André Filipe Graça Lopes

Licenciado em Ciência e Engenharia Informática

Arquitetura para um Repositório de Dados e Metadados de Observação Terrestre

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Orientador: João Carlos Gomes Moura Pires,
Professor Auxiliar, Universidade Nova de Lisboa

Coorientadores: João Carlos Antunes Leitão,
Professor Auxiliar, Universidade Nova de Lisboa

Armanda Rodrigues,
Professora Auxiliar, Universidade Nova de Lisboa

Júri

Presidente:	Nuno Manuel Ribeiro Preguiça
Arguente:	Maribel Yasmina Campos Alves Santos
Vogal:	João Carlos Gomes Moura Pires

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2019

Arquitetura para um Repositório de Dados e Metadados de Observação Terrestre

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Às Anas da minha vida.

Agradecimentos

Começo por agradecer ao meu orientador, João Moura Pires, pela disponibilidade e dedicação ao longo da conceção e desenvolvimento das ideias apresentadas nesta dissertação. Os ensinamentos que adquiri das nossas múltiplas reuniões extravasam o contexto académico e profissional: aprendi a abordar os problemas a um nível concecional; apercebi-me de quão importante é conseguir argumentar cada decisão tomada; e aprendi a fazer uma melhor gestão do meu tempo e stresse. Não posso deixar de agradecer também ao professor João Leitão pela ajuda nos detalhes mais técnicos de implementação. Adicionalmente, interessa deixar aqui o agradecimento à professora Armanda Rodrigues. Finalmente, importa reconhecer e agradecer o contributo que o professor Carlos Damásio teve nesta dissertação.

Agradeço também aos restantes professores do Departamento de Informática da FCT-UNL por todo o conhecimento que me transmitiram e pela disponibilidade que sempre demonstraram. Em especial, importa agradecer à NOVA-LINCS pela bolsa de investigação que me foi atribuída no contexto desta dissertação.

À minha mãe, Ana Paula Lopes, que sacrificou muito para que eu tivesse as condições necessárias para frequentar o ensino superior. Espero que esteja orgulhosa, pois foram os valores que transmitiu que me moldaram na pessoa que sou hoje. Ao meu irmão, Vasco Lopes, por ter sido o exemplo pelo qual toda vida me guiei e pelo suporte técnico (gratuito) muitas vezes prestado. Agradecer também ao meu tio, Sérgio Paulo, por me ajudar sempre a tomar melhores decisões, auxiliando-me sempre que precisei.

Um agradecimento especial à Ana Pereira pela constante presença e paciência nos meus momentos de cansaço e frustração. Estou grato por nunca me ter deixado desistir, e por me incentivar sempre a fazer melhor. Obrigado por me acompanhares neste percurso, sem ti não seria tão gratificante.

Resta agradecer aos meus amigos e colegas por me terem acompanhado nas exigências da experiência académica, bem como nos seus muitos aspetos positivos, contribuindo sempre para o meu sucesso pessoal e académico.

Um grande obrigado a todos!

Resumo

A deteção remota tem vindo a tornar-se o método mais importante de recolha de informação sobre a superfície terrestre. Neste momento, existem cerca de 700 satélites ativos que têm como missão a observação da Terra. Os dados gerados por estes revelam claras caraterísticas de *big data*: volume, velocidade, diversidade. No que toca ao volume, no final de 2017, a NASA armazenava cerca de 25 *petabytes*. Em relação à velocidade com que é feita a ingestão, esta, em 2017, era de quase 12 *terabytes*/dia. Já a diversidade surge devido às diferentes aplicações, metamodelos, fontes e formatos dos produtos.

Neste contexto, os múltiplos fornecedores já têm os seus próprios repositórios e catálogos, nos quais os *standards* de representação de dados e metadados diferem. Portanto, é importante definir uma camada de abstração intermédia. Outro desafio no desenvolvimento destes repositórios passa pela heterogeneidade dos dados e metadados de deteção remota. Sendo que os metamodelos dos diferentes produtos devem ser extensíveis, de modo a incorporar domínios aplicacionais específicos. Finalmente, aliado ao repositório, existe a necessidade de incorporar cadeias de processamento local.

A abordagem para endereçar os problemas referidos condensa-se em cinco pontos: automatização da ingestão de dados e metadados; especificação hierárquica do metamodelo; linguagem de especificação ETL; mecanismo de interrogação de alto nível; e *framework* de processamento local.

Por fim, a arquitetura proposta foi avaliada recorrendo a um conjunto de casos de estudo reais. Nesta avaliação foram assinaladas as vantagens de realizar todo o fluxo de desenvolvimento numa só plataforma. Paralelamente, comparou-se com a plataforma Google Earth Engine, representante do estado da arte, cujo foco é a computação. Tendo-se concluído que a abordagem proposta não oferece tantas garantias de escalabilidade, mas, na perspetiva da catalogação, a presença de uma gestão colaborativa dos produtos e dos seus metamodelos é uma grande vantagem face às outras soluções.

Palavras-chave: Deteção remota, metamodelos, imagens de satélite, ingestão de dados, heterogeneidade, ETL

Abstract

Remote sensing has become the most important method for collecting information about the Earth surface. Presently, there are around 700 active satellites with earth observation as its mission. The generated data clearly reveals big data features: volume, velocity, diversity. Concerning volume, NASA, in the end of 2017, stored about 25 petabytes. Regarding the data ingestion speed, it is currently around 12 terabytes/day in the same system. Moving on to the diversity, this arises due to the different applications, metamodels, sources and formats of remote sensing products.

In this context, multiple providers have their own repositories and catalogs, in which data and metadata standards differ. Consequently, there is a need to define an intermediate abstraction layer. Another challenge in the development of this type of repositories is the data and metadata heterogeneity. Furthermore, the products metamodel should be extensible, in order to incorporate specific domains. Finally, coupled with the repository, it is important to allow the incorporation of local processing pipelines.

The approach chosen to address the mentioned problems comes down to five points: automatic data and metadata ingestion; hierarchical metamodel specification; ETL specification language; high-level interrogation mechanism; and local processing framework.

Lastly, the proposed architecture was evaluated through a set of real case studies. In this evaluation were highlighted the advantages of integrating all the development flow in only one platform. Additionally, the proposed solution was compared to Google Earth Engine platform, which is focused in the computation and represents the state of the art. It has been concluded that the proposed architecture does not offer so many guarantees of scalability. But from a cataloging perspective, the presence of collaborative management of the products and their metamodels is a great advantage over other solutions.

Keywords: Remote sensing, metamodels, satellite imagery, data ingestion, heterogeneity, ETL

Índice

Lista de Figuras **xvii**

Lista de Tabelas **xix**

Listagens **xxi**

1 Introdução **1**
1.1 Contexto e Motivação . . . 1
1.2 Problema . . . 3
1.3 Solução Proposta . . . 5
1.4 Contribuições . . . 6
1.5 Organização do Documento . . . 7

2 Estado da Arte e Trabalho Relacionado **9**
2.1 Deteção Remota . . . 10
2.1.1 Programas . . . 10
2.1.2 Órbitas . . . 11
2.1.3 Projeções . . . 13
2.1.4 Sensores . . . 13
2.1.5 Resoluções . . . 15
2.2 Produtos de Deteção Remota . . . 16
2.2.1 Cadeias de Processamento . . . 17
2.2.2 Formatos . . . 18
2.3 Metamodelos . . . 21
2.3.1 ISO 19115 . . . 21
2.3.2 INSPIRE . . . 23
2.3.3 UMM . . . 24
2.4 Catálogos e Plataformas . . . 26
2.4.1 Copernicus Open Access Hub . . . 27
2.4.2 DIAS . . . 31
2.4.3 Earth Explorer . . . 34
2.4.4 Earth Data . . . 34
2.4.5 Google Earth Engine . . . 36

2.5 Conclusão 39

3 Abordagem **41**
3.1 Descrição Geral 41
3.2 Arquitetura 42
3.2.1 Protocolo de Distribuição da Ingestão 45
3.3 Fontes de Dados 49
3.3.1 Heterogeneidade das API 49
3.3.2 Fluxo dos Dados e Metadados 51
3.3.3 Procedimento de Especificação das Fontes de Dados 54
3.4 Metamodelo 56
3.4.1 Procedimento de Especificação 56
3.4.2 Extensibilidade 58
3.4.3 Linguagem de Especificação de ETL 59
3.5 API 63
3.6 Conclusão 64

4 Protótipo **67**
4.1 Especificação 67
4.1.1 Requisitos 68
4.1.2 Instanciação da Arquitetura 68
4.1.3 Infraestrutura 71
4.2 Detalhes de Implementação 73
4.2.1 Configuração das Fontes de Dados 73
4.2.2 Especificação do Metamodelo 75
4.2.3 Especificação hierárquica de ETL 77
4.2.4 Tradução de XML para JSON 77
4.2.5 Mapeamento dos *endpoints* na API 78
4.2.6 *Deploy* Automatizado 79
4.3 Conclusão 81

5 Avaliação **83**
5.1 Metodologia de Avaliação 84
5.2 Contexto Experimental 85
5.3 Casos de Estudo 86
5.3.1 Deteção Remota de Parcelas Agrícolas 86
5.3.2 Deteção Remota de Estruturas Permanentes 91
5.3.3 Deteção Remota do Estado da Vegetação em Faixas de Gestão de Combustível de Incêndios 95
5.3.4 Fusão de Imagens de Satélite 98
5.4 Conclusão 101

6 Conclusão **103**
6.1 Trabalho Futuro 105

Bibliografia **109**

I Anexos **121**

Lista de Figuras

1.1 Ingestão de dados diária do EOSDIS. Informação sobre os últimos três anos e previsão até 2022. [31] . . . 3

2.1 Diferentes níveis de excentricidade (e) de uma órbita. [49] . . . 12
2.2 Zonas da projeção UTM. [53] . . . 13
2.3 Bandas (B1, B9, B10) do Sentinel-2 com resolução espacial de 60 por 60 metros. [59] . . . 16
2.4 Formato de produtos Sentinel-2 nível 2A. [67] . . . 19
2.5 Exemplo de estrutura de um ficheiro HDF5. [73] . . . 21
2.6 Escalabilidade do modelo UMM. [90] . . . 25
2.7 Elementos dos principais perfis UMM. [92] . . . 26
2.8 Página inicial no catálogo Copernicus Open Access Hub. . . . 29
2.9 Apresentação dos resultados no catálogo Copernicus Open Access Hub. . . . 29
2.10 Arquitetura que serve de suporte ao Copernicus Open Access Hub. [94, p.7] 30
2.11 Arquitetura simplificada do GEE. [117] . . . 37
2.12 Diagrama de componentes do editor de código do GEE. [118] . . . 38

3.1 Arquitetura do sistema. SGBD - Sistema de Gestão de Base de Dados; SSH - *Secure Shell*. . . . 43
3.2 Diagrama de sequência simplificado, que representa o caso de sucesso de comunicação entre Orquestrador e Master. . . . 46
3.3 Diagrama de sequência simplificado, que representa o caso de sucesso de comunicação entre Master e Worker de forma a executar uma unidade de trabalho. 47
3.4 Diagrama de hierarquia entre adaptadores para fontes de dados, com representação adicional dos adaptadores por composição. OT - Obervação Terrestre. 54
3.5 Técnica de especificação hierárquica do metamodelo, acompanhada de uma possível instanciação. OAH - Open Access HUB. . . . 56

4.1 Instanciação da arquitetura do sistema, com ilustração das tecnologias usadas. 69
4.2 Representação de uma possível arquitetura de um *swarm cluster*, com um Swarm Manager e três Swarm Nodes [137] . . . 72
4.3 Arquitetura de *containers* Docker, com representação das interações entre os mesmos. . . . 73

5.1 Parcelas agrícolas detetadas na região de Salvaterra de Magos. Imagens fornecidas pelo autor da dissertação. 87
5.2 Comparação entre GHSL de 2015, à esquerda; e resultado da classificação XGBoost, à direita. Imagem fornecida pelo autor da dissertação. 92
5.3 Análise temporal do estado das FGCI correspondentes a estradas na região de Mação, recorrendo ao algoritmo Random Forest. Vermelho - intervencionadas; branco - não intervencionadas. Imagens fornecidas pelo autor da dissertação. 96
5.4 Imagem gerada através do algoritmo ESTAIR para a região de Santa Cruz da Trapa e São Cristovão de Lafões, correspondente a 02-08-2017. Imagem fornecida pelo autor da dissertação. 99

I.1 Página inicial do Data Finder. 121
I.2 Página inicial do CREODIAS Browser. 122
I.3 Visualização de produtos no CREODIAS Browser. 122
I.4 Página inicial no catálogo Onda DIAS. 123
I.5 Página inicial no catálogo Earthdata. 123
I.6 Apresentação dos resultados no catálogo Earthdata. 124
I.7 Apresentação parcial dos metadados no catálogo Earthdata. 124
I.8 Conjuntos de dados frequentemente acedidos no catálogo do GEE. [117] . . . 125

Lista de Tabelas

2.1 Resoluções de satélites. 15

3.1 Tabela de possíveis instanciações de especificação de ETL para diferentes atributos. Omissão das expressões das *queries* por serem demasiado extensas. . . 61

4.1 Tabela com o mapeamento entre as operações descritas na Secção 3.5 em *endpoints* concretos. 78

Listagens

4.1 Excerto da configuração da API de acesso a produtos do Copernicus Open Access Hub. Omissão de elementos representada por **(...)**. 74
4.2 Excerto da configuração da API de catalogação do Copernicus Open Access Hub. Omissão de elementos representada por **(...)**. 74
4.3 Excerto da especificação do metamodelo dos produtos Landsat 8 nível 1. Omissão de elementos representada por **(...)**. 75
4.4 Excerto da configuração dos índices. 76
4.5 Excerto da especificação das extrações da API de catalogação do Copernicus Open Access Hub. Omissão de elementos representada por **(...)** 77
4.6 Dockerfile a partir do qual é gerada a imagem Docker da API. 79
4.7 Especificação do Dockerfile incluída no ficheiro de *build* do SBT. 79
4.8 Excerto do `docker-compose` da arquitetura descrita na Fig 4.3. Omissão de elementos representada por **(...)** 80

Capítulo 1

Introdução

1.1 Contexto e Motivação

As plataformas de observação terrestre permitem a monitorização da Terra ininterruptamente. Na verdade, estas tecnologias de observação terrestre tornaram-se um símbolo de capacidade científica e tecnológica, de força económica e de segurança nacional. Isto levou a um crescimento no desenvolvimento de tecnologias de satélite, e por isso a um aumento do número de aplicações que recorrem a estes dados. Consequentemente, a deteção remota tem vindo a tornar-se um dos métodos mais importantes de recolha de informação sobre a superfície terrestre.

De acordo com a Union of Concerned Scientists (UCS), segundo dados de 2019 [1], em toda a história foram enviados 8378 satélites para órbita. Destes apenas 4994 estão em órbita, dos quais somente 2062 se encontram ativos. Só em 2017 e 2018, foram lançados 835 satélites. Por outro lado, 81 países já enviaram os seus próprios satélites desde o primeiro lançamento em 1962. Os países que lideram na quantidade de satélites lançados são os Estados Unidos da América (EUA) (911), a China (300) e a Rússia (154). Se se agrupar todos os países em território europeu, o número de satélites atinge os 447. É de ressalvar, contudo, que nem todos os satélites têm como missão a observação terrestre. Para este tipo de missão, o número de satélites operacionais é de 769.

As aplicações que recorrem a dados de observação são múltiplas e em áreas distintas. De forma a ilustrar a importância da deteção remota na China, numa iniciativa do ministério da agricultura, foi desenvolvido um sistema de monitorização agrícola a nível nacional [2]. Este sistema inclui módulos de monitorização de: mudanças nas áreas cultivadas; rendimento das culturas; humidade do solo; e desastres naturais. Passando à temática dos incêndios, num *survey* de 2018 [3], sobre as aplicações das imagens hiperespectrais em incêndios, é descrita a forma como a deteção remota já foi aplicada

aos diferentes estágios de incêndios: prevenção, deteção, supressão e levantamento dos danos. Mais concretamente, as aplicações de deteção remota incluem: criação de mapas de tipos de combustível [4–7], os quais são importantes, por exemplo, na modelação do movimento do incêndio; análises de risco de incêndio [8–10], o que é indispensável para a sua prevenção; deteção de incêndios ativos [11, 12], permitindo uma reação mais rápida; mapeamento de zonas queimadas [13–19] e análise da severidade dos danos [20–22], ambos indispensáveis para uma melhor canalização dos recursos; e, por fim, mapeamento da recuperação da vegetação [23–26], ajudando no planeamento florestal. Concluindo, as aplicações destes dados vão desde a agricultura e monitorização de incêndios, até outras áreas de investigação que não foram referidas, como a hidrologia, a ecologia, a meteorologia, a oceanografia, a glaciologia ou a geologia [27].

Os dados de deteção remota revelam claramente caraterísticas de *big data*: volume, velocidade e diversidade/complexidade. No que toca ao volume, de acordo com a National Aeronautics and Space Administration (NASA), o Earth Observing System Data and Information System (EOSDIS) - referido na Secção 2.1.1 - armazenava cerca de 25 petabytes, no final de 2017 [28]. Sendo que em 2020 prevê-se o seu dobro, e em 2025 cerca de 300 petabytes, ou seja, cerca de 12 vezes mais que o volume atual. Em relação à velocidade com que é feita a ingestão dos dados neste sistema (Fig. 1.1), atualmente é de cerca de 12 terabytes/dia, prevendo-se que em 2020 seja o dobro desta taxa e que em 2022 a velocidade de ingestão chegue aos 130 terabytes/dia, ou seja, 11 vezes maior que a atual [28]. Outra perspetiva a considerar na velocidade dos dados é a distribuição dos mesmos. Segundo dados de 2013 [29], o fluxo de dados entre o sistema e os seus utilizadores era cerca de 20 terabytes diários. Passando à diversidade dos dados, esta surge devido às diferentes aplicações que os dados de deteção remota têm. Estas manifestam-se em áreas distintas, como monitorização ambiental de terreno, atmosférica, hidrológica, e muitas outras. Por este motivo, os *datasets* disponibilizados são diversos, de diferentes fontes e com diferentes formatos. Estes três fatores trazem uma grande complexidade à gestão dos dados e metadados, tornando o seu acesso e integração mais difícil.

Em resposta ao rápido crescimento dos dados de observação terrestre, é particularmente importante identificar formas de extrair informação rapidamente dos dados captados pelos sensores presentes nos satélites. Os algoritmos e cadeias de processamento são múltiplos e complexos. De facto, um dos aspetos que poderá restringir o valor dos dados é justamente a velocidade com que são processados e depois disponibilizados. Segundo um *survey* de 2015, a reduzida velocidade de processamento dos dados não tem conseguido acompanhar as técnicas de recolha dos mesmos [30]. Por esta razão, é necessária uma automatização total do fluxo de processamento, o que inclui automatização do pré-processamento, processamento, armazenamento e disponibilização e, possivelmente, fusão de dados de múltiplas fontes.

Movendo o foco para o contexto, é de assinalar que esta dissertação está integrada num conjunto de dissertações que têm como temática a monitoração de recursos naturais, como referido na Secção 5.2. Dentro deste conjunto, estão presentes dissertações que produzem

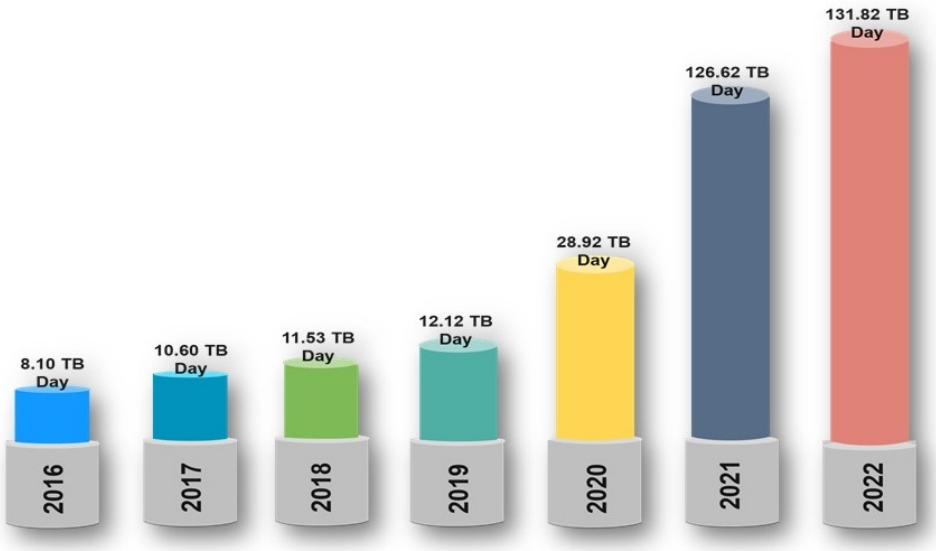


Figura 1.1: Ingestão de dados diária do EOSDIS. Informação sobre os últimos três anos e previsão até 2022. [31]

novos produtos de deteção remota, havendo portanto a possibilidade da incorporação do processamento destes no repositório proposto. Assim, existe uma integração da solução apresentada num contexto real, o que ajudará na validação da mesma, sendo um dos objetivos provar que o desenvolvimento de novos produtos de observação terrestre teria sido mais sistemático caso se tivesse recorrido a esta plataforma.

Finalmente, notar que esta dissertação foi financiada por fundos nacionais através da Fundação para a Ciência e a Tecnologia (FCT), no âmbito do projeto UID/CEC/04516/2019 (NOVA LINCS).

1.2 Problema

Como referido na Secção 1.1, os dados de deteção remota e os seus produtos suportam diversas aplicações. A verdade é que o volume e a complexidade da informação gerados pelas diferentes missões e correspondentes satélites são consideráveis. A metainformação é igualmente complexa, pois nesta está incluída a especificação de cada satélite e os dados específicos da imagem. A primeira componente de metainformação referida inclui, nomeadamente, informação sobre as várias resoluções (temporal, espacial, espectral e radiométrica); nos dados específicos das imagens, estão presentes elementos espaciotemporais, juntamente com informação sobre a qualidade destes dados.

Para além da informação originalmente recolhida pelos satélites, denominados produtos básicos, existem também produtos derivados. Dentro dos derivados surgem: índices diretamente oferecidos pelos fornecedores ou calculados por terceiros, produtos com o objetivo de melhoria espacial ou temporal das imagens originais ou índices derivados e

produtos de classificação (*e.g.* terreno, área). A título de exemplo, o CORINE Land Cover [32] é um produto de classificação, inserido no programa de monitorização da Terra do Copernicus, tendo este um papel importante no processo de atualização da cartografia de uso e ocupação de solo a nível europeu. Mais especificamente, as classes da classificação incluem: territórios artificiais, áreas agrícolas e agroflorestais, florestas, meios naturais e seminaturais, zonas húmidas ou corpos de água.

Nesta fase, fica clara a necessidade de um repositório. Na verdade, os múltiplos fornecedores já têm os seus próprios repositórios e catálogos. No entanto, existem algumas insuficiências nestes. Portanto, o objetivo desta dissertação foca-se na definição de uma arquitetura e implementação de um protótipo de um repositório de dados e metadados de observação terrestre, que responda aos seguintes desafios enumerados:

1. Heterogeneidade dos dados e metadados;
2. Extensibilidade dos metamodelos;
3. Expressividade nas interrogações;
4. Incorporação de cadeias de processamento local.

Nestes repositórios os dados e os metadados são representados em modelos e formatos padronizados. Apesar de existirem pontos em comum, estes *standards* não são transversais a todos os fornecedores. O que resulta numa maior dificuldade na sua integração e usabilidade por parte dos utilizadores. Nomeadamente, existem casos em que não existe diferença semântica de um atributo do metamodelo, mas ainda assim os diferentes fornecedores têm nomes e formatos de representação distintos. Concretizando o referido, na API do catálogo oficial do Copernicus (Secção 2.4.1), o início da data de aquisição de um produto é designado `beginposition` e é representado no seguinte formato: 2019-07-31T05:14:17.554Z; já no Earth Explorer (Secção 2.4.3), o atributo semanticamente equivalente está registado como `Start Time` e é representando da seguinte forma: 2019:212:05:14:17.5540000. Portanto, é importante definir uma camada de abstração intermédia, na qual estas diferenças sejam indistinguíveis ao utilizador.

Passando ao segundo ponto, as consequências da ausência de extensibilidade dos metamodelos subdividem-se em: dificuldades na integração de novos tipos de produtos e contrariedades ao incorporar um domínio aplicacional específico. Isto é, ao existir esta extensibilidade, caso seja adicionado um novo tipo de produto, não será necessário reformular o metamodelo, mas sim estender o existente. Por outro lado, as necessidades de metainformação da NASA não são as mesmas de uma equipa focada, por exemplo, na área agrícola, tornando-se assim importante existir a capacidade de estender os elementos de metainformação dos produtos existentes aos próprios desse domínio.

Relativamente à expressividade nas interrogações, esta refere-se à qualidade das pesquisas destes produtos de deteção remota. Esta é bastante limitada nas interfaces existentes, restringindo-se fundamentalmente apenas a elementos como a cobertura de nuvens,

o nível de processamento, o tipo de produto e os sempre presentes neste tipo de produtos: espaço e tempo. É também importante referir que o facto dos metamodelos serem extensíveis tem um papel importante na solidificação desta expressividade.

Finalmente, a possibilidade deste tipo de repositórios incorporar cadeias de processamento local facilita todo o ciclo de produção de novos produtos de observação terrestre. De momento, este tipo de desenvolvimento enquadra-se genericamente nos seguintes passos: pesquisa-se os produtos, acede-se aos mesmos, processa-se e, por fim, armazena-se os produtos computados. Relativamente à pesquisa, o problema passa pela necessidade que existe em obter produtos de diferentes fornecedores, com metamodelos distintos e disponibilizados em vários catálogos. Passando ao acesso aos mesmos, os desafios resultam: das limitações impostas nas interfaces (nomeadamente de largura de banda); das diferentes formas de acesso (*e.g.* HTTP - síncrono e assíncrono, FTP); da ausência de automatização do *download* dos produtos; e da ausência de partilha destes dados, de forma a eliminar a redundância ao descarregar estes produtos para uma infraestrutura partilhada. Tendo presentes os dados, passa-se então à fase de computação de novos produtos. Por conseguinte, surge o problema de decidir onde será armazenado e disponibilizado o produto computado. Adicionalmente, no ciclo atual de desenvolvimento, após a disponibilização de um produto, este tornar-se-ia relativamente estático, não existindo uma ferramenta de enriquecimento do seu metamodelo. Resumidamente, é fundamental que o procedimento de pesquisa, processamento, armazenamento, enriquecimento do metamodelo e disponibilização de novos produtos seja executado sistematicamente numa só plataforma.

