrastructure of Cloud Computing," *2010 International Conference on Intelligent Computing and Cognitive Informatics,* 2010.

[2] "YouTube," [Online]. Available: http://www.youtube.com/. [Acedido em 28 09 2013].

[3] "Vimeo, Your videos belong here," [Online]. Available: https://vimeo.com/. [Acedido em 28 09 2013].

[4] "Picasa," [Online]. Available: http://picasa.google.com/. [Acedido em 28 09 2013].

[5] "Flicker," [Online]. Available: http://www.flickr.com/. [Acedido em 28 09 2013].

[6] "Dropbox," [Online]. Available: https://www.dropbox.com/. [Acedido em 28 09 2013].

[7] "Google Drive," [Online]. Available: https://drive.google.com/. [Acedido em 28 09 2013].

[8] P. Mell e T. Grance, "The NIST Definition of Cloud Computing," *National Institute of Standards and Technology Special Publication 800-145,* p. 7, 2011.

[9] J. Chen, M. Lai, Y. Huang, K. Yang e G. Zhou, "The Business Model of Cloud Storage," *CLOUD COMPUTING 2010 : The First International Conference on Cloud Computing, GRIDs, and Virtualization,* 2010.

[10] "Amazon Simple Storage Service (Amazon S3)," [Online]. Available: http://aws.amazon.com/pt/s3/. [Acedido em 28 09 2013].

[11] "Windows Azure Storage," [Online]. Available: http://www.windowsazure.com/en-us/documentation/services/storage/. [Acedido em 28 09 2013].

[12] "Google Cloud Storage," [Online]. Available:

https://cloud.google.com/products/cloud-storage. [Acedido em 28 09 2013].

[13] “Smart Business Storage Cloud,” [Online]. Available: http://www-935.ibm.com/services/us/en/it-services/smart-business-storage-cloud.html. [Acedido em 28 09 2013].

[14] “Microsoft SkyDrive - Access files anywhere.,” [Online]. Available: https://skydrive.live.com/. [Acedido em 28 09 2013].

[15] “Otixo: all your cloud files,” [Online]. Available: http://otixo.com/. [Acedido em 28 09 2013].

[16] “CloudKafé - Organizing your cloud,” [Online]. Available: https://www.cloudkafe.com/. [Acedido em 28 09 2013].

[17] “Entity Framework,” [Online]. Available: http://msdn.microsoft.com/en-us/data/ef.aspx. [Acedido em 03 08 2013].

[18] “The Repository Pattern,” [Online]. Available: http://msdn.microsoft.com/en-us/library/ff649690.aspx. [Acedido em 16 08 2013].

[19] “Factory Method Design Pattern,” [Online]. Available: http://sourcemaking.com/design_patterns/factory_method. [Acedido em 16 08 2013].

[20] “The OAuth 1.0 Protocol,” [Online]. Available: http://tools.ietf.org/html/rfc5849. [Acedido em 15 08 2013].

[21] “The OAuth 2.0 Authorization Framework,” [Online]. Available: http://tools.ietf.org/html/rfc5849. [Acedido em 15 08 2013].

[22] “Introducing OAuth 2.0,” [Online]. Available: http://hueniverse.com/2010/05/introducing-oauth-2-0/. [Acedido em 15 08 2013].

[23] “DotNet Open Auth,” [Online]. Available: http://dotnetopenauth.net/. [Acedido em

19 08 2013].

[24] “Representational State Transfer (REST),” [Online]. Available: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. [Acedido em 25 08 2013].

[25] “Dropbox Core API,” [Online]. Available: https://www.dropbox.com/developers/core/docs. [Acedido em 17 07 2013].

[26] “Google Drive API Reference,” [Online]. Available: https://developers.google.com/drive/v2/reference/. [Acedido em 17 07 2013].

[27] “Apache Solr,” [Online]. Available: http://lucene.apache.org/solr/. [Acedido em 01 09 2013].

[28] “Lucene .Net Search Engine Library,” [Online]. Available: http://lucenenet.apache.org/. [Acedido em 10 07 2013].

[29] “Xapian,” [Online]. Available: http://xapian.org/. [Acedido em 10 07 2013].