1.3 Solução Proposta

Após a análise da literatura e do trabalho relacionado, fica evidente a complexidade e a diversidade dos sistemas de observação terrestre. Por este motivo, existem diversas abordagens possíveis para a resolução do problema formulado. Tendo em conta as alternativas, chegou-se a uma solução que é possível condensar em cinco aspetos principais: (i) automatização da ingestão de dados e metadados; (ii) especificação hierárquica do metamodelo; (iii) linguagem de especificação de *extract*, *transform*, *load* (ETL); (iv) mecanismo de interrogação de alto nível; (v) *framework* de processamento local.

Começando pela automatização da ingestão de dados e metadados, é através desta que se obtém os metadados necessários à pesquisa dos produtos de observação terrestre. Por outro lado, uma das necessidades que surge é a importância em ter localidade dos dados a processar, ou seja, a existência de mecanismos automatizados de recolha dos dados necessários à computação. Sendo estes dados de um volume assinalável, devem existir estratégias de modo a gerir os recursos existentes. Por um lado, existir a possibilidade de configurar o repositório de forma a que tenha armazenado dados com uma certa parametrização, como, por exemplo, imagens do Sentinel-2, de nível 1, no território agrícola português. Noutra perspetiva, é necessária a opção de realizar *prefetching*, isto

é, descarregar os produtos presentes no resultado da pesquisa, de forma a aplicar uma computação de seguida.

Relativamente ao segundo ponto, especificação hierárquica do metamodelo, este modo de especificação permite tirar partido da natureza hierárquica destes produtos. Na raiz da especificação surgem elementos que são comuns a qualquer produto de deteção remota, nomeadamente, os espaciotemporais. Desce-se depois para o nível do fornecedor, que para cada um dos seus produtos mantém metainformação específica do seu catálogo. Posteriormente, desce-se para o nível do programa, depois para o da plataforma, e, por fim, nas folhas desta árvore estão presentes os elementos de metainformação específicos daquele tipo de produto.

No que diz respeito à especificação de ETL, esta é feita de forma declarativa, indicando a extração que se pretende executar, a transformação que será aplicada e o mapeamento dessa extração no metamodelo. De forma a ilustrar, caso seja necessário extrair o dia do ano da data de aquisição de uma certa imagem, na especificação da extração indica-se: o ficheiro no qual é executada essa extração; a *query* da extração; a transformação que passará a data para dia do ano; e o mapeamento para o metamodelo, indicando o nome do campo correspondente.

Passando ao mecanismo de interrogação, este, para além de ter uma expressividade espaciotemporal maior do que a dos catálogos existentes, terá um maior número de dimensões pesquisáveis. Pois o facto de existirem mecanismos de extensibilidade dos metamodelos, permitirá que a expressividade das interrogações esteja constantemente a ser enriquecida. Ilustrando o referido, no caso de estudo da Secção 5.3.1, é definida uma extensão ao metamodelo do produto que agrega a classificação de culturas nas parcelas agrícolas previamente identificadas.

Para finalizar, a *framework* de processamento local visa estabelecer uma cadeia de processamento com os seguintes passos: pesquisa dos produtos necessários à computação, acesso aos mesmos, computação do novo produto, integração do produto no catálogo e um ciclo de ingestão *ad hoc* de metainformação em regime colaborativo, ao corrigir ou acrescentar elementos do metamodelo.

1.4 Contribuições

As principais contribuições desta dissertação são:

- ***Survey* dos catálogos de observação terrestre**. Focando a análise em três vertentes: cobertura, metamodelos e caraterísticas funcionais. Ao nível da cobertura são descritos os tipos de produtos disponibilizados, mas também a extensão espaciotemporal dos mesmos. Relativamente aos metamodelos, é indicado o uso de algum tipo de *standard* e discutida a pertinência dos diferentes elementos do metamodelo. Por fim, são apresentadas as caraterísticas funcionais, nomeadamente, descrevendo as interfaces de acesso aos produtos;

- **Proposta de uma arquitetura para um repositório de dados e metadados de observação terrestre**. Nesta arquitetura estão cobertas necessidades funcionais como gestão automática das cópias locais, automatização da ingestão de dados e metadados, mecanismos *ad hoc* de informação e mecanismos de interrogação de alto nível. Para além destes aspetos, são também assegurados alguns não funcionais, tais como, tolerância a falhas, escalabilidade, manutenibilidade, usabilidade e portabilidade. A tolerância a falhas é garantida através de um mecanismo de recuperação. Adicionalmente, é apresentada uma escalabilidade horizontal em grande parte das componentes do sistema, o que torna o problema da escala financeiro e não tecnológico. A extensibilidade dos metamodelos permite que a plataforma evolua sem grandes sobressaltos de manutenção. Passando à usabilidade, esta é satisfeita a partir de uma API bem documentada. Finalmente, a portabilidade é certificada através do empacotamento dos diferentes serviços do sistema recorrendo ao Docker [33];

- **Implementação de um protótipo de repositório baseado na arquitetura proposta**. Este incidirá sobre um subconjunto da arquitetura. Este protótipo estará em execução, garantindo o acesso a um conjunto de produtos de deteção remota que apenas estão acessíveis nesta plataforma, sendo estes resultantes de dissertações elaboradas em paralelo a esta;

- **Proposta de uma *framework* de processamento de produtos de observação terrestre**. O foco principal desta é criar um paradigma colaborativo na cadeia de processamento deste tipo de produtos. Para este efeito é definida uma cadeia de processamento numa só plataforma: pesquisa de produtos, acesso aos mesmos, computação de novos produtos, incorporação dos mesmos e, finalmente, um ciclo de enriquecimento do metamodelo.

1.5 Organização do Documento

O restante do documento está organizado da seguinte forma: no **Capítulo 2** são apresentados os conceitos, o trabalho relacionado e as implementações concretas de repositórios existentes que sejam relevantes para a elaboração desta dissertação. Relativamente ao **Capítulo 3**, é neste que se aprofunda a solução proposta. Passando ao **Capítulo 4**, após apresentar a solução, é neste capítulo que se descreve o desenvolvimento protótipo, o qual inclui um subconjunto representativo das potencialidades desta solução. Transitando para o **Capítulo 5**, neste é descrita a avaliação da solução, começando por apresentar a metodologia de avaliação, passando para a descrição do contexto experimental e, por fim, a formulação e análise de um conjunto de casos de estudo. Finalmente, de forma a concluir esta análise, no **Capítulo 6**, são apontadas algumas conclusões que foram recolhidas ao longo do desenvolvimento desta dissertação, juntamente com o trabalho futuro, o qual se acredita que levaria a melhorias do sistema desenvolvido.

Capítulo 2

Estado da Arte e Trabalho Relacionado

Neste capítulo são apresentados tanto o estado da arte em relação à gestão de dados de observação terrestre como o trabalho relacionado nesta mesma área.

Na Secção 2.1 está incluída uma apresentação da teoria geral de deteção remota. Começar-se-á por apresentar os princípios; passando depois para a exposição dos diferentes programas de observação terrestre e suas aplicações; e, por fim, serão analisados em maior detalhe os diferentes conceitos que influenciam a informação obtida - órbitas, sensores, resoluções e projeções.

No que diz respeito à Secção 2.2, são definidos alguns conceitos como produtos de dados, coleções e cenas. Em seguida, na Secção 2.2.1, refere-se a distinção entre produtos básicos e derivados; define-se os níveis de processamento; e apresenta-se uma instância de cadeia de processamento. Por fim, na Secção 2.2.2, é exposta a estrutura dos produtos Sentinel-2 nível 2A e Landsat 7 nível 1, analisando alguns dos formatos a que recorrem.

Relativamente aos metamodelos (Secção 2.3), são apresentados os *standards* da família ISO 19100, focando a análise no ISO 19115. Passando aos *standards* adotados pela European Space Agency (ESA), emerge a diretiva Infrastructure for Spatial Information in the European Community (INSPIRE). Já no final desta secção, é apresentado um modelo proposto pela NASA, designado Unified Metadata Model (UMM), que permite uma integração eficiente dos diferentes *standards*.

Na Secção 2.4 apresenta-se o estado da arte correspondente aos repositórios de dados e metadados de observação terrestre, onde são abordados os seguintes elementos: as iniciativas da ESA; dois catálogos americanos - um do United States Geological Survey (USGS) e outro da NASA; e, por fim, uma plataforma de uma entidade externa - a Google com o Google Earth Engine (GEE). Finalmente, na Secção 2.5, é apresentada uma visão crítica sobre os diferentes pontos referidos ao longo deste capítulo.

2.1 Deteção Remota

É importante abordar este tópico já que, durante o desenvolvimento desta dissertação, muitos foram os desafios causados pela complexidade e diversidade dos dados e metadados resultantes da observação terrestre. Tendo a dissertação como alicerce esta área, torna-se fundamental definir de forma rigorosa do que realmente se trata a deteção remota. No livro *Introduction to Remote Sensing* [34, p. 6], são analisadas uma série de definições para este conceito, sendo proposta a seguinte definição:

"Deteção remota é a prática de derivar informação sobre a superfície terrestre e aquática da Terra, usando imagens adquiridas de uma perspetiva superior que recorrem à radiação eletromagnética numa ou mais regiões do espectro eletromagnético, refletida ou emitida pela superfície da Terra."

Esta é uma definição sobretudo teórica do conceito, pois numa perspetiva mais prática, a ESA define este conceito como: "uma forma de recolher e analisar dados para obter informação sobre um objeto, sem que o instrumento usado nessa recolha esteja em contacto direto com o objeto." [35] Referem também que um elemento-chave nos sistemas de deteção remota é o modo como são geridos os dados obtidos. Um ponto fulcral para esta dissertação é justamente a forma como são processados, armazenados e disponibilizados os dados obtidos.

2.1.1 Programas

De acordo com os dados reunidos pela UCS [1], apenas 769 dos 2062 satélites operacionais que orbitam a Terra têm como objetivo a observação terrestre, ou seja, cerca de um terço do total. Estes são impulsionados por programas e missões com diferentes objetivos - apenas alguns deles são abordados nesta dissertação.

Começando pelos programas da NASA, surgem os satélites Landsat (1 a 8), Terra (EOS AM) e Aqua (EOS PM). No que diz respeito aos Landsat, apenas dois (7 e 8) dos lançados ainda continuam operacionais [36]. Em relação ao Terra e ao Aqua, ambos usam como instrumento-chave o Moderate Resolution Imaging Spectroradiometer (MODIS) [37] (caracterizado na Secção 2.1.4). As imagens obtidas pelo Terra têm aplicações como analisar a composição atmosférica, previsão meteorológica, variabilidade do clima ou monitorização de incêndios [38]; já no caso do Aqua, as aplicações incluem analisar a temperatura da superfície do mar, a humidade dos solos e a temperatura atmosférica [39]. Ainda no contexto dos Estados Unidos da América, a National Oceanic and Atmospheric Administration (NOAA) tem, entre outros, os programas Geostationary Operational Environmental Satellite (GOES) e Suomi Polar-orbiting Partnership. As imagens obtidas pelos satélites do primeiro programa são aplicadas principalmente em previsão meteorológica [40]. Já o segundo trata-se de uma parceria com a NASA, cujo objetivo é recolher imagens do terreno, dos oceanos e também medidas atmosféricas, cobrindo ainda assim os requisitos operacionais de previsão meteorológica [41].

Passando aos programas europeus, destacam-se as organizações ESA e European Organisation for the Exploitation of Meteorological Satellites (EUMETSAT). Na primeira importa destacar o programa Copernicus, neste estão incluídas as diferentes missões Sentinel: Sentinel-1 (A e B), Sentinel-2 (A e B), Sentinel-3 (A e B), Sentinel-4, Sentinel-5 Sentinel-5P e Sentinel-6. As imagens obtidas a partir de cada uma destas missões têm aplicações específicas [42]. No caso da EUMETSAT, para além de cooperarem com a ESA em algumas missões do Copernicus, também têm as suas próprias missões como a Meteosat. Esta é uma missão cujos dados obtidos em tempo real ajudam, por exemplo, na monitorização das nuvens de cinzas vulcânicas, o que se revela importante na gestão do tráfego aéreo [43]. Outra missão do EUMETSAT é o Satellite Application Facility on Land Surface Analysis (LSA SAF). Esta foca-se no desenvolvimento e processamento de produtos de satélite que caraterizam as superfícies continentais através de produtos de vegetação, evapotranspiração ou relativos a incêndios. Importa também referir que o Instituto Português do Mar e da Atmosfera (IPMA) é responsável por hospedar os produtos referidos *a priori*.

Por fim, a agência japonesa Japan Aerospace Exploration Agency (JAXA) tem como programa mais popular o ALOS. Uma das instâncias desta missão é o ALOS-2, este é utilizado para monitorizar catástrofes em todo o mundo, gerir recursos naturais, e também na área da cartografia [44].

Para além dos programas de satélite, na NASA também existe um programa com o intuito de gerir o ciclo de vida dos dados relacionados com as ciências da Terra, designado Earth Science Data Systems (ESDS) [45]. Um dos objetivos deste programa é maximizar o retorno científico dos dados obtidos noutras missões. É deste programa que surgem projetos como o Earth Science Data and Information System (ESDIS), no qual está incluído o EOSDIS [46], que providencia à NASA a gestão de dados das ciências da Terra resultantes de múltiplas fontes - satélites, aeronaves e medidas no terreno. Para além deste programa da NASA, existem iniciativas globais como o Committe on Earth Observation Satellites (CEOS) [47], cujo objetivo passa por assegurar uma coordenação internacional de programas de observação terrestre, promovendo o intercâmbio de dados entre membros. Dentro deste surgiu uma rede designada International Directory Network (IDN), que pretende assistir os investigadores a localizar informação sobre os *datasets* disponíveis. Como contribuição da NASA para esta rede foi criado o sistema Global Change Master Directory (GCMD) [48], que visa facilitar os estudos sobre as ciências da Terra e as mudanças globais.

2.1.2 Órbitas

As órbitas dos satélites classificam-se em três vertentes: altitude, excentricidade e inclinação. A altitude, ou distância entre os satélites e a superfície da Terra, determina quão rápido o satélite se move à volta da Terra; a excentricidade da órbita indica o desvio comparativamente a uma órbita perfeita, observando a Fig. 2.1, é possível analisar os

diferentes níveis de excentricidade; a inclinação indica o ângulo da órbita do satélite em relação à linha do Equador.

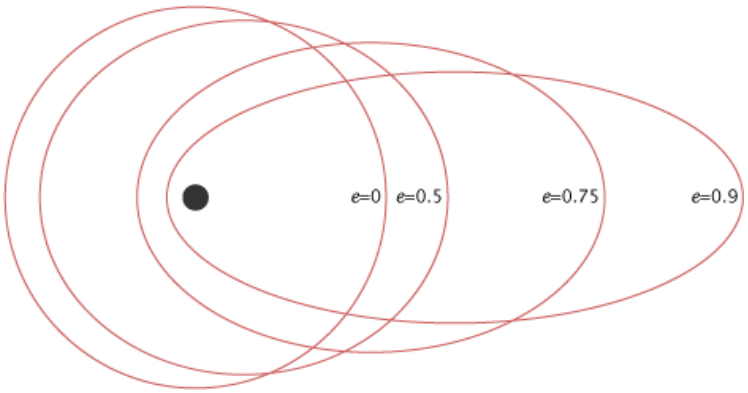


Figura 2.1: Diferentes níveis de excentricidade (e) de uma órbita. [49]

Variando as três vertentes referidas, reduz-se os tipos de órbitas em High Earth Orbits (**HEO**), Medium Earth Orbits (**MEO**), e Low Earth Orbits (**LEO**) [50]:

- **HEO**: Por ser uma órbita com maior altitude, mantém a mesma velocidade que a Terra tem durante o seu movimento de rotação. Assim, o satélite manter-se-á sempre na mesma longitude. A este tipo de órbita denomina-se geossíncrona. Adicionalmente, se a órbita tiver excentricidade e inclinação a zero (linha do equador), então terá uma órbita geoestacionária, isto é, estará sempre acima do mesmo ponto da superfície terrestre. Esta é uma caraterística importante, pois mantém uma visualização constante da mesma área. Um exemplo de um satélite que segue uma órbita geoestacionária é o GOES, permitindo enviar nova informação meteorológica na escala temporal de minutos.

- **MEO**: É uma órbita com uma altitude mais reduzida. Os seus subtipos mais notórios são as órbitas semissíncronas, isto é, demoram doze horas a completar uma órbita. Os satélites com este tipo de órbita são, principalmente, usados por sistemas de GPS.

- **LEO**: São altamente excêntricas (órbitas polares) e cobrem a Terra duas vezes por dia, uma com luz e outra na escuridão. Estas órbitas polares sempre que se cruzam com o equador têm sempre a mesma hora local, isto é, o ângulo de luz solar na superfície da Terra é consistente, apesar deste mudar com as estações do ano. Isto permite comparar imagens da mesma estação ao longo de vários anos sem se ser enviesado devido à luminosidade, criando ilusões de mudança. Os satélites com este tipo de órbita obtêm dados com aplicações em observações científicas e monitorização do clima. A título de exemplo tanto o Landsat 7 e 8 seguem uma órbita deste tipo [51].

2.1.3 Projeções

Existem diversos sistemas de coordenadas e alguns desses são importantes na forma como as imagens captadas pelos satélites são subdivididas em *tiles* ou cenas (referidas na Secção 2.2). Um dos sistemas populares é o World Geodetic System 84 (WGS84), que é usado no GPS, tratando-se este do *standard* para a geografia, cartografia e aviação. No que diz respeito à Europa, o sistema de coordenadas adotado é o European Terrestrial Reference System (ETRS); porém, o mais relevante para a área de deteção remota é o sistema Universal Transverse Mercator (UTM), e é neste que esta dissertação se irá focar. Esta relevância é observável no caso dos produtos Landsat, nos quais as cenas são recortadas de acordo com as zonas de projeção do UTM.

O UTM divide a Terra em grelhas de 6º de longitude e 8º de latitude, segmentando assim o planeta em 60 zonas verticais UTM. A verdade é que, quando se projeta bidimensionalmente usando este sistema, a distorção dentro de cada uma destas zonas é muito baixa (0,1%) [52]. De outro ponto de vista, na projeção referida, a escala fica cada vez mais distorcida à medida que a distância à linha do equador aumenta. Isto é observável na Fig. 2.2, numa projeção Mercator, a Gronelândia parece ter metade do tamanho de todo o continente sul americano, quando na verdade é nove vezes mais pequena. Concluindo, a principal vantagem deste tipo de projeção é a definição de uma grelha de zonas UTM (Fig. 2.2) dentro das quais a distorção da distância entre quaisquer dois pontos é baixa.

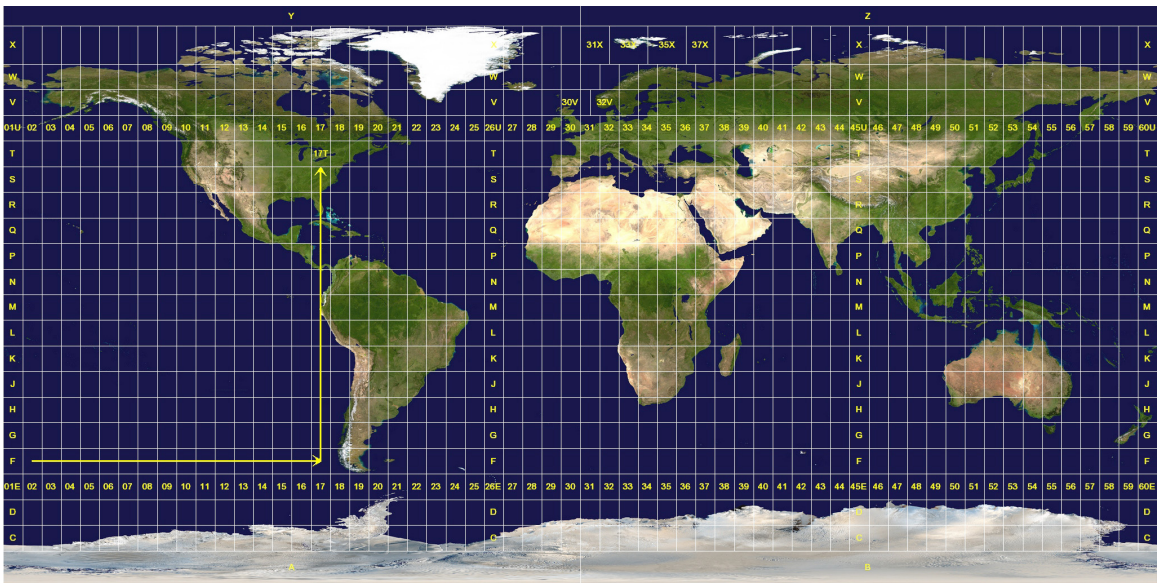


Figura 2.2: Zonas da projeção UTM. [53]

2.1.4 Sensores

Em deteção remota, os dispositivos usados para medir a radiação que chega ao instrumento do satélite são denominados sensores. Estes, muitas das vezes, têm a capacidade de obter imagens em bandas do espectro eletromagnético, para além da zona do visível.

De forma a perceber como é que estas imagens são geradas, é importante perceber que informação é possível extrair sobre os objetos a partir da radiação emitida e refletida. Estes refletem parte da luz que incide sobre eles, e esta porção geralmente fornece informação sobre a cor do objeto. Para além disto, também é emitida radiação que dá informação sobre a temperatura. Esta radiação eletromagnética é uma onda que transporta energia, e é caracterizada pelos diferentes comprimentos de onda. Assim, de forma a medir a radiação emitida e refletida pelos objetos, é necessário medir a energia nos diferentes comprimentos de onda. A título de exemplo, cada cor tem um comprimento de onda específico, visto que há comprimentos que não são visíveis ao olho humano, tais como infravermelhos, micro-ondas ou ondas rádio.

Após a recolha dos dados por parte dos sensores, são geradas imagens em que o valor de cada píxel, designado número digital, não estará calibrado em nenhum tipo de unidade física. Se o objetivo é interpretar os valores dos píxeis em termos de unidades físicas, como a radiância ou refletância, é necessário processamento adicional. A radiância trata-se da quantidade de radiação proveniente de uma área. De forma a derivar a mesma a partir do número digital, é necessário aplicar transformações que normalmente ou são especificadas pelo fornecedor ou estão incluídas nos metadados da imagem. Tipicamente, são aplicadas correções atmosféricas nas imagens de radiância, de forma a que o valor do píxel represente a refletância. A refletância representa a porção de radiação que incidiu sobre a superfície em relação à que foi refletida. Destacam-se dois tipos de refletância: Top of Atmosphere (TOA) e Bottom of Atmosphere (BOA): na primeira estão representadas as nuvens e gases atmosféricos; já a segunda não é afetada nem por nuvens nem por qualquer outra componente atmosférica [**p\IeC {\'\i }xel**].

No que diz respeito aos sensores, existem dois tipos principais [54]:

- **Ativos**: Estes iluminam os objetos que estão a ser observados, isto é, emitem radiação na direção do alvo a ser analisado. Após esta emissão, o sensor detetará e medirá a radiação refletida. As duas principais vantagens deste método são a capacidade de capturar imagens durante a noite e a filtragem de alguns dos erros provocados por nuvens ou más condições climáticas. Um exemplo deste tipo de sensor é o Radio Detection And Ranging (RADAR), que emite radiação que posteriormente é refletida pelo alvo. Esta é detetada e medida, registando-se uma estampilha temporal que permite calcular a distância ao objeto. Através desta informação é possível gerar uma imagem bidimensional da superfície. Um dos satélites que usa este tipo de sensor é o Sentinel-1, mais concretamente, usando o instrumento Synthetic Aperture Radar (SAR), o qual permite obter um mapa mais detalhado do terreno;

- **Passivos**: Estes detetam a energia que é naturalmente emitida ou refletida pela cena observada. Para este tipo de sensores, é necessário existir uma fonte de radiação, que é normalmente a luz solar. Um dos problemas desta abordagem é o facto da cobertura de nuvens poder introduzir erros, devido à reflexão da luz solar nessa mesma cobertura. Por outro lado, a vantagem é que permite detetar a radiação de

diferentes comprimentos de onda do espectro eletromagnético. Um dos satélites com este tipo de sensor é o Terra, mais concretamente com um instrumento MODIS, que tem a capacidade de obter imagens em trinta e seis bandas do espectro.

2.1.5 Resoluções

A resolução de um instrumento de um satélite tem quatro dimensões: **temporal**, **espacial**, **espectral** e **radiométrica**.

Em relação à resolução **temporal** [55], esta define-se como sendo a frequência com que um satélite revisita uma certa localização. O fator temporal é importante, nomeadamente, na obtenção de imagens de fenómenos pouco duradouros ou em comparações multitemporais.

No que diz respeito à resolução **espacial** [56], esta refere-se ao tamanho do objeto mais pequeno que consegue ser identificado, isto é, se um sensor tem uma resolução espacial de vinte metros, cada píxel representa $20m \times 20m$ de terreno.

Passando agora à resolução **espectral** [57], esta descreve a capacidade de um sensor de definir intervalos reduzidos de comprimentos de onda. Sensores com uma menor resolução não conseguirão, por exemplo, distinguir as cores nos comprimentos de onda visíveis, pois não são sensíveis, individualmente, à energia refletida pelo azul, verde e vermelho. Os sensores que captam a energia emitida em diferentes intervalos em múltiplas resoluções espectrais são denominados de sensores multiespectrais. Dentro desta categoria surgem os hiperespectrais, que detetam centenas de bandas espectrais aos longo do visível, (Near Infrared) NIR e Mid Infrared (MIR).

Por fim, a resolução **radiométrica**[58], esta quantifica a informação presente num píxel, sendo expressada em bits. Em imagens a preto e branco de câmaras digitais são de 8 bits, o que significa que a informação está restringida a 256 valores. No caso das imagens a cores têm-se três canais (vermelho, verde, azul), quantificando um total de 24 bits, ou seja, há 256 valores possíveis de cada uma das cores. Portanto, este tipo de resolução representa a sensibilidade à energia eletromagnética. De facto, quanto maior a resolução radiométrica, mais sensível será o dispositivo à deteção de pequenas diferenças na energia emitida ou refletida.

Tabela 2.1: Resoluções de satélites.

Satélite	**Temporal**	**Espacial**	**Espectral**	**Radiométrica**
[59] Sentinel-2 (2A e 2B)	5	10/20/60	13	12
[60, p.6, 49] Landsat 7	16	15/30	8	8
[61] MODIS (Terra e Aqua)	1-2	250/500/1000	36	12
Unidade	**dias**	**metros**	**nº de bandas**	**bits**

Conforme apresentado na Tabela 2.1, a resolução espacial de um instrumento é dependente de uma banda em particular. Ou seja, no caso do Sentinel-2, para bandas B1, B9

e B10, a resolução espacial é de 60m (Fig 2.3); para outras bandas, a resolução ou é de 10 ou de 20. Adicionalmente, observa-se a grande diferença de resolução temporal, espacial e espectral do MODIS em relação ao Sentinel-2 e ao Landsat 7. Isto permite fazer análises temporais mais precisas numa escala mais global. Por outro lado, a elevada resolução espectral permite que cada intervalo de bandas tenha a sua aplicação, como as bandas 24 e 25, cujo uso primário é analisar a temperatura atmosférica [62].

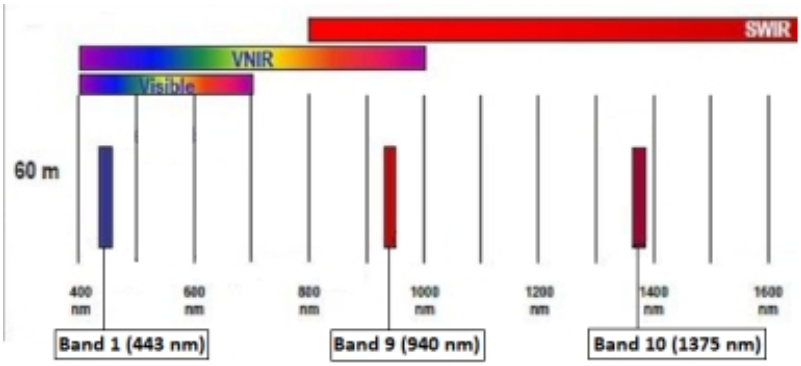


Figura 2.3: Bandas (B1, B9, B10) do Sentinel-2 com resolução espacial de 60 por 60 metros. [59]

2.2 Produtos de Deteção Remota

Os produtos de dados de deteção remota não são mais do que uma coleção de *datasets* obtidos de uma certa fonte, processados e disponibilizados ao utilizador num determinado formato. Os dados em bruto da deteção remota, obtidos através da radiação eletromagnética, contêm bastantes distorções e erros. Estes dados têm portanto de ser processados de forma a corrigi-los. Após este processamento, estes são empacotados como produto e disponibilizados aos utilizadores para as mais diversas aplicações. Existindo uma diferença assinalável entre os dados recolhidos originalmente e os disponibilizados.

Estes produtos geralmente estão organizados em coleções e cenas. Na verdade, a terminologia usada nem sempre é a mesma nas diferentes instituições. Porém, segundo a NASA, as coleções (também conhecidas por *series*) são grupos de dados que podem representar: a mesma *release* de um dado *dataset*; conjuntos de dados gerados numa experiência; uma campanha; ou até um teste a um algoritmo. Nalguns casos é possível encapsular coleções noutras, ou seja, uma coleção que é baseada em múltiplas outras. Relativamente à utilidade deste conceito, uma coleção tem informação transversal a todas as cenas que estão contidas nesta, sendo possível fatorizar esta informação comum. Passando às cenas, também designadas *datasets* e *tiles* (caso representem projeções UTM), estas são os agrupamentos de dados mínimos que conseguem ser independentemente geridos (*e.g.* descritos e disponibilizados). As cenas têm o seu próprio metamodelo e herdam os atributos definidos ao nível da coleção. Quanto à cardinalidade, as coleções contêm zero ou

mais cenas, mas uma cena é obrigatoriamente associadas a uma coleção [63]. Um exemplo do referido é a coleção Landsat 8 nível 1 que inclui as diferentes cenas obtidas a este nível. Esta organização permite uma manutenção mais fácil da qualidade dos metadados, pois não existe repetição de metadados comuns entre as cenas.

2.2.1 Cadeias de Processamento

As cadeias de processamento referem-se ao processamento que é realizado aos dados desde que são recolhidos até serem disponibilizados ao utilizador. Consoante o grau de processamento, destas cadeias resultam dois tipos de produtos: os básicos ou os derivados. Quanto aos básicos, estes têm níveis definidos de processamento. Por outro lado, os produtos cujo processamento extravasa o nível dos básicos são considerados derivados. De referir que cada passo de processamento e informação gerada pelo mesmo são registados nos metadados do produto.

Focando nos produtos básicos, a definição dos níveis de processamento deste tipo de produto não está completamente uniformizada, uma vez que cada fornecedor define estes de formas distintas. Na NASA definiu-se que cada produto está inserido num nível de processamento de 0 a 4 [64], existindo uma relação hierárquica entre eles:

- **Nível 0**: Dados reconstruídos e não processados, recolhidos diretamente do instrumento na máxima resolução. Este processamento é necessário de forma a eliminar artefactos de comunicação, como cabeçalhos de comunicações ou duplicação de dados. Os produtos neste nível de processamento não são disponibilizados, estes na verdade são processados de forma a produzir produtos de mais alto nível;

- **Nível 1**: Dados referenciados temporalmente e com metainformação acrescentada, o que inclui coeficientes de calibração radiométrica e geométrica e, por outro lado, parâmetros de georreferenciação.

- **Nível 2**: Dados com a mesma resolução e localização que os de nível 1, mas com a derivação de variáveis geofísicas. Como por exemplo, índices de concentração de gelo no mar.

- **Nível 3**: Variáveis mapeadas em escalas espaciotemporais uniformes, normalmente com alguma completude e consistência.

- **Nível 4**: *Output* de um modelo ou resultados de análises de dados de mais baixo nível, como, por exemplo, variáveis derivadas de múltiplas medições. Um dos produtos de nível 4 é o SMAP Soil Moisture, que estima a humidade ao nível do solo.

Os níveis definidos parecem bastante abstratos, portanto é importante concretizar alguns deles num caso específico, o Sentinel-2 [65]. Os tipos de produtos disponibilizados por este são do nível 1C e 2A, não tendo uma correspondência clara aos definidos pela NASA. Começando pelo nível 0, neste são dados em bruto comprimidos. Passando

ao nível 1A, é neste que os dados são descomprimidos. Adicionalmente, é desenvolvido neste nível um modelo geométrico, permitindo que qualquer píxel da imagem seja georreferenciado. Em relação ao nível 1B, é neste que são aplicadas as correções radiométricas e refinado o modelo geométrico definido em 1A. Relativamente aos produtos disponibilizados, os produtos de nível 1C são compostos por *tiles* de dimensão $100 \times 100 km^2$. Em relação às medições radiométricas deste nível, o valor do píxel (abordado na Secção 2.1.4) corresponde a uma reflexão TOA, sendo disponibilizados os parâmetros para transformar em radiância. Adicionalmente, é neste nível que são geradas máscaras para nuvens. Já no nível 2A são gerados, a partir de algoritmos de classificação de cena e correções atmosféricas, produtos de refletância BOA a partir dos de nível 1C. De assinalar que estes produtos só têm uma cobertura global desde dezembro de 2018, sendo que caso seja necessário produtos deste nível anteriores a essa data está do lado do utilizador usar ferramentas como o Sentinel-2 Toolbox [66]. Para além do processamento, esta ferramenta permite também a visualização e análise dos produtos do Sentinel-2.

2.2.2 Formatos

O formato de dados de deteção remota é normalmente selecionado com base num conjunto de fatores. Nestes estão incluídos a tecnologia de processamento e armazenamento, a distribuição do sistema e os *standards* existentes. A incompatibilidade dos formatos de diferentes fontes é um desafio na utilização destes dados. O ideal seria que todos os dados fossem gerados num formato universal aplicável a qualquer sistema, missão ou nível de produto. Um dado que contribui para a não existência desse formato é a diversidade de áreas das aplicações que as tecnologias de deteção remota suportam. Outro ponto importante é o facto das técnicas de computação e armazenamento estarem em constante evolução. No entanto, existem formatos que são reconhecidos como *standard* dentro de contextos específicos. De forma a analisar os formatos mais relevantes, serão abordados os produtos do Sentinel-2 nível 2A, Sentinel-3 e Landsat 7 nível 1. Através destes é possível demonstrar a variedade de formatos entre fornecedores. Adicionalmente, mesmo dentro da missão Copernicus, dado que os diferentes Sentinel tem diferentes aplicações, fica clara a multitude de formatos de representação dos produtos.

Sentinel-2 nível 2A. Começando pelos produtos do Sentinel-2 nível 2A, estes são disponibilizados num formato designado SENTINEL-SAFE. Tendo sido este desenhado para atuar como um formato comum de armazenamento e transferência dentro das infraestruturas de observação terrestre da ESA [67]. Focando num caso concreto, os produtos nível 2A do Sentinel-2 são organizados numa diretoria (Fig. 2.4) que inclui na sua raiz: um ficheiro designado manifest.safe (XML), que guarda informação genérica do produto em XML; uma imagem de pré-visualização em formato JPEG2000; o ficheiro INSPIRE.xml (Secção 2.3.2); uma subdiretoria com a definição do esquema XML; e uma subdiretoria com os *tiles*. Descendo ao nível da diretoria dos *tiles*, esta está subdividida em dados

auxiliares, imagens e indicadores de qualidade. Em relação às imagens, estas são disponibilizadas em formato GML-JPEG2000 com diferentes resoluções espectrais e espaciais. No que toca aos indicadores de qualidade, são disponibilizados ficheiros em formato XML que representam a qualidade do produto, do formato, da geometria, da radiometria e do sensor. Ao nível do píxel [68, p .70-71], em formato GML, para cada banda são disponibilizadas máscaras de defeitos em píxeis, de ausência de dados, de píxeis saturados e de polígonos que indicam as áreas da imagem com qualidade degradada. Adicionalmente, em formato JPEG2000, são apresentadas máscaras de probabilidade de nuvens e de neve, nas quais cada píxel representa a probabilidade de existir, respetivamente, nuvens e neve.

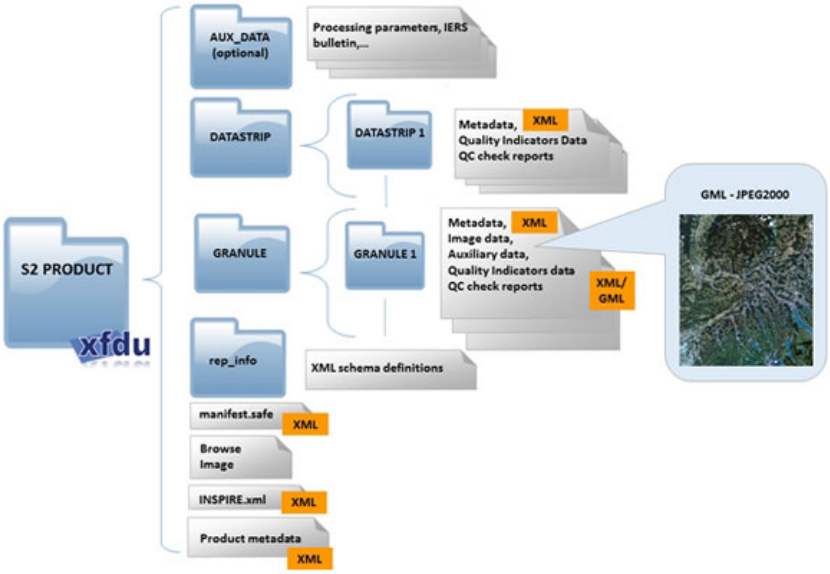


Figura 2.4: Formato de produtos Sentinel-2 nível 2A. [67]

No contexto da análise aos produtos nível 2A do Sentinel-2 foram referidos dois formatos que são importantes clarificar: JPEG2000 e GML. O primeiro trata-se de um sistema de codificação e compressão de imagem, as caraterísticas principais deste formato são a sua escalabilidade, compressão sem grandes custos na qualidade, capacidade de aceder a diferentes níveis de resolução e qualidade e integração de metadados em formato XML. No que diz respeito ao GML, este é uma gramática XML para transporte e armazenamento de informação geográfica que inclui tanto propriedades espaciotemporais como outras. Sendo este uma gramática XML, é possível inclui-lo num ficheiro JPEG2000 [69]. Neste contexto, o GML assume um papel importante na georreferenciação, geometria e radiometria da imagem, entre outros metadados.

Sentinel-3. Sob outra perspetiva, nos produtos do Sentinel-3 estão presentes um conjunto de ficheiros com extensão *.nc*, juntamente com o ficheiro XML manifest. Este último consiste na informação genérica do produto e processamento, tal como no Sentinel-2. Em relação aos ficheiros das medições e anotações, estes estão escritos em formato Network

Common Data Form (netCDF) e incluem dimensões, variáveis e atributos associados. Dependendo do instrumento usado na medição, as medidas obtidas pelos Sentinel-3 (A, B e C) são usadas, por exemplo, na topografia dos oceanos, na avaliação do vapor de água ou na radiação térmica emitida pela Terra. Sendo estes dados tão dimensionais, o formato netCDF tem um papel importante, pois o seu modelo é destinado a dados científicos em matriz. De facto, este é um formato popular para modelos atmosféricos e geofísicos (*e.g.* gravidade, barometria, magnetismo). Complementarmente, os produtos que recorrem a este formato têm as seguintes caraterísticas: autodescritivos, pois incluem metainformação; portáveis, pois são independentes da máquina; acesso aleatório, permitindo aceder a subconjuntos dos dados de uma forma eficiente; anexáveis, permitindo que os dados sejam anexados sem que seja necessário redefinir a sua estrutura; e, por fim, partilháveis, existindo a possibilidade de múltiplas escritas e leituras concorrentes.

Landsat 7 nível 1. Passsando à análise dos produtos de dados do Landsat 7 nível 1, estes são representados em formatos como FAST-L7A, GeoTIFF ou HDF-EOS5 [70, p. 5].

O FAST-L7A é um formato de representação de dados e metadados de um *dataset*, tendo sido criado especificamente para os dados do Landsat 7.

Em relação ao formato GeoTIFF [71], este define um conjunto de *tags* que descrevem a informação cartográfica associada a uma imagem *raster* TIFF que tenha como origem, por exemplo, um satélite. Este tem como objetivo principal associar a uma imagem *raster* uma projeção, descrevendo-a. Na verdade, este formato não permite substituir os *standards* de metadados existentes, mas enriquecer um formato popular de imagens *raster* (TIFF) com georreferenciação.

Finalmente, o HDF-EOS5 foi desenhado para suportar os dados científicos do sistema EOS. Este é baseado no formato HDF5, o qual consiste num modelo de dados, formato de ficheiro e biblioteca de I/O. Tendo sido delineado para armazenar, transferir, gerir e arquivar dados complexos [72, p. 4-5]. Este formato vem portanto responder à necessidade de lidar com conjuntos de dados volumosos, com variedade nos tipos e estruturas, e com diversos metamodelos. Voltando ao modelo de dados, este disponibiliza estruturas e operações que permitem a criação, armazenamento e acesso de praticamente qualquer estrutura de dados ou coleção de estruturas. O modelo inclui: uma classe ficheiro; duas principais classes, *datasets* e grupos; algumas classes de suporte como atributos e tipos de dados; e os metadados. Na Fig. 2.5 está ilustrada a forma hierárquica como se relacionam as classes referidas. Neste exemplo, o ficheiro (raiz) é constituído por uma árvore de grupos, sendo que nas folhas estão sempre presentes os objetos indivisíveis: *datasets*. De realçar que cada um dos objetos é acompanhado de metadados, fornecendo assim um mecanismo de herança dos mesmos. O modelo de dados descrito foi estendido de forma a originar o definido no HDF-EOS5. Neste último são concretizados os conceitos do HDF5 no domínio da deteção remota.

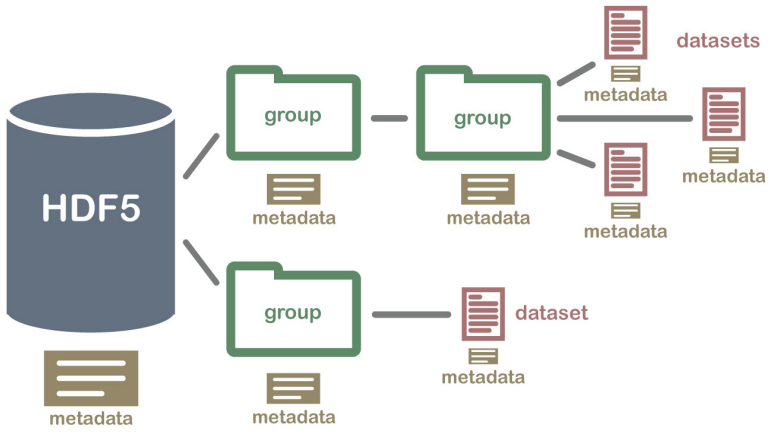


Figura 2.5: Exemplo de estrutura de um ficheiro HDF5. [73]

2.3 Metamodelos

Os metadados são informação estruturada de acordo com um metamodelo. Esta metainformação permite obter informação acerca dos dados sem ter conhecimento total do seu conteúdo. Os benefícios da existência de metadados incluem uma gestão facilitada dos dados, uma melhoria na qualidade dos mesmos e a garantia de interoperabilidade. Em relação à primeira vantagem referida, os metadados ajudam na gestão de grandes *datasets* das seguintes formas: organizando os dados; reduzindo o risco de duplicação; e, por outro lado, tornando mais eficiente a pesquisa em grandes volumes de dados. Por outro lado, ajuda a garantir a qualidade dos dados pois esta, sendo definida por um conjunto de *standards*, possibilita a automatização do controlo da qualidade dos dados. Por fim, um ponto muito importante é o facto dos *standards* de metadados permitirem uma partilha de recursos entre organizações, melhorando a interoperabilidade entre as mesmas [74, p.8-9].

De facto, os produtos são acompanhados de metadados. Estes existem tanto a nível da coleção como da cena. No que diz respeito à coleção, os elementos dos metadados que a descrevem são transversais a todos os membros da mesma. Os metadados a este nível podem incluir informação: de pesquisa (*e.g.* palavras-chave, *abstract*, contactos); da plataforma; de caraterísticas do sensor e instrumento; e de informação geral sobre a linhagem (fontes de dados e passos de processamento). No que diz respeito às cenas, estas têm os seus próprios metadados e herdam os atributos definidos a nível da coleção. Os metadados presentes neste nível incluem informação espaciotemporal e de qualidade dos dados [75].

2.3.1 ISO 19115

O *standard* ISO 19115 tem sido o bloco fundacional na modelação de metadados de informação geográfica. A verdade é que este é o *standard* mais importante na definição

do metamodelo dos dados de deteção remota. Este surge de um conjunto de *standards* para informação geográfica (Série ISO 19100). Dentro deste conjunto surgem *standards* para descrever dados espaciotemporais (ISO 19107 e ISO 19108), serviços de informação geográfica (ISO 19119) e um formato (GML - Secção 2.2.2) com base em XML, usado na transmissão de dados espaciais (ISO 19136) [74]. Voltando ao ISO 19115, este define o esquema necessário para descrever informação e serviços geográficos. A informação que providencia inclui: a identificação, a extensão geográfica, a qualidade e os aspetos espaciotemporais. Esta informação ajuda na pesquisa, seleção, acesso, transferência e uso dos dados [76]. Segundo a NASA, de forma a garantir conformidade com o ISO 19115, é necessário considerar os seguintes *standards* [77]:

- **ISO 19115-1**: trata-se de uma revisão do ISO 19115. Todos os *standards* estão sujeitos a mudanças, pois apesar de todo o esforço na abrangência das suas definições, a tecnologia evolui e o metamodelo deve ser estendido e às vezes corrigido;
- **ISO 19115-2**: é uma extensão que define o esquema dos metadados que descreve as imagens e os dados em grelha. Este *standard* disponibiliza informação sobre os equipamentos de medição usados, as propriedades do sistema de medição e sobre os métodos numéricos e procedimentos computacionais na digitalização dos dados em bruto [78];
- **ISO 19115-3 e 19139**: descrevem o procedimento usado para gerar o esquema XML a partir dos modelos conceptuais definidos pelos *standards* ISO 19115-1 e 19115-2 [79, 80];
- **ISO 19157**: este estabelece os princípios para descrever a qualidade de dados geográficos, definindo os componentes necessários para a descrição da qualidade dos dados. Para este efeito, especificam-se os componentes para registar as medidas efetuadas e componentes que descrevem o procedimento geral, não definindo qualquer mínimo aceitável de qualidade destes dados [81].
- **ISO 19156**: este define um modelo de transferência de informação que descreve atos de observação e os seus resultados para diferentes comunidades científicas [82]. Juntamente com o *standard* Open Geospatial Consortium (OGC) 10-025r1 define um perfil para descrever produtos de observação terrestre [83]. Este fornece um esquema *standard* de metadados dos produtos de observação terrestre. Os metamodelos dos produtos são subdivididos em temáticas (*e.g.* ótica, radar, atmosférica,...), pois os produtos resultantes destas são bastante diferentes entre si. Adicionalmente, neste *standard* são descritos os mecanismos que estendem estes esquemas para aplicações específicas.

Analisando mais em detalhe o ISO 19115, este tem 22 elementos de metadados principais. Estes dividem-se em três categorias: obrigatórios, opcionais e condicionais. O

significado das duas primeiras é bastante claro, mas quanto aos elementos condicionais, estes são elementos que se tornam obrigatórios caso certos pré-requisitos sejam cumpridos. Quanto aos elementos obrigatórios, estes são apenas sete dos 22 principais, sendo estes maioritariamente de identificação do *dataset* (*e.g.* título, tópico, *abstract*, contactos...). Nos elementos condicionais e opcionais surgem, por exemplo, a localização geográfica, a resolução espacial, o identificador do ficheiro, a linhagem, a localização *online* do recurso, o formato de distribuição [74]. Estes elementos considerados não obrigatórios são de importância assinalável na área da deteção remota.

2.3.2 INSPIRE

A diretiva INSPIRE tem como objetivo a criação de uma infraestrutura de dados espaciais no contexto da União Europeia. Esta permite a partilha de informação espacial entre as organizações, facilita o acesso público e a extração de informação que suporte decisões políticas [84]. No contexto dos metadados que descrevem informação espacial, o *standard* no qual se inspira esta diretiva é o ISO 19115. Sendo que, segundo a diretiva, todos os elementos de metadados definidos nesta podem ser expressos pelo ISO 19115. Apesar disso, a conformidade com a INSPIRE não garante conformidade total com ISO 19115 [85, p. 10-13].

Na análise do metamodelo definido por esta diretiva [85, p. 15-60], para cada uma destas categorias usar-se-á a letra M para os elementos de metadados obrigatórios e C para os condicionais. Adicionalmente, analisar-se-á a obrigação da presença dos elementos na perspetiva dos tipos de recurso serem coleções ou cenas, excluindo desta análise os serviços.

No que diz respeito à identificação, os elementos que se destacam são: título (M), resumo (M), tipo de recurso (M), URL (C), identificador único (M) e linguagem do recurso (C). De notar que o URL é condicional, pois este pode não estar disponível, mas caso esteja é obrigatório. Em relação à linguagem do recurso, esta só é obrigatória caso o recurso inclua informação textual. Passando à classificação de dados espaciais, destaca-se o tópico (M) que representa o principal tema do recurso. Quanto às palavras-chave, surgem o valor (M) da mesma e o vocabulário (C). Este último só é necessário caso alguma das palavras seja originária de um vocabulário controlado (*e.g.* ontologia). Em relação à localização geográfica, o elemento que se destaca é a área geográfica (M), que define uma delimitação geográfica do recurso. Para além da referência geográfica também existe a temporal, sendo os elementos presentes os seguintes: extensão temporal (M), data de publicação (C), criação (C) e revisão (C). No que diz respeito à extensão temporal, esta pode ser definida por data individuais, intervalos, ou uma mistura destas duas. Quanto à condicionalidade dos outros elementos, esta deve-se à obrigatoriedade de pelo menos uma das três ser definida. Outro aspeto importante é a qualidade dos dados, na qual surgem a linhagem (M) e a resolução temporal (C). A primeira trata-se de uma declaração sobre o histórico do processamento e a qualidade geral do recurso; a segunda é apenas

condicional, mas torna-se obrigatória caso seja possível especificá-la. Por outro lado, a conformidade surge com um papel na interoperabilidade dos sistemas. É nesta temática que surge a especificação (M) e o grau (M). A primeira é uma referência aos requisitos usados como base avaliativa da conformidade dos dados. Já o grau é o elemento que indica se foram cumpridos os requisitos de conformidade definidos na especificação. Passando aos termos legais, são definidas as limitações (M) e as condições (M) ao acesso público, a organização responsável pelos dados e a sua respetiva função. Por fim, são definidos os metadados dos metadados, isto é, o contacto dos responsáveis, a data em que estes foram criados e a língua em que estes foram expressos.

Com base nesta análise é possível depreender que este é um *standard* bastante mais rigoroso e extensivo que o ISO 19115, no qual se inspira. Pela presença das caraterísticas referidas anteriormente, a sua aplicação é mais difícil. Em 2017, foi desenvolvido um relatório [86, p.15-16] que avalia o estado da implementação da diretiva INSPIRE em países da União Europeia. Neste conclui-se que tem existido um aumento constante de documentação sobre os dados, sendo que no total 87% dos metadados de produtos de organizações europeias está de acordo com o metamodelo definido por esta diretiva.