[30] “IFilter interface,” [Online]. Available: http://msdn.microsoft.com/en-us/library/ms691105(v=vs.85).aspx. [Acedido em 02 09 2013].

[31] “COM: Component Object Model Technologies,” [Online]. Available: http://www.microsoft.com/com/default.mspx. [Acedido em 29 09 2013].

[32] “Adobe PDF iFilter 9 for 64-bit platforms,” [Online]. Available: http://www.adobe.com/support/downloads/detail.jsp?ftpID=4025. [Acedido em 25 09 2013].

[33] “ASP.NET Web API,” [Online]. Available: http://msdn.microsoft.com/en-us/library/hh833994(v=vs.108).aspx. [Acedido em 01 09 2013].

[34] “Introduction to Model/View/ViewModel pattern for building WPF apps,” [Online]. Available: http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx.

[Acedido em 29 09 2013].

[35] F. Rivoal, H. Lie, T. Çelik, D. Glazman e A. v. Kesteren, “http://www.w3.org/TR/css3-mediaqueries/,” [Online]. Available: http://www.w3.org/TR/css3-mediaqueries/. [Acedido em 29 09 2013].

[36] “Media Queries,” [Online]. Available: http://www.w3.org/TR/css3-mediaqueries/. [Acedido em 01 09 2013].

[37] “Can I use CSS3 Media Queries?,” [Online]. Available: http://caniuse.com/css-mediaqueries. [Acedido em 01 09 2013].

[38] “AJAX Introduction,” [Online]. Available: http://www.w3schools.com/ajax/ajax_intro.asp. [Acedido em 29 09 2013].

[39] “File Sync & Online Backup - Access and File Sharing from Any Device,” [Online]. Available: https://www.sugarsync.com/. [Acedido em 28 09 2013].

Anexos

ANEXO A – Especificação de Casos de Utilização

UC1 – Registar

<table>
<tr><td>Cenário</td><td colspan="2">Registo de utilizador para utilização do Cloud Cokpit.</td></tr>
<tr><td>Evento Inicial</td><td colspan="2">O utilizador acede ao formulário de registo do sistema</td></tr>
<tr><td>Atores</td><td colspan="2">Utilizador</td></tr>
<tr><td>Pré-Condição</td><td colspan="2">O utilizador não tem a sessão iniciada no sistema.</td></tr>
<tr><td>Pós-Condição</td><td colspan="2">O utilizador fica registado no sistema e o seu login será efetuado através de email/password.</td></tr>
<tr><td colspan="3">Fluxo de Eventos</td></tr>
<tr><td>Ator</td><td colspan="2">Sistema</td></tr>
<tr><td>3. Acede ao formulário de registo do sistema;
4. Insere o e-mail e a password que pretende utilizar.</td><td colspan="2"></td></tr>
<tr><td></td><td colspan="2">5. Valida que o e-mail ainda não existe no sistema;
6. Regista o utilizador no sistema como inativo;
7. Envia e-mail com um endereço para o utilizador ativar a sua conta.</td></tr>
<tr><td>8. Navega para o endereço recebido por e-mail.</td><td colspan="2"></td></tr>
<tr><td></td><td colspan="2">9. Ativa a conta no sistema;
10. Redireciona o utilizador para a área de trabalho do utilizador, com o login efetuado.</td></tr>
<tr><td colspan="3">Exceções</td></tr>
<tr><td colspan="3">• No ponto 3 se o e-mail já existe no sistema e já estiver ativo, o utilizador é informado desta situação;
• No ponto 3 se o e-mail já existe no sistema e não estiver ativo, o utilizador é informado desta situação e é reenviado o e-mail de ativação de conta para o email do utilizador e o caso de utilização continua no ponto 6;</td></tr>
</table>

UC2 – Login

Cenário	Login no Sistema
Evento Inicial	O utilizador acede ao formulário de Login do sistema, selecionado a opção de inserir o e-mail/password
Atores	Utilizador
Pré-Condição	O utilizador não tem a sessão iniciada no sistema.
Pós-Condição	O utilizador fica com a sessão iniciada no sistema, tendo acesso à sua área de trabalho.