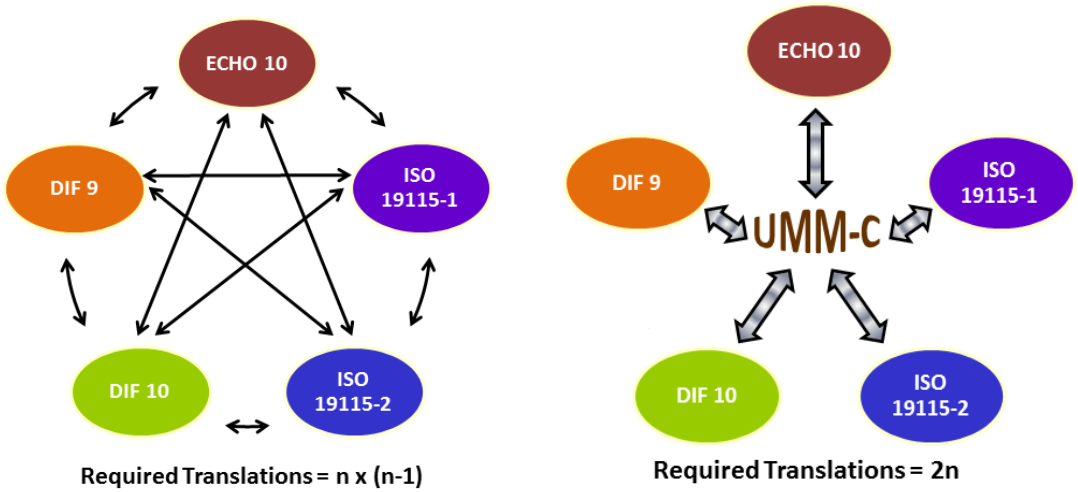
2.3.3 UMM

Este modelo surgiu no contexto da criação de um repositório comum de metadados (CMR) - Secção 2.4.4. Os metadados presentes no CMR são disponibilizados por diferentes fornecedores. Por este motivo é necessário suportar uma variedade de *standards* de metadados que incluem: DIF 9, DIF 10, ECHO 10, SERF, ISO 19115-1 e ISO 19115-2. Quanto aos DIF [87] e SERF [88], estes são usados no GCMD. Os DIF são formatos de intercâmbio de diretorias para metadados de coleções. O SERF é um *standard* de metadados que visa a descrever serviços e aplicações. Em relação ao ECHO 10, este é um formato de metadados de coleções e cenas. Por fim, os *standards* ISO que são definidos mais em detalhe na Secção 2.3.1.

Havendo esta necessidade de suportar todos estes *standards*, os criadores do CMR consideraram standardizar todos os metadados para um formato que oferecesse garantias de interoperabilidade - ISO 19115. No entanto, os custos de converter todos os sistemas para que estes tivessem a capacidade de gerar metadados de acordo com este *standard* eram elevados. Portanto, a solução acabou por ser continuar a suportar os múltiplos *standards*, desenvolvendo-se um método de tradução entre estes. Foi desta premissa que surgiu o UMM [89], um metamodelo extensível que disponibiliza uma *framework* de tradução entre *standards* suportados pelo CMR.

A caraterística distintiva do UMM é o facto de, em vez de mapeamentos entre todos os *standards*, cada um destes é mapeado para um UMM, que facilmente poderá ser de novo traduzido para qualquer outro *standard*. Este mecanismo é bem ilustrado na Fig. 2.6, na qual se percebe a escalabilidade da solução. Sendo n o número de *standards* de metadados suportados pelo CMR, na ausência deste método ter-se-ia que definir n × (n-1) traduções, o

que representa todas as combinações entre *standards*(Fig. 2.6a). Com este método, apenas seriam necessárias 2n (Fig. 2.6b).



(a) Sem UMM

(b) Com UMM

Figura 2.6: Escalabilidade do modelo UMM. [90]

O UMM define perfis de metadados correspondentes a conceitos chave como coleções ou cenas. Cada um destes perfis define o esquema dos elementos necessários de forma a assegurar metadados de qualidade. A definição de um elemento de um perfil UMM inclui: mapeamento dos *standards* de metadados suportados pelo CMR; restrições aos seus valores; descrição; cardinalidade; possíveis conflitos no processo de unificação; e futuras recomendações.

Os perfis definidos atualmente são as coleções (UMM-C), cenas (UMM-G), serviços (UMM-S), variáveis dos produtos de dados (UMM-Var), visualizações (UMM-Vis), metametadados (UMM-M), e um perfil transversal a todos estes (UMM-Common). As coleções e cenas já foram definidas na Secção 2.2; os serviços representam informação específica de uma área de estudo; o UMM-M define metamodelos que permitam definir a qualidade dos metadados; e por fim o UMM-Vis representa metadados dos produtos de visualização criados a partir das coleções ou cenas. De todos os referidos, apenas o UMM-C, UMM-G, UMM-S e UMM-Common foram implementados [91] até à data. Na Fig. 2.7 observa-se os elementos genéricos dos três primeiros perfis referidos como já implementados. Nesta estão ilustradas as diferenças em termos de metamodelos das coleções, cenas e serviços. A título de exemplo, as coleções têm palavras-chave de forma a facilitar a sua pesquisa, por outro lado, as cenas têm informação sobre a qualidade dos dados, e por fim, os serviços, não apresentam informação espaciotemporal.

Em relação à evolução do UMM, este é revisto pelo menos uma vez por ano. As mudanças neste são inevitáveis, pois a qualidade dos metadados é um aspeto crítico. De forma a assegurar esta qualidade são necessárias estas atualizações periódicas de forma

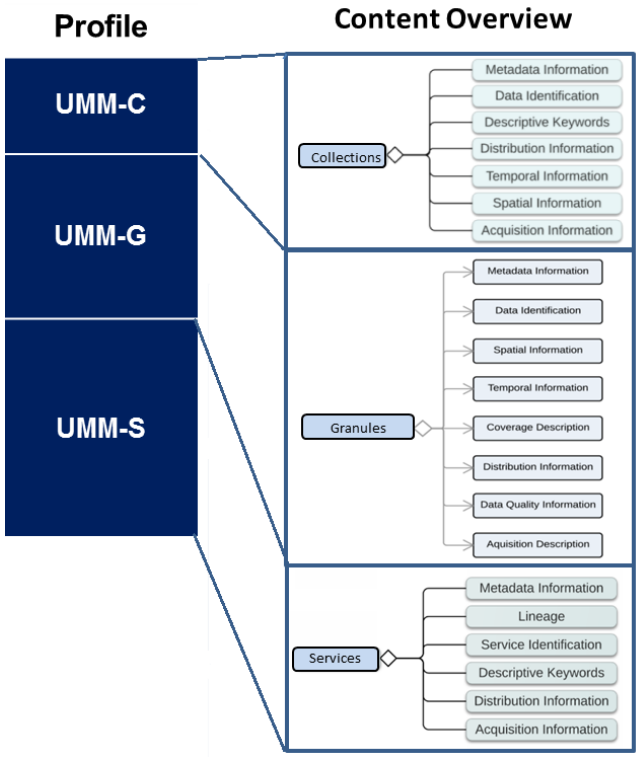


Figura 2.7: Elementos dos principais perfis UMM. [92]

a corresponder aos interesses dos utilizadores. O reverso deste raciocínio é o facto destas mudanças poderem ter um impacto negativo na compatibilidade de certos sistemas. Portanto, tem de ser feita uma gestão cuidadosa deste *trade-off*.

2.4 Catálogos e Plataformas

A forma como os fornecedores disponibilizam os seus produtos tem um papel importante na utilização dos mesmos pela comunidade. Esta é feita através de catálogos, nos quais é possível pesquisar e visualizar os produtos, e por fim, aceder aos mesmos. No processo de pesquisa é indispensável que as interrogações sejam suficientemente expressivas e eficientes. A visualização tem um papel importante no refinamento dos resultados obtidos na pesquisa. E por fim, é essencial que existam mecanismos eficientes de acesso aos produtos. Normalmente, associada à interface gráfica, existem API que ajudam na automatização do processo referido anteriormente. Noutra perspetiva, existem catálogos incorporados em plataformas de processamento na *cloud*. Uma das vantagens desta abordagem é reduzir os recursos despendidos no *download* dos produtos, pois o processamento será realizado na infraestrutura em que estão armazenados.

Em termos europeus surgem destacados o Copernicus Open Access Hub e as plataformas DIAS. Nos EUA, a NASA disponibiliza o catálogo Earth Data e a USGS disponibiliza

o Earth Explorer. Adicionalmente, existem alternativas privadas como o EOS Land Viewer [93]. Este está incluído numa plataforma que disponibiliza também serviços de armazenamento, processamento e visualização. Ainda no contexto de iniciativas privadas, a Amazon e Google, associada às suas soluções de *cloud* para deteção remota, também disponibilizam os seus próprios catálogos.

De forma a perceber o que está envolvido na construção de um repositório, é importante estudar alguns destes catálogos e plataformas. Cada um dos analisados serão abordados principalmente em três vertentes: cobertura e âmbito; metamodelos; e características funcionais que estes promovem, tanto ao nível da interface gráfica como das API que disponibilizam.

2.4.1 Copernicus Open Access Hub

O sistema de acesso aos dados do Sentinel recolhe os produtos automaticamente dos segmentos responsáveis por gerá-los (Payload Data Ground Segment - PDGS). De seguida, publica-os *online*, num conjunto de pontos de disseminação designados *hubs*. Acedendo a estes, os utilizadores poderão explorar as coleções e cenas através de uma interface gráfica ou de *scripting*. Neste momento existem quatro *hubs* [94, p.6-8]: Copernicus Open Access Hub (Open Hub), Copernicus Services Hub (ServHub), Collaborative Hub (ColHub), International Hub (IntHub).

O primeiro oferece acesso grátis aos dados do Sentinel numa base de registo pessoal. Devido ao elevado número de utilizadores ativos e à necessidade de assegurar larguras de banda satisfatórias, o número de *downloads* concorrentes está limitado a dois. Por outro lado, não existem restrições de retenção no acesso aos produtos. Isto é, não existe um período depois do qual estes deixam de estar disponíveis. No caso do ServHub, este garante acesso livre aos serviços Copernicus (atmosfera, mar, terreno, clima, segurança e emergência). Estes serviços disponibilizam produtos especializados nos domínios referidos. De forma a ilustrar, no caso do serviço de emergência, este complementa a informação derivada das imagens de satélite com dados *in-situ*. Sendo esta providenciada aos atores envolvidos numa gestão de uma determinada crise. Relativamente aos pormenores técnicos, neste é possível realizar dez *downloads* concorrentes, não existindo também nenhum período de retenção.

Passando ao ColHub, este é aberto a estados-membro do Copernicus. Os downloads concorrentes são no máximo dez, sendo que neste caso já existem períodos de retenção entre catorze a trinta dias.

Por fim, o IntHub está aberto a parceiros internacionais (*e.g.* NASA, NOAA, USGS). O número máximo de *downloads* concorrentes é de 10, e os períodos de retenção são de trinta dias. Estas políticas de retenção têm que ver com a suposição de que os produtos serão redistribuídos a partir da infraestrutura da entidade que efetuou o *download*. De referir que os *hubs* listados são operacionalizados pela ESA e incluem todos os produtos do Sentinel, exceto os produtos marinhos do Sentinel-3 nível 1 e 2. Estes são disponibilizados

através do serviço Copernicus Online Data Access (CODA), que é um dos catálogos da EUMETSAT.

Em Portugal, foi criado um catálogo baseado no Open Hub, designado IPSentinel[95]. Este armazena e disponibiliza, na sua infraestrutura, dados dos satélites Sentinel relativos ao território português, incluindo a área de responsabilidade de busca e salvamento no Atlântico. Uma das vantagens da utilização deste catálogo é o facto de permitir o acesso mais rápido aos dados do Sentinel-1. Isto deve-se à presença de uma estação em território nacional que é a primeira a obter dados do território nacional que apenas mais tarde serão disponibilizados nos repositórios da ESA. A implementação da infraestrutura foi desenvolvida através de uma estreita parceria entre a Direção-Geral do Território (DGT) e o IPMA. Relativamente ao acesso ao catálogo, este é homólogo aos outros *hubs*, já que se trata de uma adaptação do código aberto disponibilizado pela ESA.

Focando no Copernicus Open Access Hub [96], os produtos disponibilizados são: os de nível 0, 1 e 2 do Sentinel-1; os de nível 1C e 2A do Sentinel-2; os de nível 1 e 2 do Sentinel-3 (excluindo os produtos marinhos); e para concluir, os de nível 1B e 2 do Sentinel-5P [97]. De forma a retratar a importância deste repositório é importante apresentar algumas estatísticas [94, p. 20, 30]. Segundo dados de 2017, estão armazenados 4.81 petabytes neste *hub*, sendo que destes: 63% correspondem a dados Sentinel-1; 29% a dados do Sentinel-2; e apenas 8% a dados do Sentinel-3. Em relação à velocidade, no mês de novembro de 2017, a média de dados ingeridos por dia foi de 5,82 terabytes. No que diz respeito à disponibilização destes dados, em 2017, os 110 mil utilizadores registados totalizaram 28 petabytes de *downloads*, representando isto três vezes o volume do ano anterior.

Em termos funcionais, o catálogo permite o acesso aos dados através de duas principais componentes: via interface gráfica e através de API REST. Começando pelas funcionalidades da primeira componente referida, ao pesquisar existem dois elementos principais na interface (Fig. 2.8): o mapa e a barra de pesquisa. Para além destas componentes de acesso, também é disponibilizada uma vista do catálogo em CSV.

O primeiro permite selecionar a área geográfica de interesse e, para ajudar na seleção da mesma, é possível visualizar diferentes camadas do mapa (*e.g.* Open Street Maps, Sentinel-2 sem nuvens). De realçar que existem dois modos de interação com o mapa: navegação, no qual é possível ajustar a área visível do mapa; desenho, no qual é possível desenhar a área geográfica, através da qual se pretende filtrar a pesquisa.

No caso da barra de pesquisa, esta permite interrogar o repositório. Nestas pesquisas é possível filtrar o nível do produto, o instrumento, o modo de aquisição, o nome da plataforma, a data inicial e final da recolha dos dados obtidos via satélite, a data de ingestão no *hub*, a coleção, o nome do ficheiro que representa o produto, a área geográfica, a órbita, o modo do sensor, o tipo do produto, a classificação temporal (tempo real ou não), e a percentagem de nuvens. Adicionalmente, é permitido indicar um atributo segundo o qual serão ordenados os resultados desta pesquisa e, caso seja necessário combinar alguns destes parâmetros, recorre-se aos normais operadores lógicos (AND, OR e NOT) e

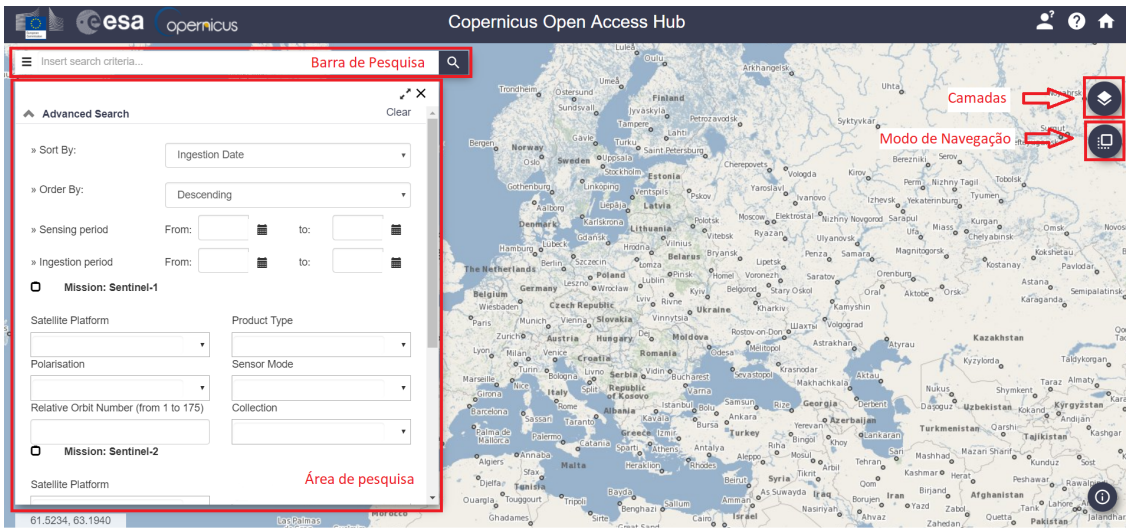


Figura 2.8: Página inicial no catálogo Copernicus Open Access Hub.

a simples expressões regulares. Após a pesquisa, é exposta a lista de resultados (Fig. 2.9), e para cada produto é apresentada uma pequena imagem, o nome do produto, o URL, a data de início da recolha, o nome do instrumento, o nome do satélite, tamanho do produto e, no mapa, a área geográfica correspondente ao produto. Nesta fase, as opções são as seguintes: visualizar os detalhes do produto, fazer *zoom* na área do mesmo, *download*, remover dos resultados da pesquisa ou juntá-los a um conjunto máximo de 100 produtos, de forma a descarregá-los em *batch*. Selecionando a opção da visualização dos detalhes dos produtos, são apresentados os metadados (relativos ao produto, à plataforma e ao instrumento) e a estrutura do produto.

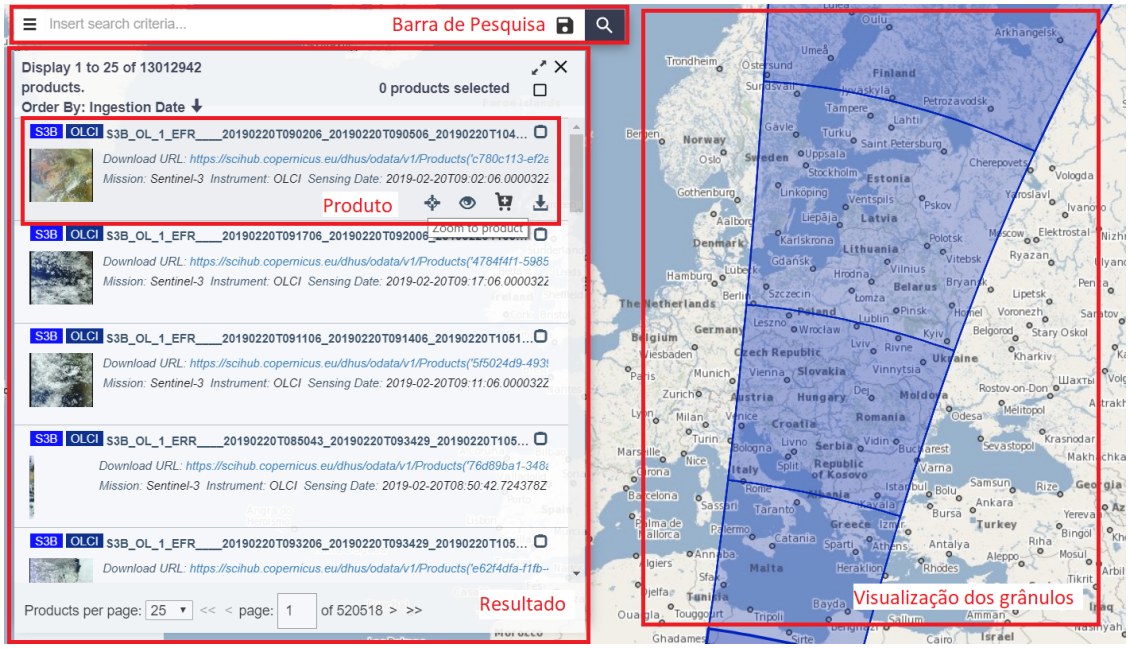


Figura 2.9: Apresentação dos resultados no catálogo Copernicus Open Access Hub.

Relativamente às API, este *hub* expõe duas: uma recorrendo a Open Data Protocol (OData) e outra a OpenSearch (Solr). O primeiro [98] trata-se de um *standard* que define o conjunto das melhores práticas na definição e consumo de RESTful API. No contexto do Copernicus Open Access Hub, este aceita a pesquisa e *download* dos produtos, permitindo o desenvolvimento de *scripts* que façam o *download* em *batch*. Já o Solr [99] trata-se de um sistema de pesquisa. Este é uma tecnologia complementar ao OData e, de facto, pode servir como motor de pesquisa da API REST implementada segundo o *standard* OData. Ou seja, a especificação dos *endpoints* segue as linhas do ODATA, mas o sistema que suporta as pesquisas resultantes da interação com os utilizadores é o Solr.

Adicionalmente, o OData permite selecionar o formato em que é apresentado o resultado, tendo como opção: atom, XML, json, metalink e CSV. Outra funcionalidade interessante é o facto de permitir descarregar ficheiros específicos dos produtos. Um exemplo de utilização desta funcionalidade é se apenas se pretender descarregar a imagem de uma resolução espectral específica.

Por fim, na Fig. 2.10 está representada a arquitetura do sistema que suporta este catálogo. Nesta é observável que os produtos são automaticamente recolhidos dos segmentos terrestres do Sentinel. Posteriormente, são pré-processados de maneira a serem ingeridos no repositório. Tendo estes dados armazenados, estes são publicados nos diferentes *hubs*, permitindo o acesso por parte dos seus utilizadores. Através da flexibilidade desta arquitetura, é possível expandir a configuração dos *hubs* de forma a acomodar os diferentes requisitos operacionais das múltiplas partes interessadas no Copernicus.

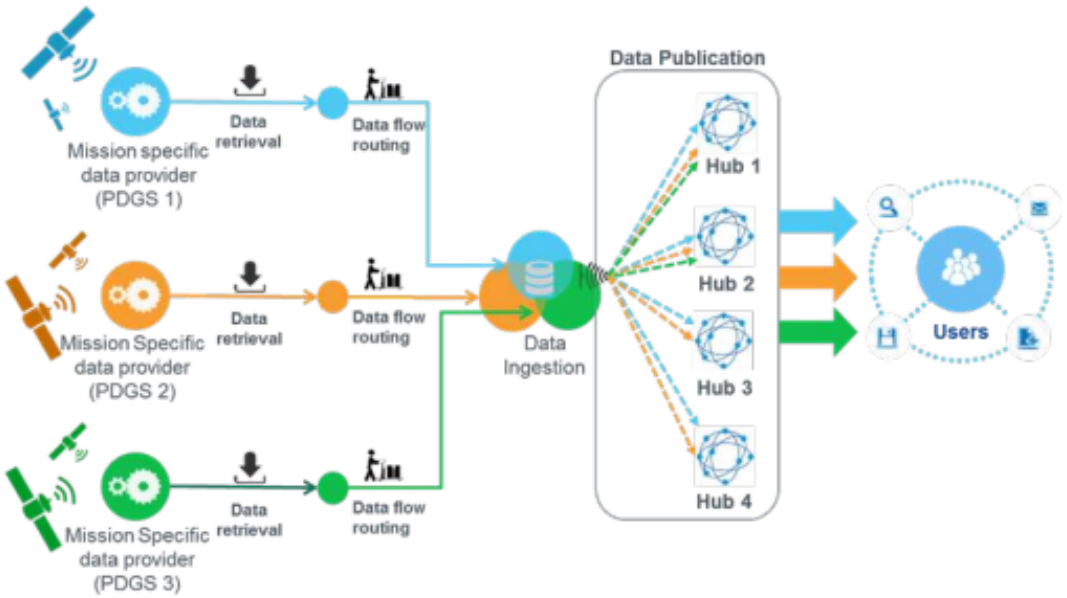


Figura 2.10: Arquitetura que serve de suporte ao Copernicus Open Access Hub. [94, p.7]

2.4.2 DIAS

No programa Copernicus, os dados originários das diferentes missões eram disponibilizados em plataformas distintas. Alguns dos quais referidos ao longo da Secção 2.4.1. Sendo que o utilizador era o responsável por descarregar, processar e armazenar os produtos computados. De forma a facilitar e standardizar o acesso aos dados, em 2018, foram financiadas cinco plataformas baseadas em tecnologias de *cloud*: **CREODIAS** [100], **ONDA** [101], **Mundi** [102], **WEkEO** [103] e **Sobloo** [104]. Estas são conhecidas como Data and Information Access Services (DIAS). E, para além de oferecerem um acesso centralizado, também disponibilizam ferramentas de processamento. Nesta fase da descrição é legítimo questionar: se o objetivo é que exista um único ponto de acesso a estes dados, por que motivo foram desenvolvidas cinco plataformas? A justificação desta decisão passa pelo estímulo à competitividade entre os fornecedores destas. Esta estratégia procura elevar a qualidade do serviço, beneficiando o utilizador final.

As cinco plataformas DIAS não se limitam a disponibilizar os dados do programa Copernicus. Providenciam também a habilidade de processar e combinar dados de outras fontes. Em relação aos dados disponibilizados, nestes estão incluídos os dados do Sentinel (ESA e EUMETSAT), dos produtos dos serviços Copernicus, de satélites comerciais, entre outros. Através deste acesso transversal, é possível que os utilizadores desenvolvam novas aplicações hospedadas na *cloud*. Eliminando a necessidade de descarregar grandes volumes de ficheiros de múltiplas fontes de forma a processá-los localmente.

Cada uma destas plataformas apresenta, dentro do âmbito geral, uma proposta diferente aos seus utilizadores. No entanto serão abordadas apenas as plataformas CREODIAS e ONDA, pois as suas funcionalidades são representativas das potencialidades destas cinco plataformas DIAS. A verdade é que as plataformas CREODIAS e ONDA encontram-se em níveis de maturidade superiores às restantes, não só em relação às funcionalidades apresentadas, mas também no que diz respeito aos produtos disponibilizados.

CREODIAS. Começando pelo CREODIAS, esta é uma infraestrutura *cloud* adaptada para processar e armazenar grandes volumes de dados de observação terrestre. A capacidade de processamento é acessível aos utilizadores através das ferramentas disponibilizadas. Focando a análise no repositório, este armazena atualmente 10,5 petabytes com crescimentos diários de 20 terabytes [105]. Contendo uma total cobertura espacial e temporal dos produtos do Sentinel-2, -3 e -5P, do Landsat (5 ao 8), do Envisat/Meris e dos serviços Copernicus. Em relação ao Sentinel-1, só alguns produtos estão acessíveis com cobertura total, que é o caso do Sentinel-1 GDR. No caso do Sentinel-1 SLC, este tem cobertura temporal total em território europeu, mas fora da Europa a cobertura é de apenas seis meses. Adicionalmente, são disponibilizados conjuntos de dados de outros fornecedores, tanto comerciais como entidades que computaram os seus produtos na plataforma CREODIAS.

As interfaces de acesso aos dados referidos são múltiplas[106]. O principal método de acesso aos dados do CREODIAS é a interface SWIFT/S3, pois estes estão armazenados no serviço de armazenamento de objetos Amazon S3, o que é fundamental para a escalabilidade, disponibilidade e desempenho desta solução. A interface referida permite que os utilizadores acedam a produtos ou apenas a parte deste, como cenas e metadados. Podendo-se até aceder a partes específicas de um ficheiro, o que é bastante útil em aplicações que necessitem de extrair *tiles* individuais de imagens em formato JPEG-2000.

Relativamente à pesquisa dos produtos, é disponibilizada uma API REST designada Data Finder API. As interrogações a esta, para além da sua expressividade espaciotemporal, permitem filtrar por coleções, tipo de terreno, percentagem de gelo/neve/nuvens/água ou caraterísticas do satélite. Adicionalmente, é possível fazer interrogações com linguagem natural (*e.g.* Lisbon 28 June 2016). Associada a esta API, existe uma interface gráfica (Fig. I.1) que ajuda o utilizador na geração do URL do *endpoint*. Ao comparar com a interface gráfica do Copernicus Open Access Hub, os princípios de desenho são os mesmos. No entanto, o CREODIAS disponibiliza outra página, designada EO Browser (Fig. I.2), focada na visualização e navegação dos dados disponíveis. No geral, esta permite filtrar pelas coleções principais, pela percentagem de nuvens e pelo intervalo temporal. Após obter os resultados, é possível visualizá-los em diferentes bandas e índices (Fig.I.3). Mas o mais distintivo é o facto de permitir fazer pesquisas INSPIRE e interrogações SPARQL. A primeira funcionalidade permite perceber que um dos metamodelos usados é o definido pela diretiva INSPIRE (Secção 2.3.2). Por outro lado, o facto de recorrer a ferramentas da *web* semântica permite uma maior expressividade nas interrogações, como por exemplo, pesquisando imagens do Sentinel-1 que mostrem aeroportos em Espanha.

As vantagens desta plataforma são diversas. Para além da diversidade nas interfaces de acesso, existe uma oferta vasta nas coleções disponíveis. Sendo que apenas não é feita a disponibilização dos dados, mas também de ferramentas de processamento que maximizem o valor dos dados disponíveis. O facto do processamento ser realizado na infraestrutura do CREODIAS evita que os utilizadores desperdicem grandes volumes de armazenamento em dados raramente acedidos. A gestão da infraestrutura é realizada pelos responsáveis por esta plataforma. Ainda assim, o utilizador tem um *dashboard* no qual consegue gerir as máquinas virtuais criadas, tal como nos típicos fornecedores *cloud*. Nestas máquinas é possível aceder de maneira eficiente ao armazenamento de objetos referido anteriormente. Em relação ao processamento, este passa por aceder a estas máquinas, não existindo uma API de computação como no caso do GEE (Secção 2.4.5). De realçar que após o processamento de novos produtos, estes podem ser armazenados nesta mesma infraestrutura, sendo isto faturado consoante o volume de dados. Por outro lado, não se identificou nenhum mecanismo que permita integrá-los de forma sistemática no catálogo.

Em relação a alguns dos dados que não estão disponíveis, a plataforma tem apontadores para repositórios remotos que facilitam este acesso. Outra caraterística funcional

importante é a homogeneidade dos metadados, da catalogação, das API e das funcionalidades entre qualquer coleção presente na plataforma. Finalmente, convém assinalar que a documentação é bastante boa.

ONDA. Passando à plataforma ONDA, esta permite que os seus utilizadores partilhem os seus dados e construam as suas aplicações de deteção remota na *cloud*. Com este fim em mente, é disponibilizado um repositório de dados de observação terrestre. Na verdade, esta plataforma ainda se encontra em desenvolvimento, mas já disponibiliza alguns dados e funcionalidades interessantes. No seu repositório estão, neste momento, disponíveis 15 petabytes e incluem todos os produtos dos Sentinel-1, -2 e -3, do Envisat ASAR, do Landsat 8 e dos serviços Copernicus do mar e do terreno. Planeia-se que no futuro sejam incluídos: os produtos do Sentinel-5P; os serviços Copernicus da atmosfera; os produtos do SUOMI NPP, Envisat MERIS, ENVISAT AATSR; os serviços Copernicus de emergência; e alguns conjuntos de dados de medições *in situ* [107].

Os serviços disponibilizados nesta plataforma subdividem-se em diferentes categorias: acesso aos dados; recursos na *cloud* com ferramentas embutidas de processamento, desenvolvimento, distribuição e administração; e serviços de gestão. Neste último estão incluídos serviços de armazenamento e disponibilização dos dados de fornecedores externos. Tal como no CREODIAS, é possível restringir o acesso dos dados a determinados utilizadores. Em relação ao acesso aos dados, os metadados seguem o perfil criado a partir do ISO 19156 referido na Secção 2.3.1. Sendo que existe uma lista pré-definida de metadados para cada uma das coleções, como por exemplo, dos produtos do Sentinel-1 [108]. Voltando ao acesso, este é feito através da interface gráfica ou das API disponibilizadas. A interface gráfica (Fig.I.4) é bastante semelhante ao Open Hub (Secção 2.4.1). Uma vez que a pesquisa não oferece grande expressividade, pois, para além da pesquisa textual, apenas é permitido filtrar pela missão e período de recolha do produto. Quando são apresentados os resultados, existem opções para focar o mapa na zona do produto, para descarregar o produto e para ver mais detalhes. Esta última permite observar os metadados do produto com informação relativa à plataforma, instrumento, sensor, processamento, órbita e geografia. Passando às API disponibilizadas, uma delas recorrendo a ODATA e a outra é uma interface Elastic Node Server (ENS). A API ODATA desenvolvida é bastante semelhante à do Open Hub. Quanto ao ENS, este estende os sistemas de armazenamento de objetos (*e.g.* S3), estabelecendo uma árvore lógica de nós. Estes nós podem ser localizados, interrogados e acedidos semanticamente através dos seus nomes, desconsiderando os seus formatos e localizações físicas. A verdade é que o arquivo contém milhões de produtos, portanto é importante organizá-los numa hierarquia que antecipe a utilização por parte dos utilizadores.

2.4.3 Earth Explorer

A U.S. Geological Survey (USGS) foi criada em 1879, partindo de uma iniciativa do congresso americano. A missão da USGS passa por recolher, monitorizar e analisar informação sobre os diversos recursos naturais. Dentro deste contexto, foi criado um centro científico de observação de recursos terrestres, designado EROS. O centro mantém um arquivo que tem a maior coleção de imagens da superfície terrestre atualmente. Nestas estão incluídos dados e imagens de satélite, fotografias áreas, *datasets* de cobertura terrestre e cartografia digital. Por outro lado, o EROS é a principal fonte de imagens e produtos da missão Landsat.

O arquivo do EROS, como referido, oferece uma vasta e interessante oferta de produtos. A disponibilização desta informação é feita através do catálogo Earth Explorer [109]. Neste estão presentes: fotografias áreas; imagens de radiómetros de alta resolução (AVHRR), e índices de vegetação associados (*e.g.* NVDI); testes de calibração/validação; imagens de satélites comerciais; coleções de fotografias militares, que deixaram de ser confidenciais; modelos digitais de elevação (DEM), que representam a superfície da Terra; mapas digitais dos EUA com informação espacial, estatística e temática representada num formato vetorial ou *raster*; o arquivo completo de todos os satélites Landsat; as coleções do Land Processing Distributied Active Archive Center (LPDAAC), incluindo o MODIS; imagens do Sentinel-2; entre muitos outros. De facto, a inclusão de todos estes produtos é a principal vantagem do Earth Explorer relativamente a outros catálogos.

As principais interfaces de acesso são a API REST e a gráfica. De referir que o acesso a certos produtos apenas está disponível através de uma API disponibilizada para encomendas [110]. A título de exemplo, para aceder a um produto de Landsat 8 nível 2 é necessário fazer um pedido nesta API, e só passado um certo tempo é que se receberá uma notificação a indicar que esse produto já está disponível para *download*. De assinalar que, sendo este catálogo bastante mais antigo do que os referidos até aqui, as interfaces disponibilizadas são naturalmente bastante mais arcaicas que as dos outros catálogos. Relativamente ao metamodelo deste catálogo, este recorre ao *standard* Content Standard for Digital Geospatial Metadata (CSDGM). Este foi criado pelo Federal Geographic Data Committee (FGDC), um comité do governo dos EUA que promove a coordenação no desenvolvimento, utilização e disseminação de dados geoespaciais, no contexto americano. O CSDGM define que nos metadados deve estar incluída obrigatoriamente informação de identificação e da origem dos metadados. Caso se aplique, também é incluída informação sobre a qualidade dos dados, a geografia, a geometria, tipos/entidades/atributos dos *datasets* e sobre o acesso ao mesmos [111].

2.4.4 Earth Data

O EOSDIS, referido na Secção 2.1.1, foi desenhado como um sistema distribuído. De facto, existem centros de arquivo em diferentes pontos dos EUA e cada um está encarregado de uma área especializada (*e.g.* terreno, atmosfera, oceano). Estes são designados

Distributed Active Archive Centers (DAACs) e asseguram que os dados das missões de observação terrestre estão acessíveis aos utilizadores. Mais concretamente, os DAACs são responsáveis por processar, arquivar, documentar e distribuir os dados, tanto dos satélites de observação terrestre como dos programas de medição *in situ*. Em relação à catalogação, a informação presente nos DAACs pode ser interrogada através do Earthdata Search [112], sendo este suportado pelo CMR, referido na Secção 2.3.3.

O Earthdata Search disponibiliza uma interface gráfica (Fig.I.5) na qual é possível filtrar por data, palavras-chave, plataforma, instrumento, área geográfica, organização, nível de processamento e classificação temporal (tempo real ou não). Adicionalmente, é possível fazer uma pesquisa em linguagem natural pelo tópico, coleção ou nome da localização. Após a pesquisa são apresentados os resultados (Fig.I.6), podendo ordenar-se pela relevância, utilização ou data. Nesta fase, existe a possibilidade de analisar os metadados da cena. Esta informação está disponibilizada em diferentes *standards*, nomeadamente ECHO 10 e ISO 19115, o que é possível devido ao UMM (Secção 2.3.3). Por outro lado, também são apresentados os metadados ao nível da coleção (Fig.I.7), incluindo URL relacionados, cobertura temporal e espacial, palavras-chave, *abstract*, informação sobre o responsável pelo processamento e armazenamento. Tal como ao nível da cena, os metadados das coleções são disponibilizados em diferentes *standards*.

CMR. Relativamente ao sistema que suporta o serviço de pesquisa, este trata-se do CMR da NASA [113, 114]. O repositório tem como função catalogar todos os metadados, dados e serviços do sistema EOSDIS. Antes da construção deste repositório, os fornecedores de metadados da NASA precisavam de ter em conta múltiplos sistemas, em que cada um deles tinha diferentes formatos e mecanismos de submissão e atualização dos metadados. Esta inconsistência reduzia o valor dos metadados, pois tornava mais complicada a pesquisa e o uso deste tipo de dados. Assim, o CMR baseou-se no trabalho feito pelo ECHO e GCMD, construindo um repositório de metadados unificado. De referir que o ECHO [115] é um sistema anterior ao CMR que também tinha objetivos de unificação dos metamodelos. Esta unificação beneficia os fornecedores e utilizadores, pois os primeiros apenas têm de ingerir os seus metadados no sistema, e na perspetiva dos utilizadores é possível a pesquisa de dados do ESDIS e IDN num só sistema.

Ao desenhar este sistema, a NASA permitiu que se melhorasse a qualidade, a consistência e a usabilidade dos metadados por parte dos utilizadores. Para isto, os metadados são geridos a nível concecional, ou seja, há uma categorização do tipo de metadados, existindo cenas, visualizações, variáveis, documentação, serviços, entre outros. Para além disto, os milhões de registos de metadados são agora disponibilizados através de pesquisas que seguem os *standards* de pesquisas espaciotemporais. Adicionalmente, incorpora mecanismos de avaliação dos metadados tanto automáticos como manuais, o que assegura uma maior qualidade. Finalmente, também suporta os múltiplos *standards* de metadados através do já referido UMM (Secção 2.3.3).

No que diz respeito à gestão dos metadados, no CMR existem três ferramentas principais: Metadata Management Tool (MMT), docBUILDER e um *dashboard* de enriquecimento dos metadados. Começando pelo MMT, este permite que os autores criem, atualizem, apaguem ou revejam os registos de metadados do CMR campo a campo. Em relação ao docBUILDER, os autores dos metadados podem atualizar a descrição de conjunto de dados, tendo esta de ser compatível com o perfil UMM-C (Secção 2.3.3). Por fim, o *dashboard*, é neste que os metadados ao nível da coleção e cena são corrigidos, completados e as inconsistências removidas. Inicialmente, existe uma geração automática destes metadados e uma verificação automática de qualidade. Após esta fase, os registos ficam disponíveis para verificação manual. Nesse momento, o operador classifica os problemas identificados em três classes (*i.e.* alto, médio, baixo), podendo também acrescentar uma recomendação de como os solucionar. Tendo esta classificação, são gerados relatórios automaticamente, nos quais podem ser assinalados o número de problemas por classe, juntamente com as recomendações para os resolver. Por outro lado, também é possível gerar relatórios de mais alto nível, sumarizando as métricas de todos os registos de um certo fornecedor.

2.4.5 Google Earth Engine

Grande parte dos catálogos até agora apresentados são referentes aos fornecedores "primários"deste tipo de dados. De forma a enriquecer este *survey*, apresenta-se agora um fornecedor *third party* - a Google através do GEE [116].

O GEE é uma plataforma de processamento, suportada por tecnologias *cloud*, especializada em dados geoespaciais, tirando partido da enorme capacidade computacional da Google. Esta foi desenhada não apenas tendo em conta os interesses dos especialistas de deteção remota, mas também os das audiências com insuficiência técnica para recorrer a recursos de computação de larga escala. Para este efeito, esconde-se os detalhes: de ingestão dos dados de processar uma enorme variedade de formatos; de gerir uma base de dados; de alocar máquinas; entre outras tarefas necessárias à computação. A análise desta plataforma será suportada por duas vertentes principais: catálogo e arquitetura. Visto que o catálogo inclui a cobertura, âmbito e metamodelo do mesmo. Por outro lado, na arquitetura são exploradas as caraterísticas funcionais.

Catálogo Neste está presente todo o arquivo do Landsat, Sentinel-1 e Sentinel-2. Incluindo também previsões climáticas, dados de cobertura do terreno e outros conjuntos de dados ambientais, geofísicos e também socioeconómicos, como apresentado na Fig. I.8. Apesar de a oferta ser muito ampla, é possível fazer *upload* de um produto, por forma a processá-lo. De referir que isto não se trata de uma automatização da ingestão, mas sim de um mecanismo de *upload ad hoc* para uma área privada.

Relativamente ao metamodelo, este é bastante genérico, tendo informação sobre a localização, data de aquisição e condições sobre as quais as imagens foram recolhidas

e processadas. Ilustrando a expressividade deste metamodelo, seria possível pesquisar imagens diurnas do Landsat 8 que intersetem qualquer parte do território português no dia do ano de 80 a 104, dos anos 2010 até 2012, com uma cobertura de nuvens inferior a 5%.

Outro ponto interessante deste catálogo é o facto dos utilizadores terem a possibilidade de partilhar os produtos processados e os seus *scripts*, e até recorrer a ferramentas de criação de aplicações *web* de exploração dos dados computados. De referir que o mecanismo de inserção de novos tipos de produtos não envolve a definição sistemática de um metamodelo, não existindo também a possibilidade de colaboração no enriquecimento do mesmo. Finalmente, após o desenvolvimento de um certo algoritmo, é possível produzir sistematicamente o produto resultante.

Arquitetura Em 2017, os responsáveis por idealizar esta plataforma publicaram um artigo no qual apresentam uma arquitetura simplificada da mesma [117]. Na Fig. 2.11 é representada essa arquitetura, sendo que esta é possível dividir-se em três camadas: apresentação, computação e armazenamento.

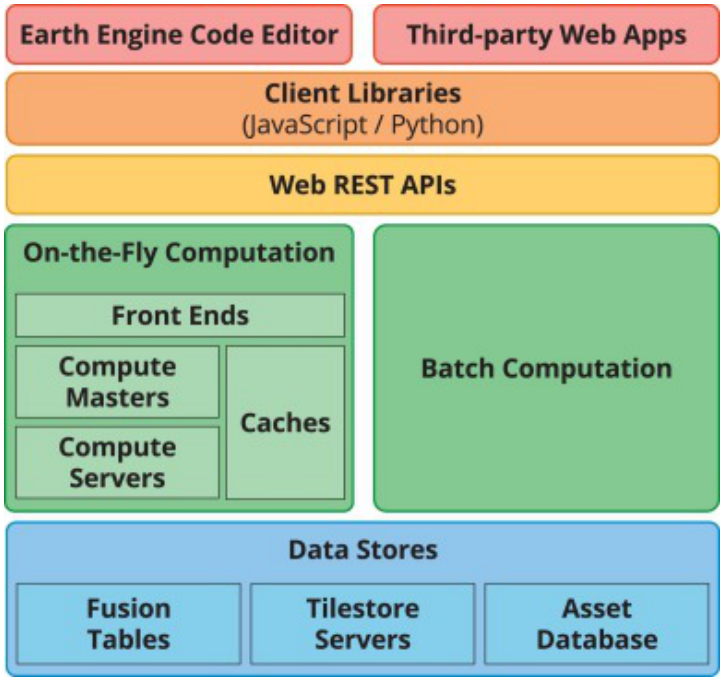


Figura 2.11: Arquitetura simplificada do GEE. [117]

Começando pela primeira camada referida, o editor de código do GEE e as aplicações *web* de entidades externas, que recorrem às bibliotecas tanto de Javascript como de Python, de forma a interrogar o sistema através da API REST. Focando no editor de código, este permite desenhar cadeias complexas de processamento geoespacial. Na Fig. 2.12 estão representadas um conjunto de caraterísticas que permitem tirar vantagem da API do GEE. Para além de editar o código, é também possível gerir os *scripts*, procurar conjuntos de dados, visualizar os produtos de deteção remota no mapa, fazer o *debug* da solução e

até partilhar os *scripts*. Relativamente às API Python e Javascript, estas incorporam um paradigma de programação influenciado pela necessidade de distribuição tanto do armazenamento como da computação. Uma vez que ao existir um paradigma poderão surgir alguns problemas de flexibilidade na implementação de novos algoritmos, isto é, as operações são pré-definidas pela API. Um exemplo concreto deste facto surgiu no contexto do desenvolvimento respetivo ao caso de estudo da Secção 5.3.4. Na implementação de um dos algoritmos, era necessário contar o número de píxeis de uma imagem Landsat (resolução de 30x30m) dentro de um píxel de uma imagem MODIS (resolução de 250x250m). Sem recorrer a API do GEE bastaria fazer uma interseção dos píxeis. Já no GEE, segundo o autor, era bastante mais complicado,sendo necessário seguir os seguintes passos: (i) criar um polígono para cada píxel da imagem MODIS; (ii) sobrepor esse polígono com a imagem Landsat; (iii) e, por fim, para cada polígono, executar a operação reduce, através da qual era possível obter a contagem de píxeis Landsat que existem dentro do mesmo.

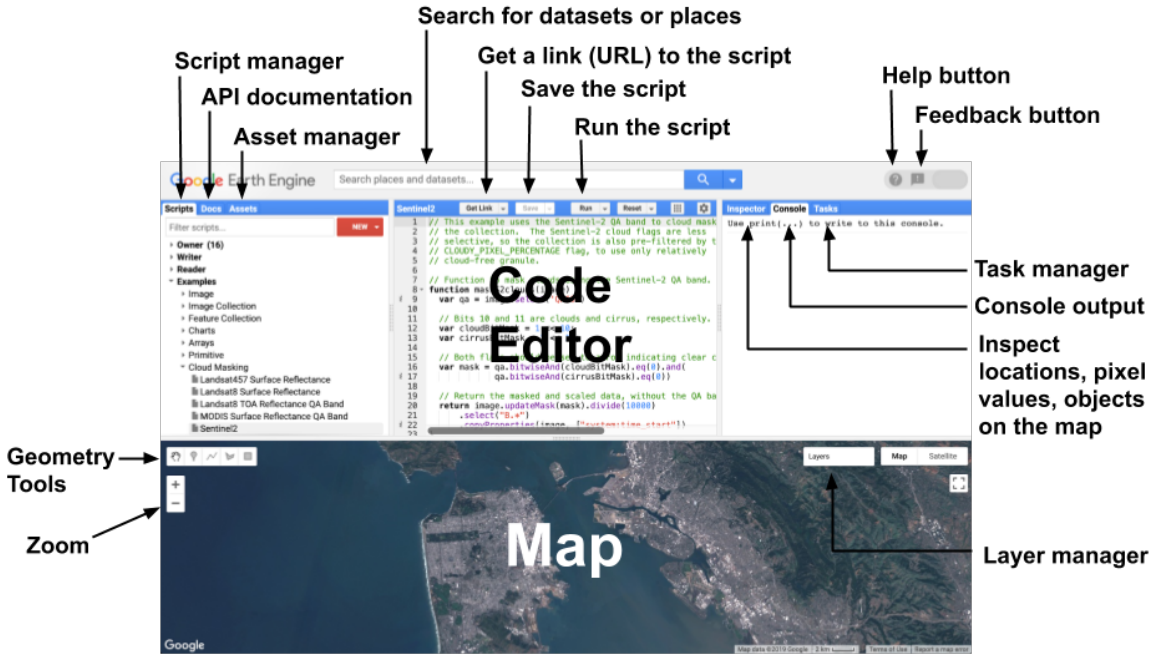


Figura 2.12: Diagrama de componentes do editor de código do GEE. [118]

Após a submissão de *queries* interativas, os servidores *front end* subdividem estas e enviam as unidades de trabalho resultantes para os *masters*, os quais gerem um conjunto de servidores de computação. As computações em *batch* são executadas de forma semelhante, mas recorrem a uma *framework* de processamento paralelo para gerir a distribuição - FlumeJava [119]. Geralmente, o ciclo de desenvolvimento começa pela exploração dos dados e implementação do algoritmo de uma forma interativa. Após esta fase, aplica-se o mesmo numa escala maior através do mecanismo de processamento em *batch*, o qual irá materializar o produto resultante no formato correspondente. Quando se usa a expressão "aplicar numa escala maior"neste contexto, pretende-se assinalar que, no contexto das *queries* interativas, a computação é apenas executada parcialmente. De forma a ilustrar,

caso se esteja a desenvolver no editor de código do GEE, pode ser suficiente computar apenas a área visualizada correspondente ao mapa presente no editor de código (Fig. 2.12). Isto permite que o utilizador não tenha preocupações em projetar o resultado para áreas específicas, já que esse passo é feito de forma automática.

Passando à camada de armazenamento, esta tem uma componente de metadados e outra de dados. Os primeiros estão armazenados na Asset Database, e contêm os metadados das imagens, permitindo pesquisas eficientes. Relativamente aos dados, é nos Tilestore Servers que está presente a informação em formato *raster*. Por outro lado, a informação vetorial, está presente nas Fusion Tables [120], uma base de dados que suporta tabelas de dados geométricos (pontos, linhas, polígonos).

2.5 Conclusão

Ao longo deste capítulo foi apresentada uma pesquisa relativa ao estado da arte e trabalho relacionado das principais vertentes envolvidas na construção de um repositório de dados e metadados de observação terrestre. Sendo o objetivo deste capítulo enquadrar a solução apresentada para o problema formulado, importa agora terminar com uma síntese enriquecida de uma visão crítica.

Relativamente aos metamodelos, os utilizados nos catálogos atuais são demasiado genéricos e por isso pouco expressivos para domínios aplicacionais específicos. É bastante difícil definir um metamodelo para múltiplos produtos sem que se perca a especificidade de cada um deles. Portanto, é importante existirem mecanismos de extensão do metamodelo, que partam de um conjunto de elementos de metainformação comum. Complementarmente, estes metamodelos não devem ser estáticos, pois num ambiente colaborativo existem sempre novos elementos de metainformação, resultantes da computação desses produtos.

Passando aos formatos e estrutura dos produtos, como foi demonstrado ao longo deste capítulo, estes são complexos e integram uma grande variedade de metainformação. Dos três produtos que foram apresentados na Secção 2.2.2, todos os formatos de representação dos dados e metadados são bastante distintos, mesmo quando o fornecedor é comum.

Em relação aos catálogos e plataformas, um dos principais problemas é realmente a expressividade nas pesquisas. O que seria suavizado com a presença de um mecanismo colaborativo de enriquecimento da metainformação. Por outro lado, algo que dificulta a pesquisa é o facto de muitas das vezes ter de se consultar vários destes catálogos. Existe portanto a necessidade de que este tipo de repositório tenha mecanismos simples de incorporação de novos tipos de produtos. Relativamente ao acesso aos produtos, por estes fornecedores servirem necessidades tão amplas, têm de limitar o acesso às suas API. Imaginando que todos os elementos de uma equipa necessitam do mesmo produto, cada um terá de fazer o *download* do mesmo. Na verdade, este problema não se põe nas plataformas, pois estas incorporam processamento na infraestrutura onde estão armazenados os dados.

Concretizando estes problemas nos exemplos estudados, importa dividir a análise entre catálogos e plataformas. Inseridos na categoria dos catálogos surgem soluções como: Copernicus Open Access Hub, Earth Explorer e Earthdata. Estes não incorporam plataformas de processamento, o que não permite tirar partido da localidade dos dados. Noutra perspetiva, todos estes são catálogos genéricos, não tendo produtos de um domínio específico. Por este motivo, os metamodelos dos produtos disponíveis também acabam por ser demasiado genéricos. De realçar que se se agregasse os produtos disponibilizados por estes três catálogos, a oferta já seria bastante significativa. Focando o Earthdata, este é suportado pelo CMR, o qual permite fazer uma gestão bastante interessante dos metadados. No entanto, a essência das ferramentas por este disponibilizadas é garantir a correção dos metadados, e não a extensibilidade dos mesmos.