Fluxo de Eventos	
Ator	**Sistema**
1. Acede ao formulário de Login do sistema; 2. Insere o seu e-mail e password; 3. Carrega em login.	
	4. Sistema verifica as credenciais de acesso do utilizador; 5. Redireciona o utilizador para a sua área de trabalho.

Exceções
• Se as credenciais de acesso do utilizador forem inválidas o sistema notifica o utilizador desta situação e permanece no formulário de Login;

UC3 – Social Login

Cenário	Login através de um Social Login Provider (Google ou Facebook)
Evento Inicial	O utilizador acede ao formulário de Login do sistema, selecionado um Social Login Provider (Google ou Facebook)
Atores	Utilizador, Social Login Provider
Pré-Condição	O utilizador não tem a sessão iniciada no sistema.
Pós-Condição	O utilizador fica com a sessão iniciada no sistema, tendo acesso à sua área de trabalho.

Fluxo de Eventos		
Ator	**Sistema**	**Social Login Provider**

1. O utilizador acede ao formulário de Login do sistema 2. Seleciona um Social Login Provider (Google ou Facebook) 3.		
	4. Redireciona o utilizador para a página de Login do Social Login Provider	
		5. Autentica o utilizador e retorna a informação para o sistema
	6. Valida que o utilizador já se encontra no sistema 7. Redireciona o utilizador para a sua área de trabalho	
Exceções		
• No passo 5, se o utilizador ainda não se encontra registado, o sistema deve registar o utilizador com a indicação de login através de Social Login Provider e continuar no passo 6.		

UC6 – Apagar Ficheiro

Cenário	Apagar ficheiro de uma conta de cloud storage
Evento Inicial	O utilizador seleciona um ficheiro para apagar de uma conta de cloud storage.
Atores	Utilizador, Cloud Storage Service
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4.
Pós-Condição	O ficheiro selecionado é apagado da conta de cloud storage.
Fluxo de Eventos	
Ator	**Sistema**	**Cloud Storage Service**

1. Seleciona um ficheiro e a opção “Apagar”.		
	2. Pede a confirmação para apagar o ficheiro.	
3. Confirma que o ficheiro é para apagar.		
	4. Faz o pedido para apagar o ficheiro ao cloud storage service.	
		5. Confirma que o ficheiro foi apagado com sucesso.
	6. Notifica o utilizador de ficheiro apagado com sucesso.	
Exceções		
7. No passo 3 se o utilizador não confirmar que o ficheiro é para apagar, o caso de utilização termina; 8. Se o ficheiro não for apagado com sucesso o caso de utilização termina**.**		

UC7 – Adicionar Diretoria

Cenário	Adicionar diretoria numa conta de um cloud storage.		
Evento Inicial	O utilizador navega para o destino e seleciona a opção “Nova Diretoria”.		
Atores	Utilizador, Cloud Storage Service		
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4.		
Pós-Condição	A diretoria é adicionada à conta de cloud storage.		
Fluxo de Eventos			
Ator	**Sistema**	**Cloud Storage Service**	
1. Navega para o destino da nova diretoria; 2. Seleciona a opção “Nova			

Diretoria".		
	3. Faz o pedido criação de diretoria ao cloud storage service.	
		4. Cria a diretoria; 5. Confirma que a diretoria foi criada com sucesso.
	6. Notifica o utilizador que a diretoria foi criada com sucesso.	
Exceções		
• Se a criação de diretoria falhar o utilizador é notificado e o caso de utilização termina.		

<u>UC8 – Apagar Diretoria</u>

Cenário	Apagar diretoria de uma conta de cloud storage	
Evento Inicial	O utilizador seleciona a diretoria para apagar de uma conta de cloud storage.	
Atores	Utilizador, Cloud Storage Service	
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4.	
Pós-Condição	A diretoria selecionada, e todos os seus conteúdos são apagados da conta de cloud storage.	
Fluxo de Eventos		
Ator	**Sistema**	**Cloud Storage Service**
1. Seleciona uma diretoria e a opção "Apagar".		
	2. Pede a confirmação para apagar a diretoria e todos os seus conteúdos.	
3. Confirma que a diretoria é para apagar.		
	4. Faz o pedido para apagar	

	a diretoria ao cloud storage service.	
		5. Apaga a diretoria; 6. Confirma que a diretoria foi apagada com sucesso.
	7. Notifica o utilizador de diretoria apagada com sucesso.	
Exceções		
• No passo 3 se o utilizador não confirmar que a diretoria é para apagar, o caso de utilização termina; • Se a diretoria não for apagada com sucesso o caso de utilização termina.		