Passando às plataformas, nesta categoria destacam-se claramente o CREODIAS e o GEE. As vantagens comuns a estes passa pelo conceito de levar a computação aos dados, excluindo a necessidade de *download* dos tipicamente volumosos produtos de observação terrestre. Estas plataformas normalmente integram dados de múltiplos fornecedores. No entanto, não existe nestes um mecanismo que permita que os múltiplos utilizadores colaborem no enriquecimento do catálogo, tanto ao nível dos produtos disponibilizados, como na melhoria da expressividade dos metamodelos. De facto, a gestão do metamodelo não é satisfatória. Existe um foco especial nos mecanismos de computação, e não nos de catalogação.

Destacando o CREODIAS, este apresenta muitas das vantagens e desvantagens já referidas no enquadramento das plataformas. Uma grande vantagem do uso do CREODIAS é a oferta disponibilizada no seu catálogo. Dado que existe uma normalização do metamodelo dos produtos provenientes de diferentes fornecedores. A desvantagem principal passa pelas tarifas associadas à computação e armazenamento. Em vez deste modelo, é importante definir uma solução que possa ser gerida pelas equipas de acordo com as suas necessidades, garantido desta forma a independência relativamente às múltiplas soluções existentes.

Movendo o foco para o GEE, este representa o estado da arte no que diz respeito a plataformas de processamento de dados de observação terrestre. A escalabilidade dos algoritmos é garantida através do paradigma de processamento distribuído. Ainda assim, as operações pré-definidas pela API do GEE dificultam bastante a implementação de algoritmos que noutro contexto seriam relativamente simples. No que diz respeito à catalogação, o problema é análogo ao CREODIAS, existe um maior foco na computação do que no armazenamento e distribuição dos produtos gerados.

Por fim, referir que, tendo em conta todas estas conclusões, no Capítulo 3 é apresentada a abordagem proposta para a resolução do problema formulado na Secção 1.2.

Capítulo 3

Abordagem

É neste capítulo que é exposta a abordagem proposta para a resolução do problema formulado na Secção 1.2.

Começando pela solução proposta, na Secção 3.1, apresenta-se uma descrição geral da mesma. De seguida, na Secção 3.2, é apresentada uma visão sobre todos os componentes que constituem este sistema, apresentando-se a arquitetura com as preocupações necessárias de automatização da ingestão, processamento e disponibilização dos produtos. Descrita a arquitetura, na Secção 3.3, são abordadas as fontes de dados, classificando-as em diferentes tipos (Secção 3.3.1). Adicionalmente, será retratado tanto o fluxo dos dados e metadados (Secção 3.3.2), como o mecanismo de especificação destas fontes (Secção 3.3.3). Ainda no contexto da solução, na Secção 3.4, define-se como é especificado o metamodelo, o que inclui abordar: o procedimento de especificação (Secção 3.4.1); a extensibilidade (Secção 3.4.2); e, finalmente, a linguagem de especificação ETL (*i.e. extract*, *transform*, *load*) - Secção 3.4.3. Tendo as funcionalidades bem definidas, na Secção 3.5, apresenta-se a API de interação dos utilizadores com o sistema.

Finalmente, para terminar o capítulo, é apresentada uma conclusão na Secção 3.6, sendo que esta não inclui apenas um sumário deste capítulo, mas integra também uma visão crítica.

3.1 Descrição Geral

A solução desenvolvida no contexto desta dissertação consiste num repositório de dados e metadados de observação terrestre, anexado com uma plataforma de desenvolvimento colaborativo de produtos de observação terrestre.

Tal como apresentado com mais detalhe na Secção 1.3, existem cinco vetores principais nesta solução:

- **Automatização da ingestão de dados e metadados**;
- **Especificação hierárquica do metamodelo**;
- **Linguagem de especificação de ETL**;
- **Mecanismo de interrogação de alto nível**;
- ***Framework* de processamento local**;

De forma a dar resposta aos aspetos referidos, foi desenhada uma arquitetura que se baseia em quatro módulos:

- **Computação *ad hoc***: trata-se de uma área na qual os utilizadores podem aceder aos dados presentes na infraestrutura, de forma a tirar partido da localidade dos mesmos;
- **Interface**: é disponibilizada uma interface gráfica e uma API REST, servindo estas de acesso e manipulação aos recursos disponibilizados no módulo de armazenamento;
- **Ingestão**: módulo responsável por descarregar e pré-processar os dados e metadados que serão armazenados no módulo de seguida apresentado;
- **Armazenamento**: módulo no qual está armazenada a informação extraída por parte da ingestão e da metainformação gerada a partir da interação entre o sistema e os seus utilizadores.

3.2 Arquitetura

Nesta secção, é apresentada em detalhe a arquitetura do sistema que suporta o repositório de dados e metadados de observação terrestre. Esta está dividida nos seguintes módulos, como esquematicamente representado na Fig. 3.1: computação *ad hoc* - área de computação que tire partido da localidade dos dados; interface - forma de acesso por parte dos utilizadores aos recursos disponibilizados no repositório; ingestão - componente que gere a obtenção dos dados e metadados para a plataforma de várias fontes; e, por fim, armazenamento - componente responsável pela gestão do armazenamento.

Este padrão de desenho tem múltiplas vantagens ao nível da manutenção do sistema. Nomeadamente, a possibilidade de atualizar as tecnologias de uma das componentes sem ter impacto nas outras. Por outro lado, permite um maior desacoplamento das equipas de desenvolvimento, isto é, uma mudança na macrocomponente de interface poderá não envolver qualquer esforço na de armazenamento. Adicionalmente, como o *deploy* é baseado em componentes, escalar isoladamente cada uma destas torna-se não só possível como simples.

Importa detalhar a função de cada uma das subcomponentes desta arquitetura, explicitando as relações entre si.

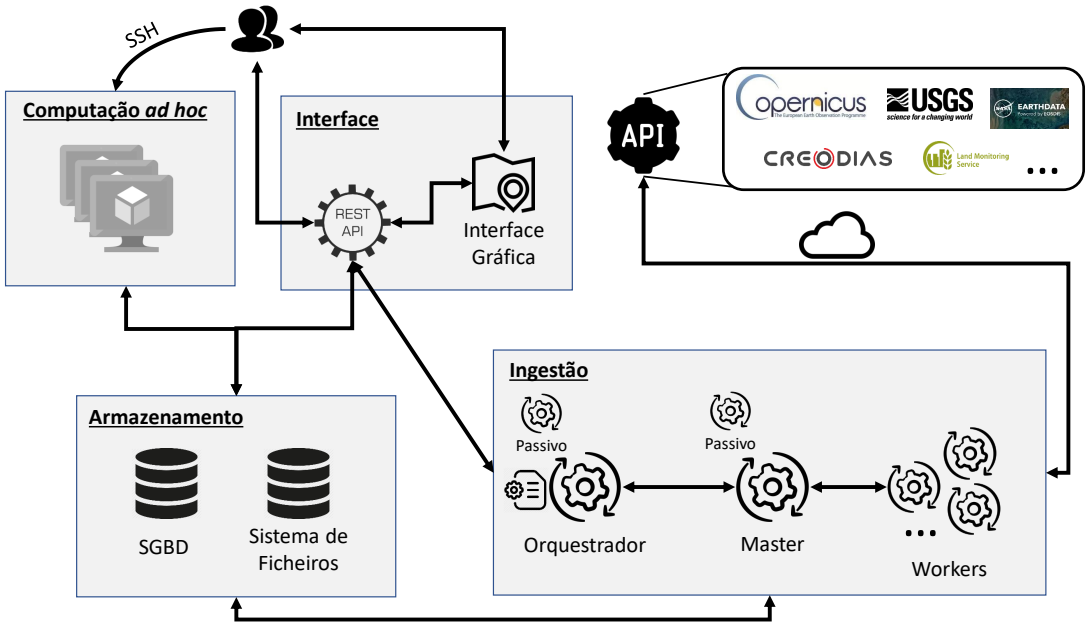


Figura 3.1: Arquitetura do sistema. SGBD - Sistema de Gestão de Base de Dados; SSH - *Secure Shell*.

Interface gráfica. Esta interface permite que os utilizadores deste repositório pesquisem o catálogo de produtos de observação terrestre, recorrendo também a ferramentas de visualização dos mesmos. Relativamente ao fluxo da componente, o utilizador acede à interface gráfica e esta interroga os serviços disponibilizados pela API REST, de forma a aceder aos recursos. É importante referir que a interface gráfica está fora do âmbito desta dissertação e, por isso, não serão providenciados mais detalhes relativamente à mesma.

REST API. Esta API disponibiliza serviços de acesso aos diferentes recursos presentes neste repositório. Esta API é pública, sendo que os utilizadores podem aceder a ela diretamente, ao invés de aceder à interface gráfica. Tratando-se de uma API de acesso aos recursos, esta está ligada ao módulo de armazenamento. Adicionalmente, esta componente interage com o módulo de ingestão, pois resultante da interação dos utilizadores com o sistema são geradas novas unidades de ingestão. Por exemplo, se o utilizador necessitar de uma imagem que não está presente na infraestrutura, é possível solicitar o *download* da mesma. No que diz respeito à especificação desta API, esta encontra-se descrita na Secção 3.5.

Computação *ad hoc*. Este módulo tem como função suportar computações *ad hoc* por parte dos utilizadores dentro da plataforma. Isto é, estando os dados presentes na infraestrutura (no módulo de armazenamento) é importante tirar partido da localidade dos dados durante a computação, evitando sobrecarregar o serviço ao descarregar produtos

volumosos. Para além disto, existem unidades isoladas (*e.g. containers* Docker) de computação para cada utilizador, de modo a realizar o processamento necessário, tendo em conta os recursos disponibilizados. Um exemplo das potencialidades deste modelo passa por instalar o QGIS [121], que se trata de um sistema de visualização, edição e análise de dados georreferenciados, em cada uma das unidades de computação, disponibilizando um acesso remoto à interface gráfica do mesmo. Desta forma, é possível levar a computação aos dados e não o contrário. Através desta área de computação são resolvidos dois problemas fundamentais: evita-se *download* de grandes volumes de dados; e tira-se partido de uma infraestrutura de computação dedicada, em vez de realizar todo o processamento na máquina pessoal do utilizador.

Orquestrador. Passando às componentes presentes no módulo de Ingestão, é importante começar pelo Orquestrador, que se trata da componente que gera as unidades de trabalho. Esta é responsável por aplicar a configuração das fontes de dados. Nesta está presente a especificação das fontes, nomeadamente, referindo a periodicidade das interrogações às mesmas - informação utilizada por parte do escalonador. Para mais detalhes relativamente a esta especificação, aconselha-se a leitura da Secção 3.3, na qual é analisada esta arquitetura de adaptadores para novas fontes. Por outro lado, nesta configuração também são definidos os passos de ETL, através de uma linguagem para esse efeito (Secção 3.4.3).

Relativamente à relação com outras componentes, resultante da interação dos utilizadores com o sistema, são criadas necessidades de ingestão no sistema. Por este motivo, o *engine* da API REST comunica ao Orquestrador essa necessidade, uma vez que todas as unidades de trabalho criadas no Orquestrador são submetidas ao Master, como descrito com maior detalhe na Secção 3.2.1.

Um elemento que ainda não foi discutido é a existência de um Orquestrador passivo. O *deploy* deste deve ser feito numa máquina distinta à do ativo. Este permite que a recuperação do Orquestrador, em caso de falha, seja mais rápida. A forma como este recupera passa por analisar os *logs* registados no SGBD, identificando interrogações periódicas que não foram executadas devido à falha.

Master. Focando agora a componente Master, é nesta que é realizado o supervisionamento dos Workers disponíveis e mantido o registo dos itens de trabalho, tanto dos pendentes como dos já realizados e os que estão a ser executados de momento. Este recebe os itens de trabalho do Orquestrador e distribui os mesmos pelas unidades trabalhadoras, designadas Workers.

Homologamente ao Orquestrador, existe também um Master passivo, que serve o mesmo propósito. Sendo que a descrição do mecanismo de recuperação é detalhada na Secção 3.2.1.

Worker. Movendo a atenção para a componente Worker, esta é resposnável pela execução das unidades de trabalho. Ou seja, comunica com as API externas, nas quais obtém, fundamentalmente, os dados e metadados de observação terrestre. Estes dados serão armazenados no sistema de ficheiros, enquanto que os metadados serão indexados no SGBD. De notar que os metadados passam ainda por uma fase de pré-processamento, na qual são definidas extrações específicas dos metadados que foram obtidos; de seguida podem ser aplicadas transformações aos dados resultantes dessa extração; e, por fim, essa unidade de metainformação é mapeada para o metamodelo.

Numa visão mais infraestrutural, é possível escalar horizontalmente o número de Workers. Isto representa uma enorme vantagem em termos de escalabilidade, pois o processamento mais pesado é justamente nesta componente.

SGBD. Como referido no parágrafo anterior, é nesta componente que estarão indexados os metadados, sendo acessíveis por parte dos utilizadores via API REST ou interface gráfica. É com base nestes metadados que é construído o catálogo de produtos de observação terrestre. Para este efeito, este SGBD deve obedecer a um conjunto de propriedades de forma a que se atinja uma expressividade, *performance* e manutenção satisfatória na gestão destes metadados.

Sistema de Ficheiros. Por fim, nesta componente estão presentes os dados propriamente ditos, como, por exemplo, as imagens de satélite nas diversas bandas. É este tipo de informação que em última análise os utilizadores pretendem processar. Numa fase de pesquisa, os metadados são importantes, mas o processamento em si é sobre os dados presentes nesta componente. Os detalhes da gestão do fluxo dos dados está incluído na Secção 3.3.2.

3.2.1 Protocolo de Distribuição da Ingestão

Dado que a tarefa de ingestão deve ser realizada de uma forma distribuída, aqui apresenta-se o protocolo entre as principais entidades responsáveis por esta ingestão - o Orquestrador, o Master e o Worker, tendo cada uma destas sido descrita no início da Secção 3.2. São estas componentes que, ao comunicar entre si, garantem a completude de cada uma das unidades de trabalho de ingestão. É importante referir que a comunicação entre estas entidades é baseada num objeto designado trabalho. Neste está encapsulada toda a interação com as fontes de dados. Na verdade, existe toda uma hierarquia definida a partir desse objeto raiz, estando essa descrita na Secção 3.3. De notar que este protocolo é baseado em trabalho prévio [122], no qual é descrita a especificação do mesmo recorrendo à biblioteca Akka [123].

A descrição deste protocolo subdividir-se-á entre a interação Orquestrador/Master e Master/Worker. Esta divisão faz sentido dado que não existe qualquer tipo de interação direta entre o Orquestrador e o Worker.

Interação Orquestrador/Master. Começando pela comunicação entre Orquestrador e Master, é no Orquestrador que está presente o escalonador que gera as unidades de trabalho periódicas, segundo a configuração. Adicionalmente, é também neste que são submetidas as unidades de trabalho externas ao sistema - resultantes da interação dos utilizadores com o sistema.

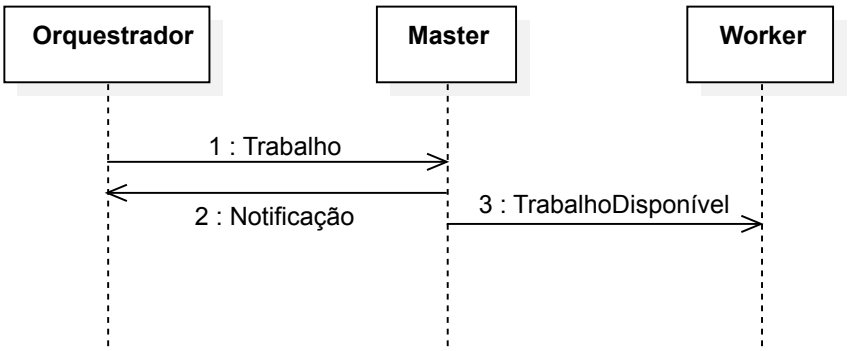


Figura 3.2: Diagrama de sequência simplificado, que representa o caso de sucesso de comunicação entre Orquestrador e Master.

No diagrama apresentado na Fig. 3.2 está representada a sequência de mensagens trocadas entre estas duas entidades. Relativamente ao Orquestrador, este envia uma mensagem com a unidade de trabalho ao Master. Posteriormente, caso a mensagem seja recebida, será enviada uma mensagem de *acknowledgment* por parte do Master. Importa notar que o Orquestrador tem um *timeout* associado ao envio de cada uma das unidades de trabalho e, caso não receba a mensagem de *acknowledgment*, irá tentar de novo mais tarde. Quando finalmente a mensagem de *acknowledgment* for recebida, o Orquestrador tem garantida a durabilidade desta submissão, ou seja, está assegurado o registo daquela unidade de trabalho.

Mudando de perspetiva para a do Master, este ao receber a mensagem na qual está incluída a unidade de trabalho, verifica se aquela já foi recebida, garantindo assim a idempotência. De forma a identificar univocamente uma unidade de trabalho, esta apresenta um identificador único que consiste numa *string* aleatória. Por outro lado, caso não tenha sido recebida, essa unidade de trabalho será persistida. De seguida, o Master contacta todos os Workers de modo a informar que existe uma nova unidade de trabalho pronta a ser executada.

Interação Master/Worker. Passando à comunicação entre Master e Worker, o primeiro tem as duas principais missões: supervisionar os Workers disponíveis; e manter registo tanto dos itens de trabalho que foram agendados como dos que estão presentemente em execução. Para este efeito, mantém-se os Workers disponíveis e persiste-se as unidades de trabalho pendentes, as que se encontram em execução, as já executadas, as que falharam e todas as que foram submetidas por parte do Orquestrador.

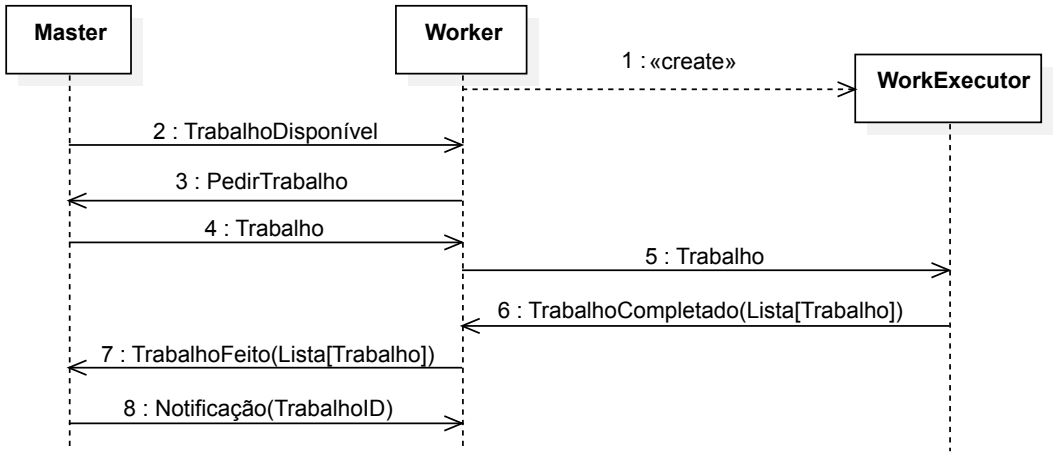


Figura 3.3: Diagrama de sequência simplificado, que representa o caso de sucesso de comunicação entre Master e Worker de forma a executar uma unidade de trabalho.

Seguindo a sequência de mensagens trocadas no diagrama apresentado na Fig. 3.3, quando o Master recebe uma nova unidade de trabalho, envia uma mensagem para cada um dos Workers, sinalizando este facto. Depois, todos os Workers que não estejam ocupados com outras unidades de trabalho, responderão ao Master que estão disponíveis para serem alocados para esse trabalho. Recorrendo a este mecanismo de *pull* das unidades de trabalho é possível controlar a *back pressure*, o que leva a que o sistema responda bem a fases de grande carga, ao invés de colapsar. Para assegurar esta propriedade, o Master não envia unidades de trabalho para Workers que já estejam ocupados, nem enche a fila de mensagens dos Workers, o que traria sobrecarga inusitadamente - tanto ao nível de memória como de processamento.

Voltando à perspetiva do Master, para cada uma destas mensagens será verificado se ainda existe trabalho, pois existem três cenários possíveis: o Worker foi o primeiro a responder à solicitação; o Worker não foi o primeiro e portanto essa unidade de trabalho já foi alocada; e, por fim, é possível, apesar de o Worker não ser o primeiro a responder, já ter surgido outra unidade de trabalho à qual este pode ser alocado. Existindo unidades de trabalho pendentes, importa verificar se aquele Worker entretanto não foi alocado a outra unidade de trabalho, o que poderá ter acontecido entre a mensagem de pedir trabalho e a do Master a processar essa mesma mensagem. Após todas estas validações, a unidade de trabalho é finalmente submetida ao Worker, tendo um *timeout* associado.

Ao receber essa unidade de trabalho, o Worker delega o processamento para uma entidade-filha - WorkExecutor. Isto permitirá que o Worker mantenha ativo o mecanismo de *heartbeat* para o Master, enquanto o WorkExecutor executa a unidade de trabalho. Assim, o Worker supervisiona o WorkExecutor, conseguindo identificar quando uma unidade de trabalho falha. Por outro lado, caso esta seja concluída com sucesso, esta retorna uma lista de unidades de trabalho identificadas durante a execução. De forma a ilustrar o que foi explicitado, numa unidade de trabalho que consista em interrogar uma API REST

é possível que esta apresente o seu resultado por páginas; portanto, é necessário gerar novas unidades de trabalho para a página seguinte, garantindo uma granularidade mais fina das mesmas.

Posteriormente, o Worker reencaminha esta mensagem para o Master, que verifica a idempotência da operação, validando se já recebeu a indicação de que aquela unidade de trabalho já foi executada. Este mecanismo é necessário, pois quando o Master demora a enviar a mensagem de *acknowledgment*, o Worker reenvia a mensagem de que o trabalho já foi executado. Por fim, o Master extrai a lista de unidades de trabalho embebida na mensagem, notificando os Workers da sua existência.

Até aqui retratou-se o caso de sucesso, ignorando a possibilidade de falha do Worker, do Master ou da unidade de trabalho, ou até do atraso da entrega de mensagens provocado pela rede. De forma a resolver este tipo de problema, o protocolo desenvolvido apresenta quatro mecanismos:

- ***Heartbeat***: o Worker envia periodicamente uma mensagem ao Master, sendo que este último guarda o registo temporal no qual foi enviada a última mensagem;

- **Verificação periódica de *timeouts***: cada unidade de trabalho tem um *timeout* associado. É nesta verificação que para cada Worker, caso esteja ocupado, valida-se se o *timeout* já foi ultrapassado. Na eventualidade da validação falhar, esta unidade de trabalho passa a pendente e são enviadas notificações deste facto. Por outro lado, caso o Worker não esteja ocupado, verifica-se se o intervalo entre o último *heartbeat* e a data atual é maior que o configurado para considerar um Worker como indisponível. Se de facto for maior, elimina-se esse Worker do conjunto dos disponíveis;

- **Recuperação do Master**: caso o Master falhe, quando arranca de novo, passa por uma fase de recuperação. Nesta carregará o estado das unidades de trabalho persistidas, sendo que a recuperação só acontecerá verdadeiramente durante o segundo ponto desta enumeração. Isto porque todos as unidades de trabalho que estavam registadas como estando em execução, mas que já excederam o *timeout*, serão registadas como pendentes, notificando os atuais Workers da nova unidade de trabalho;

- ***Backoff* e *timeout* dinâmico**: a cada unidade de trabalho está associado o número de tentativas da mesma, o *timeout* e um intervalo de *backoff* com incrementos exponenciais. Quando uma unidade de trabalho falha ou excede o *timeout*, poderá ser devido: à indisponibilidade de algum serviço; a uma falha de comunicação entre Master e Worker; ou até ao reduzido tempo de *timeout* para completar a tarefa. Por este motivo, a unidade de trabalho é submetida de novo apenas após um certo intervalo. Esse intervalo deve crescer exponencialmente de forma a garantir que os incrementos sejam suficientes para, por exemplo, o serviço necessário voltar a estar disponível. Sendo que não se pode seguir esta estratégia *ad aeternum*, e por isso, ao fim de um número de tentativas máximo, essa unidade de trabalho deve

ser descartada. Por outro lado, a cada tentativa é aumentado o intervalo de *timeout* associado à execução.

Finalmente, um dos pontos que ainda não foi abordado ao longo da descrição do protocolo foi o facto de existirem API (*e.g.* Copernicus Hub, CREODIAS, Earth Explorer) com limitações de concorrência. Isto é, existe uma limitação do tráfego, conseguindo apenas executar-se um número limitado de pedidos em paralelo. Sendo este protocolo distribuído, é necessário regular este problema de forma a que os pedidos não estejam constantemente a falhar. A solução encontrada passou por introduzir uma estrutura no Master na qual, para cada API, se mantém o registo do número de unidades de trabalho que estão a ser executadas em paralelo. Na fase de verificar a existência de unidades de trabalhos pendentes, este facto é tomado em consideração.

3.3 Fontes de Dados

Uma das grandes dificuldades na resolução do problema que esta dissertação se propõe a resolver passa pela heterogeneidade das fontes de dados. Esta surge tanto ao nível das interfaces que disponibilizam como da quantidade de fornecedores com os seus próprios catálogos e metamodelos.

Ainda no contexto das fontes de dados, importa definir o fluxo de dados entre as mesmas e o sistema, detalhando também o fluxo dos dados dentro dos diferentes subsistemas.

Após perceber o fluxo pode-se então explanar o modo como são especificadas estas fontes. Complementando com os passos necessários para criar adaptadores para novos tipos de fonte.

3.3.1 Heterogeneidade das API

Existindo esta heterogeneidade de fontes de dados de observação terrestre, importa refletir sistematicamente relativamente a esta propriedade. Para este efeito, são definidas classes de fontes, existindo mecanismos para estender a arquitetura com novos tipos de fontes. As API existentes estão incluídas genericamente nos seguintes tipos:

- **Catalogação**: este tipo de API tem como suporte os metadados dos diferentes produtos tendo o objetivo de disponibilizar uma interface de pesquisa, de forma a identificar os produtos que se procuram. Juntamente à listagem, é apresentado um resumo da metainformação para cada um dos produtos, de modo a descrevê-los sinteticamente. Relativamente ao acesso aos dados (*e.g.* imagens nas diferentes bandas), este já é realizado ao nível das classes de API descritas a seguir - Acesso e Encomenda.

- **Acesso**: classe de API que permite o acesso aos produtos propriamente ditos, sendo que muitas das vezes a estrutura interna dos produtos inclui metainformação que

não é catalogada. Geralmente, este tipo de API, por tratar do acesso a objetos de maior volume, é limitada. Esta limitação é fundamentalmente imposta através de dois mecanismos - gestão da velocidade de *download* e/ou limitação de pedidos concorrentes. Por outro lado, para aceder via esta API, existe a restrição de que os produtos devem estar disponíveis *online*, caso tal não se suceda é necessário recorrer à classe de API referida no próximo ponto;

- **Encomenda**: esta macroclasse tem dois fins principais. Por um lado, dado o enorme volume dos produtos de observação terrestre, existe a necessidade de arquivar (em *cold storage*) os que se prevêem como sendo os menos acedidos. É portanto através deste tipo de API que se requer o acesso a estes produtos. Por outro lado, alguns fornecedores disponibilizam nas suas API a possibilidade de encomendar produtos processados a pedido, como, por exemplo, solicitando a computação da imagem NDVI (índice de vegetação) de uma determinada cena. Em ambas as funcionalidades existe a noção de encomenda, ao invés de se realizar o pedido e obter a informação de forma síncrona.

De forma a demonstrar o mapeamento da formulação anterior com a realidade, seguem-se duas instanciações para cada uma das classes referidas. Começando pelo Copernicus, como referido na Secção 2.4.1, este apresenta uma API de catalogação - Open Search, e outra de acesso aos produtos em si - OData. Adicionalmente, fornece uma API designada *Long Term Archive* (LTA), que permite o acesso a produtos que estão *offline*.

Passando ao caso do Earth Explorer, referido na Secção 2.4.3, a API de catalogação e acesso é homóloga à do Copernicus. Já em relação à de encomendas, esta é um pouco diferente. Isto porque esta, para além do acesso a produtos *offline*, também cobre a componente de processamento a pedido - estando o tipo de processamento pré-definido na API.

Tendo classificado as API na vertente da sua finalidade, é importante analisar agora os mecanismos de acesso às mesmas. Tipicamente, tanto a pesquisa como o acesso aos produtos é realizado via HTTP, sendo que os metadados são disponibilizados numa API REST com uma forte componente espaciotemporal. Porém, existem outras formas de acesso aos dados e metadados de observação terrestre. Nomeadamente, no caso de produtos de análise da superfície terrestre [124] disponibilizados pelo EUMETSAT, o acesso é realizado via FTP, ou seja, recorrendo a uma técnica de *push*, tendo este tipo de produtos resolução temporal bastante alta (*e.g.* 15 minutos em produtos de temperatura da superfície terrestre [125]), este mecanismo permite uma melhor gestão do tráfego. Relativamente ao acesso em FTP, é necessário indicar ao fornecedor o IP, a porta pelos quais serão injetados os produtos, a pasta na qual se pretende armazenar e os dados de autenticação. Por fim, existem outros tipos de acesso que não são tão comuns, mas também interessantes apontar. Mais especificamente, acesso a metainformação presente num SGBD externo. Ou, por outro lado, acesso aos produtos através de um volume partilhado.

3.3.2 Fluxo dos Dados e Metadados

Definidos os tipos de fontes de dados e mecanismos de acesso existentes, interessa apresentar o fluxo de ingestão dos dados e metadados. É importante nesta fase descrever este aspeto, pois, na Secção 3.3.3, é descrito o mecanismo de especificação das fontes interrogadas durante a fase de ingestão. Este fluxo apresenta genericamente três macrofases:

Verificação Periódica → ETL → *Download* dos Produtos

Verificação Periódica. A primeira fase da ingestão passa por periodicamente interrogar as API de catalogação, de modo a verificar se existem produtos novos, dado que a componente responsável por agendar estas unidades de trabalho é o Orquestrador, como referido na Secção 3.2.

Relativamente à periodicidade destas interrogações, em alguns casos a frequência de novos produtos é realizada de acordo com as órbitas dos satélites que captaram os dados. Portanto, o escalonador tem duas principais formas de operação, dependendo do tipo de produto.

Uma delas é configurar uma frequência constante, na qual se verifica a existência de novos produtos de x em x tempo. Sendo que esta abordagem poderá resultar num problema de *polling*, ou seja, interrogar várias vezes a API sem que existam novos produtos. O que tanto levará ao desperdício de recursos como levantará a possibilidade de limitação de tráfego por parte dos fornecedores, na eventualidade de identificarem a situação. De referir que esta solução é válida para produtos cuja resolução temporal seja reduzida (*e.g.* um dia), e caso não seja disponibilizado o ficheiro vetorial que descreve as órbitas do satélite responsável pela obtenção de dados de um determinado produto.

Existindo a possibilidade de aceder ao ficheiro de descrição das órbitas, a estratégia poderá ser mais inteligente. Neste caso, apenas se verifica a existência de novos produtos de acordo com a órbita, visto que no caso do Sentinel-2 estes planos de aquisição são atualizados por períodos de um ano [126]. Recorrendo a esta estratégia é possível evitar o *polling* e identificar a ausência de um produto que deveria ter sido distribuído. Um problema que ainda não foi abordado é o facto de existir um intervalo entre captar os dados e estes serem ingeridos no catálogo. Isto é, os satélites geram as imagens, de seguida emitem-nas para um segmento onde estas são pré-processadas e, por fim, são ingeridas como produto no repositório. Naturalmente, existe um intervalo entre o primeiro passo e o último referido. Para este efeito, é necessária uma folga, devendo esta ser configurada para cada fonte de dados e tipo de produto. Se mesmo com essa folga o produto ainda não está disponível, muda-se para uma estratégia de *polling*. Se passado um tempo configurável ainda não foi disponibilizado, declara-se esse produto como ausente.

Até esta fase da descrição apenas se teve em conta os produtos lançados a partir do momento em que o sistema começa a executar. Uma das tarefas que é executada quando o sistema começa o seu ciclo de vida, é a recuperação do histórico de produtos. Para este efeito, é possível definir um *epoch*, que se trata de uma estampilha, a partir do qual será

feita esta recuperação, ignorando o que é anterior ao mesmo. Esta recuperação passa por interrogar a API de catalogação com uma componente temporal que começa no valor de *epoch* e acaba no momento em que é feita a interrogação. Sendo que o resultado será muito provavelmente apresentado por páginas, basta então iterar as mesmas e os produtos presentes em cada uma.

Finalmente, referir que nas fontes de dados por *push* - abordadas na Secção 3.3.1 - o procedimento não é igual ao descrito. Neste caso, logo que o produto se encontre disponível, este é transferido pelo fornecedor para o sistema de ficheiros da plataforma, existindo um processo que monitoriza a existência de novos ficheiros na pasta indicada. Sempre que forem detetados novos produtos nessa pasta, passa-se à fase de pré-processamento dos mesmos, que é realizada de forma semelhante à dos métodos de *pull*, sendo a única diferença o facto de toda a informação necessária já estar presente na infraestrutura local.

ETL. Identificada a existência de um novo produto, passa-se a uma fase de ETL dos metadados respeitantes ao mesmo. Primeiro que tudo importa refletir sobre o facto de se ter recorrido a uma abordagem ETL e não ELT (*extract*, *load*, *transform*). A primeira abordagem referida surgiu no contexto da gestão de Datawarehouses, por outro lado, a segunda emergiu, recentemente, com as tecnologias de *big data*.

Relativamente às diferenças fundamentais, no ETL as transformações são executadas numa área de *staging* e só depois são carregadas no sistema alvo. O que poderá ser preocupante dependendo das caraterísticas de volume e velocidade dos dados (metadados neste caso). Esta é uma abordagem que à partida é mais simples de implementar, mas mais difícil manter, pois apenas são carregados os metadados declarados, na fase de desenho, como sendo importantes. No entanto, o impacto desta dificuldade na manutenção é reduzido por um dos mecanismos propostos nesta dissertação - linguagem de especificação ETL (Secção 3.4.3) - que permite, em fase de produção, atualizar declarativamente as extrações.

Relativamente ao ELT, este processo executa as transformações no sistema alvo, ou seja, não passa primeiro por uma área de *staging*. A principal vantagem desta abordagem passa por acomodar a ingestão de grandes volumes de dados a uma maior velocidade, dado que se elimina o tempo de pré-processamento antes do *load* no sistema alvo. Uma vez que todas as transformações necessárias são executadas em *batch* no sistema alvo, e não na área de *staging*. No contexto desta dissertação, os metadados não são suficientemente volumosos para que se justifique recorrer a este tipo processo, dado que uma abordagem ELT iria acrescentar complexidade à solução. Isto porque as tecnologias subjacentes a este tipo de arquitetura são mais avançadas (*e.g.* Hadoop [127]).

Focando então no processo ETL, são descarregados um conjunto de ficheiros que descrevem o produto, nos quais podem estar incluídos, por exemplo, indicadores de qualidade como máscaras de nuvens (entre muitos outros elementos de metainformação). Posteriormente, estes ficheiros serão pré-processados, o que inclui extrair elementos de

metainformação, transformá-los e, por fim, mapeá-los no metamodelo. Sendo que o mapeamento no metamodelo passa por registar no SGBD o valor do campo correspondente. De notar que todo este processo está descrito detalhadamente na Secção 3.4.3.

***Download* dos Produtos.** Com os metadados armazenados no SGBD, existe agora a possibilidade de catalogar esse produto. Mesmo que os dados não estejam armazenados localmente, o catálogo poderá servir de *metasearcher* - fornecendo um apontador externo, através do qual esse produto poderá ser acedido na íntegra. Tratando-se este repositório não só de metadados mas também de dados, são necessários mecanismos para definir quais os produtos presentes na infraestrutura, e quais estão acessíveis remotamente. Na verdade, armazenar todos os produtos seria demasiado dispendioso em termos de recursos, tanto ao nível da ingestão, como de armazenamento e distribuição. Portanto, esta solução tem em conta que nem todos os fornecedores têm a capacidade infraestrutural de entidades como a NASA, a Google ou a Amazon.

Relativamente à forma como estes dados são ingeridos no sistema de ficheiros da plataforma, existem fundamentalmente os seguintes mecanismos:

- ***Prefetch*:** este trata-se do único mecanismo no qual os utilizadores podem requerer via API o *download* de dados para o armazenamento na infraestrutura. O nome deve-se ao facto dos utilizadores realizarem o pedido de *download* do produto, de forma a processá-lo - tirando partido da localidade dos dados. Com a presença destes produtos na infraestrutura, qualquer utilizador poderá aceder aos mesmos e não só o que solicitou o seu *download*;

- **Configuração *a priori*:** aliado ao mecanismo de *prefetching*, existe a possibilidade de descarregar automaticamente produtos que tendem a ser muito consultados. Neste caso, não se espera que seja o utilizador a realizar o pedido para descarregar determinado produto. Na configuração estarão descritos os dados que se pretende ingerir de forma automática. Desta forma, logo que for sinalizada a existência de um novo produto, verifica-se se é suposto fazer *download* dos dados que o constituem. De modo a exemplificar a granularidade deste *endpoint*, poder-se-ia declarar que se pretende descarregar *a priori* produtos do Sentinel-2, com uma percentagem de nuvens inferior a 5%, cuja cena intersete o território português, e que, em termos temporais, seja limitada a meses de verão;

- ***Push*:** como referido na Secção 3.3.1, um dos tipos de acesso a produtos de observação terrestre é através de um procedimento de *push*, por parte do fornecedor, para infraestruturas locais. Neste caso, não há possibilidade de apenas ingerir os metadados. Portanto, quando o produto é transferido, os dados seguem juntamente aos metadados.

Uma técnica que ainda não foi referida, que é transversal a todos estes procedimentos de ingestão de dados, é a redundância das fontes. Isto é, o produto não é disponibilizado

apenas pelo fornecedor original do mesmo. Existem outros que nos seus catálogos também permitem o acesso a esses mesmos produtos. Nomeadamente, entidades externas como a Amazon (com os seus conjuntos de dados abertos - *e.g.* produtos Landsat 8) e a Google (através do catálogo do GEE - Secção 2.4.5). Ao longo da Secção 2.4, foram referidas as limitações de concorrência e tráfego dos diferentes catálogos. Através desta propriedade, é possível melhorar a *performance* da ingestão, pois caso exista uma fonte sobrecarregada, é possível aceder a outra que não o esteja. De referir que a gestão dos limites de concorrência destas API é feita ao nível da componente Master do subsistema de ingestão.

3.3.3 Procedimento de Especificação das Fontes de Dados

Relativamente à especificação das fontes, esta é construída em cima de adaptadores pré-existentes. Estes estão representados na Fig. 3.4, numa relação hierárquica. Sendo que os adaptadores fora do sistema hierárquico seguem um padrão de composição. Ou seja, estes podem ser incluídos a qualquer nível da hierarquia. De notar que a instanciação de um adaptador trata-se da especificação de uma fonte, uma vez que este produz as unidades de trabalho referidas na Secção 3.2.1.

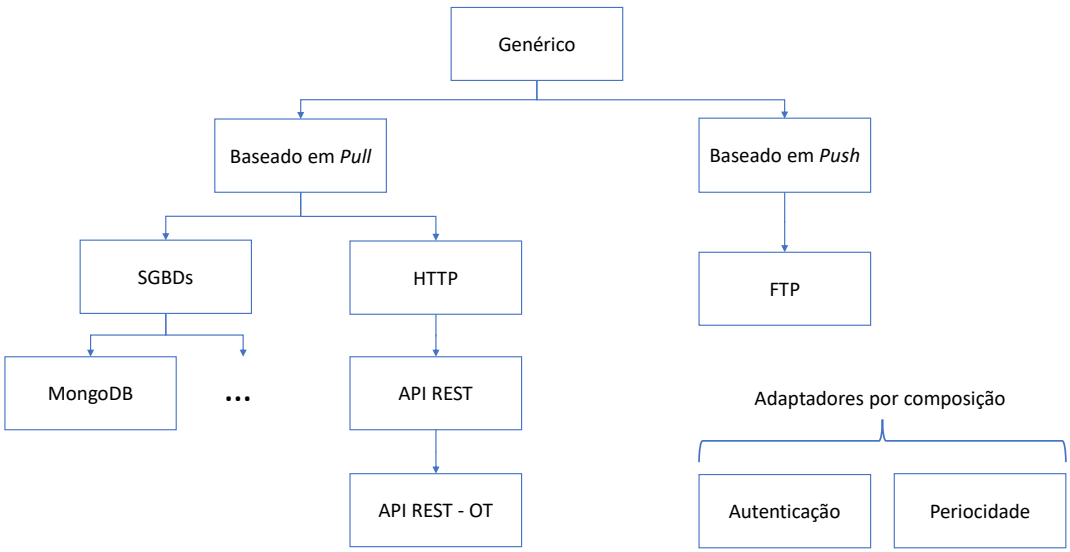


Figura 3.4: Diagrama de hierarquia entre adaptadores para fontes de dados, com representação adicional dos adaptadores por composição. OT - Obervação Terrestre.

A fase de especificação de uma nova fonte começa por analisar a hierarquia e selecionar o esqueleto sobre o qual será instanciada a nova fonte, podendo esta extensão ser feita a qualquer nível da hierarquia apresentada, e não apenas nas folhas. Após esta decisão, é possível compor o esqueleto com outros adaptadores, que estão fora dessa estrutura hierárquica. Tendo definido o *template*, implementa-se as especificidades da fonte que se pretende adicionar.

Importa agora analisar cada nível de adaptador. Começando pelo `Genérico`, este tem uma descrição, um identificador, os atributos de *backoff* (definidos na Secção 3.2.1) e a função que é executada por parte dos Workers.

Passando ao nível dos baseados em *pull* e *push*, esta divisão é meramente conceptual, não sendo acrescentada nenhuma caraterística especial. Já no nível abaixo, os SGBD terão armazenada a informação comum ao acesso a qualquer tipo de SGBD, nomeadamente o IP e porta, podendo este adaptador ser estendido, por exemplo, com as especificidades do acesso ao MongoDB.

Passando agora ao acesso mais comum, via `HTTP`, neste nível são fornecidas as funções necessárias para realizar os pedidos HTTP. Nestas estão incluídos métodos para acessos tanto síncronos como assíncronos. Ou seja, métodos em que se obtém o recurso logo após o pedido, e métodos em que é necessário aguardar por uma notificação que indique que o recurso já se encontra disponível. Caso esta seja enviada via email, pode existir a necessidade de recorrer a um cliente email, de forma a evitar uma estratégia de *polling*, o que certamente levaria a um desperdício de recursos.

Descendo ao nível da `API REST`, neste são adicionados atributos que permitem gerir as páginas de um certo pedido. Ou seja, através da página de início e do tamanho de cada uma, é fornecido um mecanismo que permite gerar novas unidades de trabalho para cada uma das páginas.

Relativamente ao `API REST - OT`, neste são fornecidas funções específicas para API REST relativas a dados de observação terrestre. Nomeadamente, acrescentando detalhes sobre o produto como o id, o tipo e o fornecedor nos *logs* de *fetch*.

Terminando a hierarquia, o adaptador `FTP` consiste basicamente num *watcher* a uma pasta que verifica se foi adicionada informação via FTP nessa pasta, de modo a processá-la.

Por fim, passando aos adaptadores por composição, existe o de `Autenticação`, que gere as credenciais e *tokens*. E por outro lado, o módulo de `Periodicidade` que fornece ao escalonador a informação de que ele precisa para agendar as unidades de trabalho. Esta informação inclui funções de gerar unidades de trabalho a partir das anteriores, avançando a janela temporal, por exemplo, fornecendo também os mecanismos necessários de recuperação do histórico.

De forma a ilustrar, no caso do adaptador para o Copernicus Open Access Hub, este apresenta duas principais API - a de catalogação e a de acesso. A primeira herda todas as pré-definições até à folha `API REST - OT`, sendo esta composta também pelos adaptadores de Autenticação e Periodicidade. Relativamente à de acesso, esta estende a mesma folha que a de catalogação, mas não apresenta o adaptador de Periodicidade.

Acrescentando uma nota final, na eventualidade de se querer adicionar uma nova fonte quando o sistema já se encontra em produção, basta seguir este método de especificação, e atualizar o sistema, sendo o fluxo igual ao descrito na Secção 3.3.2.

3.4 Metamodelo

Nesta secção são apresentadas as funcionalidades relativas ao metamodelo dos produtos de observação terrestre. Numa primeira fase, na Seção 3.4.1, é apresentado o mecanismo de procedimento de especificação dos metamodelos dos múltiplos tipos de produto. Estando definido este mecanismo, importa abordar os cenários de extensibilidade dos metamodelos especificados (Secção 3.4.2). Finalmente, na Secção 3.4.3, propõe-se um procedimento declarativo de especificação de ETL, o que inclui: obter o ficheiro; operar a extração; aplicar, eventualmente, uma transformação; e, por fim, mapear a informação pré-processada no metamodelo.

3.4.1 Procedimento de Especificação

Na ingestão dos metadados, a última fase de ETL é o mapeamento da extração no metamodelo. Para que tal aconteça, é importante que o metamodelo esteja bem definido. Caso contrário, se existe liberdade total na metamodelação, a catalogação dos produtos seria bastante mais complicada. De realçar que esta especificação conceptual é independente da linguagem usada para esse efeito.

Tão ou mais importante que a instanciação do metamodelo é a técnica de especificação, já que esta contribui para a extensibilidade do mesmo (Secção 3.4.2). É justamente por aqui se começa a descrição.

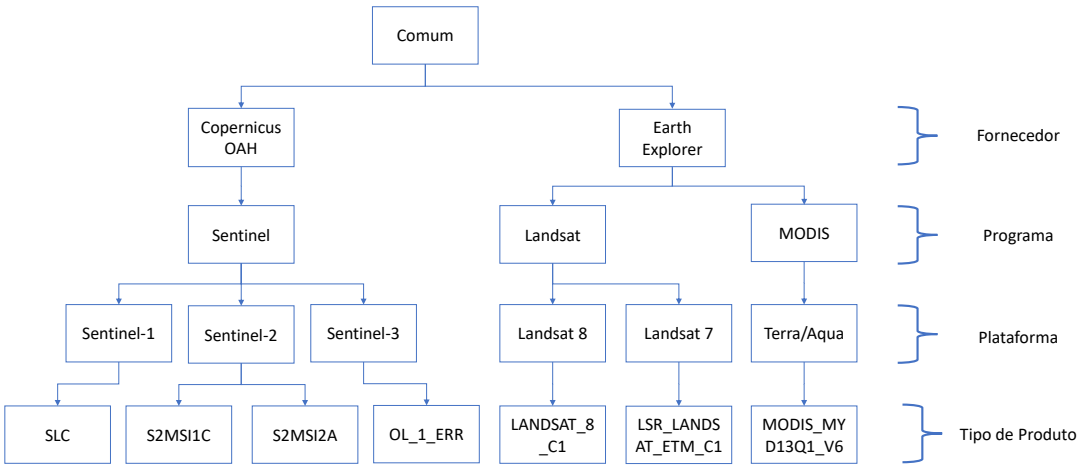


Figura 3.5: Técnica de especificação hierárquica do metamodelo, acompanhada de uma possível instanciação. OAH - Open Access HUB.

Recorrendo à técnica de metamodelação ilustrada na Fig. 3.5, existe uma herança de atributos desde da raiz até às folhas, as quais representam tipos de produtos. De notar que esta representação não é exaustiva, apresentando-se apenas uma seleção de produtos, que facilmente poderia ser outra. Em relação aos níveis definidos, estes são os seguintes: `Comum`, `Fornecedor`, `Programa`, `Plataforma`, `Tipo de Produto`.

O primeiro, tal como o nome indica, trata-se dos metadados que são comuns a qualquer tipo de produto. Descendo ao nível dos fornecedores, os catálogos destes, apesar de terem alguma informação comum entre eles, apresentam metamodelos distintos. Sendo que em alguns casos trata-se apenas de uma diferença sintática e de representação e não semântica, que é resolvida através do pré-processamento. Já ao nível do programa, as diferenças entre metamodelos já se tornam assinaláveis. De referir que nos catálogos que só disponibilizam produtos de um programa, como o Copernicus OAH com o Sentinel, a camada de Programa geralmente adiciona poucos elementos de metainformação. Dentro do mesmo programa existem diferentes tipos de satélites, sendo que estes, obviamente, apresentam as suas próprias necessidades de metamodelação. Por fim, os dados captados pelas diferentes plataformas são empacotados em diferentes produtos, nomeadamente, de acordo com o instrumento usado ou o nível de processamento. Observe-se, por exemplo, o caso do Sentinel-2 que apresenta os tipos de produto tanto de nível 1 como 2. A mesma situação se verifica com o Sentinel-3, cujo tipo de produto representado é o resultante de um dos múltiplos instrumentos presentes na sua plataforma.

A principal vantagem desta abordagem é que, no momento em que se quiser adicionar um novo tipo de produto, não é necessário definir completamente o metamodelo do mesmo. Permitindo a flexibilidade de estender, em qualquer momento, os metamodelos já definidos (Secção 3.4.2).

De modo a ilustrar uma especificação concreta, será apresentada uma possível metamodelação dos produtos Sentinel-2 de nível 2. Para este efeito, descer-se-á por nível conceptual até chegar ao tipo de produto.

Comum. Todos os tipos de produto presentes nesta plataforma herdam a metainformação definida a este nível. Esta inclui os mais genéricos: identificador único do produto; título; fornecedor; programa; plataforma e tipo de produto. Algo transversal a todos estes tipos de produto é a componente espaciotemporal que é descrita por: data de início e fim de aquisição; e um polígono com a cobertura espacial do produto. Adicionalmente, existe um campo de metainformação designado extensões, cujo objetivo é que os diferentes utilizadores acrescentem novos metadados (descrito em maior detalhe na Secção 3.5). Por fim, existe um conjunto de elementos de apontadores para dados e metadados. Nomeadamente, apontadores para as imagens presentes no sistema de ficheiros, ou até máscaras de nuvens, no caso de metadados. Genericamente, cada um destes apontadores é enriquecido de metainformação como o identificador do ficheiro, o tamanho do ficheiro e o estado do mesmo (*i.e.* remoto ou local).

Copernicus OAH. Descendo para este nível, os principais elementos são o número e direção da órbita, sendo também acrescentado um apontador para ficheiros de metainformação que são armazenados aquando da ingestão. Como, por exemplo, o ficheiro manifesto do produto, que descreve a estrutura do produto, no formato SAFE, abordado na Secção 2.2.2.

Sentinel. Neste nível já é possível definir melhor os apontadores para as imagens. Já foi referido que estes estão incluídos num conjunto de metainformação, mas no caso de imagens é acrescentada a banda, o *tile* que representa e a resolução espacial da mesma. De facto, existe a necessidade de guardar informação sobre o *tile*, pois em certos produtos estavam incluídas imagens de mais do que um *tile*.

Sentinel-2. O principal metadado acrescentado neste nível é a cobertura de nuvens na cena e o identificador do *datatake* no qual foi gerado o produto.

S2MSI2A. Já ao nível do tipo de produto, está disponibilizada informação resultante do processamento do produto de nível 1, nomeadamente, percentagens de: píxeis com grande ou média probabilidade de estarem cobertos por nuvens; píxeis classificados como sendo ou não vegetação; píxeis nos quais está presente gelo ou neve; píxeis com água; e, por fim, píxeis não classificados.

3.4.2 Extensibilidade

Uma das caraterísticas a melhorar nos repositórios existentes, é justamente a extensibilidade dos seus metamodelos, como referido na Secção 1.2. Atualmente, estes são demasiado genéricos e quando é necessário integrar aplicações específicas, estendendo os elementos de metainformação para os restritos desse domínio, não é permitida essa flexibilidade. Ilustrando o referido, uma equipa que esteja a trabalhar na área agrícola necessita de metainformação sobre vegetação, que um metamodelo genérico não acomodaria. Ao ter esta extensibilidade, a própria expressividade das interrogações é melhorada, devendo esta ser aliada a um mecanismo de interrogação de alto nível. Esta extensibilidade não se deve limitar aos momentos anteriores ao *deploy* do sistema, ou seja, é importante que seja possível atualizar os metamodelos *online*, sem prejudicar a disponibilidade dos serviços. Relativamente aos cenários de extensibilidade existentes, estes incluem-se fundamentalmente nos seguintes:

Novas extrações. Na configuração do subsistema de ingestão são especificadas as extrações que são executadas para cada tipo de produto. Esta especificação é feita de forma declarativa através de uma linguagem de especificação de ETL, analisada em detalhe na Secção 3.4.3. Desta forma, é possível acrescentar dinamicamente novas extrações. Para este efeito, é necessário registar, no esquema do metamodelo, o novo elemento. De seguida, é declarada a nova extração, sendo que existe um aspeto a considerar: é necessária transformação ou não. Caso seja, não é possível adicionar a extração de forma completamente declarativa, havendo a necessidade de programar a transformação. Na Secção 3.4.3 são descritos em detalhe exemplos deste mecanismo.