UC9 – Copiar Item

Cenário	Copiar ficheiro ou diretoria de uma conta de cloud storage.
Evento Inicial	O utilizador seleciona a opção “Copiar” sobre um ficheiro ou diretoria.
Atores	Utilizador
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4.
Pós-Condição	O item é colocado na área de transferência do sistema e marcado para cópia.

Fluxo de Eventos	
Ator	**Sistema**
1. Seleciona o ficheiro ou diretoria e a opção “Copiar”.	2.
3.	4. Coloca o item na área de transferência; 5. Marca o item na área de transferência como “COPIAR”.
Exceções	
NA	

UC10 – Cortar Item

Cenário	Cortar ficheiro ou diretoria de uma conta de cloud storage.
Evento Inicial	O utilizador seleciona a opção “Cortar” sobre um ficheiro ou diretoria.
Atores	Utilizador
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4.
Pós-Condição	O item é colocado na área de transferência do sistema e marcado para mover.

Fluxo de Eventos	
Ator	**Sistema**
1. Seleciona o ficheiro ou diretoria e a opção “Cortar”.	
	2. Coloca o item na área de transferência; 3. Marca o item na área de transferência como “MOVER”.

Exceções
NA

UC11 – Colar Item na mesma conta de Cloud Storage

Cenário	Colar um ficheiro ou diretoria da área de transferência para um destino na mesma conta de cloud storage da origem.
Evento Inicial	O utilizador escolhe o destino do item e a opção “Colar”.
Atores	Utilizador, Cloud Storage Service
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3. O utilizador associou a conta anteriormente através do caso de utilização UC4. O utilizador colocou um item na área de transferência do sistema através do caso de utilização UC9 ou UC10.
Pós-Condição	O item é colado no destino. Fica alojado apenas no destino se estava

	marcado para cortar. Se estava marcado para copiar fica alojado fisicamente na origem e no destino.

Fluxo de Eventos		
Ator	**Sistema**	**Cloud Storage Service**
1. O utilizador escolhe o destino do item e a opção “Colar”.		
	2. Valida que existe um item na área de transferência. 3. Verifica se a origem e o destino se encontram na mesma conta de cloud storage; 4. Faz um pedido ao cloud storage service para **mover o item** da localização de origem para o destino;	
		5. Executa a operação; 6. Notifica que a operação foi concluída.
7.	8. Retira o item da área de transferência; 9. Notifica o utilizador que o item foi colado com sucesso.	

Cenário alternativo 1
1. No passo 4, se o item estiver marcado para copiar, o sistema faz o pedido ao cloud storage service para **copiar o item**;

Exceções
1. No passo 2, se não existir ficheiro na área de transferência notifica o utilizador e o caso de utilização termina; 2. No passo 3 se a localização de origem for numa conta de cloud storage diferente da localização de destino, é invocado o caso de utilização no passo 3; 3. Se o colar do ficheiro falhar o utilizador é notificado e o caso de utilização termina**.**

UC13 – Pesquisar Ficheiros

Cenário	Pesquisa de ficheiros
Evento Inicial	O utilizador insere um termo de pesquisa na caixa de pesquisa
Atores	Utilizador
Pré-Condição	O utilizador deve estar com a sessão iniciada no sistema, através do caso de utilização UC2 ou UC3.
Pós-Condição	São apresentados os resultados da pesquisa

Fluxo de Eventos	
Ator	**Sistema**
1. Insere um termo de pesquisa na caixa	
	2. Consulta o índice interno e apresenta os ficheiros que se relacionam com o termo de pesquisa.

Exceções
• No passo 2 se não existirem ficheiros que se relacionem com o termo de pesquisa, o utilizador é informado que não existem resultados.