Metainformação customizável. Outro dos mecanismos de extensibilidade, é a possibilidade de qualquer utilizador publicar as suas próprias extensões do metamodelo correspondente a um tipo de produto. Para este efeito, este deve primeiro registar essa extensão no esquema do metamodelo. Este registo requer a seguinte informação: identificador da extensão; autor; data; nome do atributo; tipo do atributo; e o identificador do tipo de produto a que está a adicionar esta extensão. Estas são informações que visam certificar a proveniência destes metadados. De notar que o identificador da extensão permite agrupar um conjunto de novos atributos.

Exemplificando, uma extensão de metadados da área de gestão de incêndios poderá consistir em múltiplos novos atributos, com um identificador de extensão comum. Dando um exemplo mais concreto, um utilizador poderá estender o metamodelo dos produtos nível 2 do Sentinel-2 com informação sobre a área ardida. Mais especificamente, poderá ser integrado um ficheiro em formato vetorial nos metadados, no qual cada polígono representa uma subárea ardida. Para além disto, deve ser também acrescentado ao metamodelo um agregado relativo à percentagem de píxeis de toda a imagem Sentinel-2 que correspondam a uma área ardida. Depois da submissão desta extensão, o próprio ou outros utilizadores poderão adicionar esta metainformação às diferentes instâncias daquele tipo de produto, após o processamento por parte dos mesmos.

Novos tipos de produtos. Este cenário consiste não tanto na extensão de um metamodelo de um tipo de produto em concreto, mas sim numa inserção no conjunto de metamodelos. Com esta finalidade em mente, o primeiro passo é o registo do esquema do metamodelo - seguindo as instruções definidas na Secção 3.4.1. Sendo que, neste contexto, há basicamente dois mecanismos de ingestão: automática e *ad hoc*. No primeiro referido, para além de se registar o metamodelo, é necessário declarar as extrações com a linguagem de ETL, já múltiplas vezes referidas ao longo da presente Secção. De notar que apenas o administrador do sistema poderá completar esta operação. Por outro lado, no caso de uma ingestão *ad hoc*, o registo do metamodelo é suficiente, pois serão os utilizadores que ficarão encarregues de submeter os produtos computados via API (Secção 3.5).

3.4.3 Linguagem de Especificação de ETL

Tipicamente, os processos ETL estão bem definidos antes do sistema entrar em produção. Consistindo estes num conjunto de extrações estáveis ao longo do ciclo de execução do sistema. Neste contexto, sempre que é necessário atualizar este processo, é necessário passar por uma fase de desenvolvimento e testes, o que representa um custo significativo. Neste repositório, os metamodelos são dinâmicos, e portanto é importante que existam mecanismos que simplifiquem o processo de acrescentar extrações, reduzindo o custo e aumentando a extensibilidade. Para este efeito, foi desenvolvida uma linguagem declarativa de especificação do pré-processamento.

Tal como na especificação do metamodelo (Secção 3.4.1), esta também é definida hierarquicamente. Ou seja, para além das extrações especificadas ao nível do tipo de produto, são herdadas todas as de níveis superiores. Explicitando, às extrações definidas ao nível do Sentinel-2 nível 2 são acrescentadas as herdadas ao nível do Sentinel-2, e assim sucessivamente.

Concretizando, para cada uma das extrações são declarados um conjunto de atributos, dos quais uns são obrigatórios e outros opcionais. Estes serão aqui apresentados, estando os obrigatórios assinalados a negrito:

(i) **Nome**: identificador único da extração;

(ii) **Tipo de resultado**: tem como possíveis instanciações: univalor; multivalor; ficheiro e pasta. No caso do univalor e mulivalor, estes são mapeados diretamente para o metamodelo. Já no caso do ficheiro e pasta, o que estará presente no metamodelo são apontadores para dados presentes no sistema de ficheiros. De referir que caso seja pasta todos os atributos referidos a seguir não são aplicáveis, exceto o contexto, formato do contexto e mapeamento no metamodelo;

(iii) **Tipo de dados**: trata-se do tipo em concreto. No caso dos univalor e multivalor, as possíveis atribuições são: objeto, inteiro, booleano, data, string ou decimal. O objeto é uma estrutura de dados na qual podem estar embebidos múltiplos campos do tipo univalor ou multivalor. A título de exemplo, uma extração que obtenha os dados de uma pessoa terá incluídos múltiplos campos. No caso do ficheiro, o tipo de dados trata-se do formato do mesmo (*e.g.* XML ou JSON);

(iv) **Contexto**: este atributo identifica qual o contexto em que é executada a extração. Nomeadamente, se for uma extração de um campo presente num certo ficheiro, o contexto é o identificador do próprio ficheiro. Caso a extração corresponda à totalidade do ficheiro, o contexto é o próprio ficheiro. Por outro lado, se for uma pasta, é necessário um ficheiro manifesto que descreva a organização das pastas, sendo que este representa o contexto;

(v) **Formato do contexto**: corresponde ao formato do ficheiro no qual é executada a extração. Tipicamente, o formato é XML ou JSON;

(vi) ***Query***: tendo sido o ficheiro descarregado, é necessário aplicar a extração. Para este efeito, recorre-se a uma linguagem de consulta em ficheiros (*e.g.* XPath, JSON-Path, ...). Se se tratar da extração de um ficheiro na sua totalidade, aplica-se a *query* que corresponde à raiz. Por outro lado, é também possível extrair um fragmento e armazená-lo como ficheiro;

(vii) Tipo de pós-transformação: após executar a extração, opcionalmente, passa-se a uma fase de transformação. É importante, muitas das vezes, aplicar algum tipo de pré-processamento, nomeadamente, devido à normalização de sintaxe e formatos.

De realçar que esta especificação não é feita de forma declarativa, sendo necessário programar cada uma das transformações. Ainda assim, é necessário especificar o tipo de dados após a transformação, mapeando o resultado da mesma no metamodelo (descrito no ponto mais abaixo - Metamodelo) ;

(viii) Formato da data: este trata-se de um campo que apenas necessita de ser declarado caso o tipo do resultado seja data e se deseje aplicar um pré-processamento à mesma. A verdade é que se pode declarar esta particularidade como sendo um detalhe de implementação;

(ix) **Metamodelo**: por fim, chega-se à fase de *load* dos típicos processos ETL. Após todo o processamento descrito até aqui, o elemento é carregado no metamodelo, sendo para isso necessário indicar qual o nome do atributo do metamodelo no qual é mapeado. Para este efeito, existe uma conexão com o SGBD, e é inserido o resultado deste pré-processamento no atributo declarado. No caso de ser um ficheiro ou uma pasta, os dados são armazenados no sistema de ficheiros, enquanto os apontadores respetivos são carregados no metamodelo.

Tabela 3.1: Tabela de possíveis instanciações de especificação de ETL para diferentes atributos. Omissão das expressões das *queries* por serem demasiado extensas.

	Exemplo 1	**Exemplo 2**	**Exemplo 3**	**Exemplo 4**
i	cobEspacialExt	diaAnoAqInícioExt	máscNuvensExt	máscExt
ii	univalor	univalor	ficheiro	pasta
iii	string	data	XML	XML
iv	copernicusMD	earthexplorerMD	copernicusManif	copernicusManif
v	XML	JSON	GML	-
vi	<expressão>	<expressão>	-	-
vii	objeto	inteiro	JPEG2000	-
viii	-	YYYY/MM/dd	-	-
ix	coberturaEspacial	diaAnoAqInício	máscaraNuvens	máscaras

Antes de iterar sobre cada um dos exemplos presentes na Tabela 3.1, importa referir algo que é transversal a todos que é o facto de existir um *log* para cada uma das extrações. Sendo que este apresenta dois conjuntos de atributos, os relativos à obtenção do recurso e outro de descrição do resultado. Relativamente ao primeiro, é guardado o URL, o nome da extração, a data da obtenção, a *query* aplicada, e a informação sobre o contexto. Já em relação ao resultado, é armazenado o valor após pré-processamento. A vantagem da existência deste *log* de todas as extrações é permitir certificar a origem dos metadados e, por outro lado, executar verificações de integridade periódicas. De facto, este mecanismo contribui para a auditabilidade do sistema.

De notar que, naturalmente, as extrações que incidem sobre o mesmo ficheiro são agrupadas de forma a evitar descarregar o mesmo mais do que uma vez.

Outro ponto que importa abordar é o facto de existir a necessidade de normalizar a especificação, isto é, recorrer apenas a uma linguagem de extração, em vez de usar uma para cada formato de documento. Para este efeito, é selecionado um formato de documento no qual todos serão mapeados, sendo esta uma decisão de implementação. Ilustrando o que foi referido anteriormente, ao escolher traduzir para XML qualquer documento JSON será mapeado para XML, sendo que a linguagem de extração passa a ser, por exemplo, XPath. No entanto, é importante garantir que o método de tradução não resulta em perda de informação, ou seja, deverá ser reversível. Caso contrário é importante armazenar o documento original, por uma questão de rastreabilidade. Para mais detalhes relativos às particularidades desta tradução, consultar Secção 4.2.4.

Exemplo 1. Este consiste na extração da cobertura espacial de qualquer tipo de produto Sentinel. O tipo de resultado é univalor, mais concretamente uma string. Esta cobertura está representada em WKT [128] - que se trata de um formato textual de representação vetorial. Relativamente ao contexto na qual é executada esta extração, quando se identifica a existência deste produto através da API de catalogação, é fornecido um ficheiro em XML com um resumo dos metadados do mesmo.

De notar que o documento poderá ter de ser traduzido, dependendo da implementação. De seguida, é aplicada a extração, através da *query*. Em relação ao pré-processamento do valor obtido, o objetivo deste passa por unificar o formato com o de outros fornecedores. Para este efeito, existe uma conversão de WKT para GeoJSON. Para finalizar, o objeto, representado neste último formato referido, é mapeado no atributo `coberturaEspacial` do metamodelo.

Exemplo 2. Passando ao segundo exemplo, este trata-se da extração do dia do ano do início da aquisição de um certo produto. A motivação desta extração passa por adicionar uma maior expressividade temporal às interrogações. Sendo que, neste caso, o contexto é o ficheiro de metadados do EarthExplorer. O procedimento é semelhante ao do Exemplo 1, mas a principal diferença é na fase de transformação.

Tratando-se do pré-processamento de uma data, é obrigatório declarar qual o formato da extração, de forma a que a biblioteca usada consiga fazer o *parse*. Seguidamente, é fácil extrair do dia do ano, já que existem múltiplas bibliotecas que fornecem esta funcionalidade. De notar que pode considerar-se que a necessidade deste procedimento é um detalhe de implementação, ainda assim, este é um problema transversal a qualquer solução prática. Finalmente, após a transformação, passa-se ao passo sempre presente de mapeamento do metamodelo.

Exemplo 3. Passando agora para uma extração do tipo ficheiro, esta refere-se à máscara de nuvens em formato GML, fornecida em produtos Sentinel-2. Relativamente ao contexto em que esta é extraída, é através do ficheiro manifesto (XML) fornecido para produtos Sentinel, que se obtém a localização deste recurso.

No que toca ao pré-processamento, neste caso está representada uma transformação de um ficheiro vetorial (GML) para *raster* (JPEG2000). Sendo que este poderá enquadrar-se melhor em algum tipo de processamento a realizar pelos utilizadores. Finalmente, o ficheiro será armazenado no sistema de ficheiros, e será guardado um apontador no atributo do metamodelo designado máscaraNuvens.

Exemplo 4. Para finalizar o conjunto de exemplificações, aborda-se agora uma extração por extensão, ou seja, da totalidade de uma pasta. Neste caso em concreto trata-se de uma pasta presente nos produtos Sentinel-2, na qual estão incluídas alguns indicadores de qualidade como máscaras de nuvens, de defeitos ou de ausência de dados. Sendo que é através do ficheiro manifesto que se localiza todos os ficheiros presentes numa certa pasta da estrutura do produto. Desta forma, estes ficheiros serão todos ingeridos na plataforma, reconstruindo, no sistema de ficheiros, a pasta correspondente. Por fim, no metamodelo é registado o apontador para essa mesma pasta.

3.5 API

Até esta fase da descrição da solução, o foco foram os diferentes mecanismos internos do sistema. Algo que ainda não foi esmiuçado foi a interação entre o repositório e os seus utilizadores via API. Esta tem como suporte vários dos mecanismos abordados até esta fase. De referir que na Secção 4.2.5 é concretizado o mapeamento das diferentes operações em *endpoints*, sendo que estas subdividem-se em duas categorias de recursos acessíveis: os produtos e os seus metamodelos.

Produtos. Relativamente a este recurso, as operações disponibilizadas são as seguintes:

- **Obter metadados**: trata-se de consultar o SGBD, obtendo-se a metainformação do produto identificado pelo utilizador;
- **Obter dados**: trata-se de consultar o sistema de ficheiros, transferindo os dados da plataforma para a máquina do utilizador;
- **Pesquisar**: é dos parâmetros desta pesquisa que vive a expressividade do sistema. Estes parâmetros são um mapeamento dos elementos presentes no metamodelo dos vários produtos, uma vez que existem *endpoints* que permitem obter esta informação relativa aos metamodelos;
- **Adicionar produto**: nesta operação é necessário submeter os metadados do produto e também um *zip* com os dados do mesmo, sendo que se for a primeira vez que se adiciona um produto deste tipo é necessário registar o seu metamodelo primeiro. Caso já esteja registado, os metadados serão validados de acordo com o metamodelo e, de seguida, os metadados serão guardados no SGBD e os dados extraídos para o sistema de ficheiros;

- **Atualizar metadados**: esta operação trata-se da inserção *ad hoc* de metainformação num produto e, por isso, necessita primeiro que tal seja registado no metamodelo;
- **Pedir obtenção de dados**: após a pesquisa de produtos pode existir a necessidade de descarregar dados que de momento apenas estão acessíveis remotamente. Ao realizar esta operação, a API REST comunicará com o subsistema de ingestão, ficando este responsável por esta tarefa;
- **Remover dados de produto**: a obtenção referida no parágrafo anterior está, normalmente, associada a um passo seguinte de exploração e computação dos dados obtidos. Se, após esse processamento, identificar-se que não há planos futuros de utilização dos mesmos, o utilizador procede à sua remoção.

Metamodelos. Passando às operações de manipulação do metamodelo apresentadas, sendo que estas englobam-se nas abaixo descritas:

- **Obter metamodelos**: este *endpoint* permite ao utilizador consultar o metamodelo de cada um dos tipos de produto. Em cada metamodelo estão incluídas anotações que referem a existência de índices para os diferentes atributos, fornecendo assim os dados necessários para melhorar a eficiência das interrogações por parte do utilizador. Para finalizar, existe a necessidade desta funcionalidade, pois os metamodelos são dinâmicos, sendo estes frequentemente atualizados, nomeadamente por parte dos utilizadores;
- **Adicionar metamodelo**: antes de adicionar um novo tipo de produto é necessário o registo do metamodelo do mesmo. É através deste *endpoint* que é realizada essa operação;
- **Atualizar metamodelo**: apenas o administrador do sistema pode realizar esta operação, dado que se recorre a esta quando se adiciona uma nova extração, de modo a adicionar o novo elemento de metainformação no qual será mapeado o resultado da extração;
- **Adicionar extensão**: tal como referido na operação Atualizar metadados no parágrafo relativo aos produtos, antes dessa operação é necessário registar essa extensão por parte do utilizador. Este registo necessita da informação descrita no parágrafo Metainformação customizável da Secção 3.4.2.

3.6 Conclusão

Ao longo deste capítulo foram descritos um conjunto de mecanismos e componentes que constituem a solução proposta - uma plataforma de desenvolvimento colaborativo de produtos de observação terrestre. Abordou-se a heterogeneidade das fontes e de que

forma esta é integrada na plataforma; definiu-se um claro fluxo dos dados e metadados; apresentou-se a técnica de especificação do metamodelo; referiu-se os possíveis cenários de extensibilidade do mesmo, juntamente com uma linguagem para especificar as extrações necessárias; e, por fim, foram definidas as operações disponibilizadas aos utilizadores via API. Cada um destes aspetos contribui para a resolução do problema condensado nos quatro pontos referidos na Secção 1.2.

O objetivo desta solução não é substituir plataformas como o GEE (Secção 2.4.5) ou repositórios como o Copernicus Open Access Hub (Secção 2.4.1). Esta é indicada para equipas que trabalhem num domínio aplicacional comum, pois a flexibilidade da mesma permite que se adicione facilmente produtos específicos do domínio e, por outro lado, se enriqueça o metamodelo com dados da disciplina em que se está a trabalhar. Adicionalmente, os produtos computados são armazenados e disponibilizados de uma forma sistemática. De realçar que o foco principal da plataforma proposta não é a computação, apesar de existirem componentes para esse fim. O foco é sim criar uma *framework* de gestão colaborativa de produtos de observação terrestre, tornando-se desta forma possível que as equipas possam gerir o seu próprio catálogo e plataforma.

Em relação a catálogos como o Copernicus, esta plataforma não se apresenta como uma alternativa, mas sim como uma extensão, pois os produtos são obtidos nesses mesmos catálogos. No entanto, um dos propósitos é evitar que os utilizadores tenham de consultar diferentes catálogos, em que cada um apresenta interfaces, metamodelos e formatos distintos.

Comparando com o GEE, a preocupação deste é ser genérico no seu metamodelo e nos produtos disponibilizados, não existindo um envolvimento com o domínio em que os utilizadores estão a trabalhar. No entanto, existem vários cenários em que recorrer ao GEE faz todo o sentido, nomeadamente, se a computação for pesada. Nesse caso, o GEE abstrai toda a distribuição da computação através do paradigma de programação que oferece nas suas API.

No que diz respeito ao modelo de computação desta plataforma, este é semelhante ao do CREODIAS. Ou seja, não é imposto um paradigma como no caso do GEE. Apenas é disponibilizada a infraestrutura para a computação. Por outro lado, na plataforma proposta, a gestão do armazenamento e disponibilização dos produtos computados é bastante mais rica que o CREODIAS, aproximando-se do CMR ao nível da gestão dos metadados, uma vez que não apresenta ferramentas de correção dos metadados tão satisfatórias, mas ao nível da extensibilidade dos metamodelos acaba por ser mais avançado.

Finalmente, referir que no Capítulo 4 encontra-se toda a descrição do desenvolvimento do protótipo, o qual incide sobre um subconjunto representativo das funcionalidades definidas ao longo do presente capítulo.

Capítulo 4

Protótipo

Após apresentar a abordagem (Capítulo 3), importa descrever todo o processo de desenvolvimento do protótipo. Antes de começar a desenvolver, especificou-se o protótipo a implementar (Secção 4.1), sendo que este processo subdivide-se em três etapas principais: definição de requisitos (Secção 4.1.1), instanciação da arquitetura (Secção 4.1.2) e definição da infraestrutura (Secção 4.1.3). Na primeira, é definido o subconjunto de funcionalidades a implementar no protótipo. Já na segunda parte, é instanciada a arquitetura referida na Secção 3.2, referindo as tecnologias a que se recorreu. Para finalizar, na Secção 4.1.3, é apresentada a infraestrutura que suportará essa arquitetura.

Após a especificação do protótipo, será detalhado o desenvolvimento do mesmo na Secção 4.2. Para este efeito, seguiu-se, aproximadamente, a ordem de descrição do Capítulo 3: começou-se por referir a configuração das fontes de dados (Secção 4.2.1); passou-se à especificação do metamodelo (Secção 4.2.2); adiantou-se detalhes relativos à especificação hierárquica de ETL (Secção 4.2.3); ainda neste contexto, apresentou-se alguns detalhes relativos a traduções de XML para JSON sem perdas (Secção 4.2.4); mapeou-se os *endpoints* da API (Secção 4.2.5); e, por fim, descreveu-se a automatização do *deploy* (Secção 4.2.6).

Finalmente, na Secção 4.3, é apresentada uma reflexão relativa a todo o processo de desenvolvimento deste protótipo.

4.1 Especificação

Nesta fase da descrição da solução proposta é percetível a complexidade e extensão deste sistema. Por este motivo, o protótipo desenvolvido reflete um subconjunto representativo das funcionalidades apresentadas. A justificação da ausência de certos aspetos insere-se fundamentalmente em duas vertentes: na necessidade de evitar um aprofundamento exagerado nas funcionalidades que não contribuam para a prova de conceito; e,

por outro lado, dado o volume de componentes interessantes deste sistema, selecionar as mais importantes, tendo em conta as restrições de tempo. Adicionalmente, nesta secção serão reveladas algumas decisões e adaptações de desenho que, apesar de não serem necessárias para a compreensão geral do sistema, incluem vários aspetos de engenharia.

4.1.1 Requisitos

Até esta fase do presente documento já foram cobertos os elementos que podem ser incluídos neste protótipo. Portanto, em vez de elaborar uma lista exaustiva de requisitos, serão apontados os aspetos da solução que não estão incluídos no protótipo.

Relativamente à arquitetura, na Fig. 3.1 está representada uma área de computação *ad hoc* na qual cada utilizador está isolado dos demais, tendo recursos alocados apenas a si. No entanto, esta solução exigiria um mecanismo de gestão de *containers*, o qual cairia fora do âmbito desta dissertação, dada a complexidade de tal mecanismo. Portanto, o que é disponibilizado neste protótipo é um conjunto de máquinas, nas quais os utilizadores podem aceder de forma a computar os seus produtos, sendo que o processamento não se encontra isolado dos outros. Outra das componentes que não está presente é a interface gráfica, tendo-se já referido que esta foge ao âmbito desta dissertação, pois a visualização deste tipo de produtos constitui uma área de investigação distinta.

Passando às fontes de dados, nem todos os adaptadores presentes na Fig. 3.4 serão implementados. Tendo-se focado na forma de acesso a produtos de deteção remota mais popular, ou seja, no acesso via HTTP síncrono. Seguindo a classificação de API da Secção 3.3.1, as suportadas por este protótipo são as de catalogação e acesso. Estando ausentes as de encomenda, cuja interação é assíncrona. Esta ausência deve-se a estas serem utilizadas, principalmente, em situações em que os produtos estão *offline*, o que é a exceção e não a regra.

Por fim, no contexto da gestão dos dados, na Secção 3.3.2, mais especificamente no parágrafo *Download* dos Produtos, são apresentados os diferentes mecanismos para esse fim. No entanto, neste protótipo o único implementado foi o *prefetch*. A justificação para esta decisão passa por se ter considerado que este mecanismo é suficiente para a prova de conceito, já que através deste é possível fazer o *download* de qualquer produto. Sendo que, no caso do mecanismo de configuração *a priori*, este trata-se apenas de uma otimização para os tipos de produto frequentemente solicitados.

4.1.2 Instanciação da Arquitetura

Tendo os requisitos do protótipo definidos, passa-se à fase de instanciação da arquitetura com as tecnologias selecionadas. As diferentes escolhas resultaram de uma avaliação conceptual das diferentes possibilidades que se adequam a cada uma das componentes.

Resumindo as decisões de arquitetura, a componente de ingestão foi implementada em Scala, recorrendo à biblioteca Akka - para a implementação do protocolo de distribuição de ingestão. No que diz respeito à API REST, esta foi implementada em Java,

recorrendo à *framework* Spring. Movendo o foco para os serviços de armazenamento, o SGBD selecionado foi o MongoDB e Cassandra. E por fim, de forma a empacotar cada um destes serviços, recorreu-se às tecnologias Docker - o que contribui para uma gestão mais facilitada das operações.

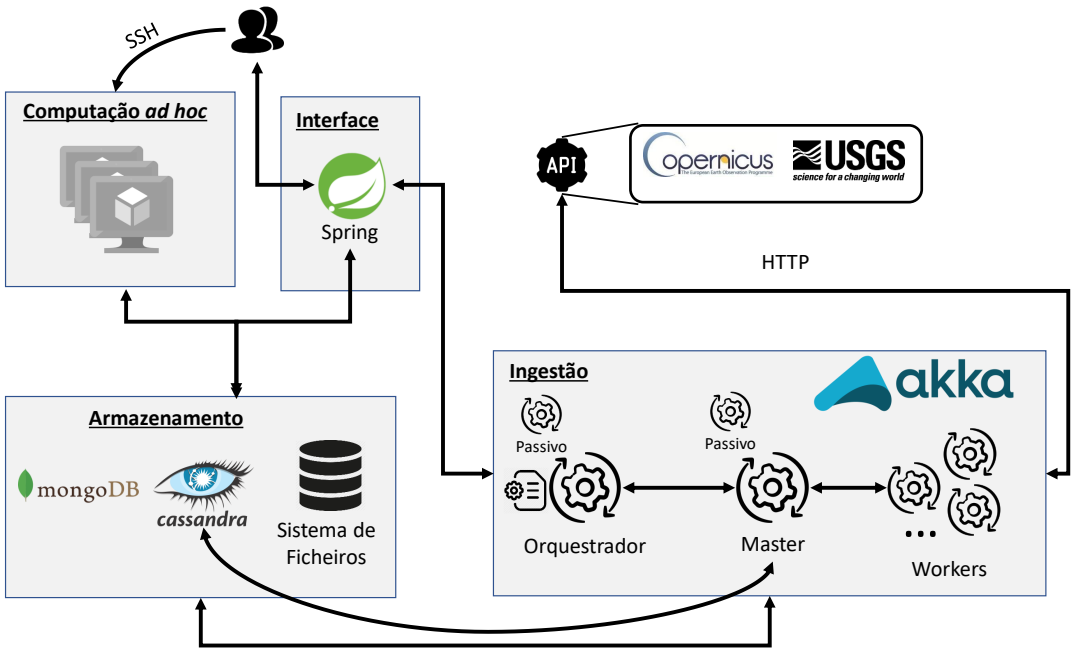


Figura 4.1: Instanciação da arquitetura do sistema, com ilustração das tecnologias usadas.

Spring [129]. Trata-se de uma *framework* Java que implementa um padrão de desenho de inversão de controlo e injeção de dependências. Nesta está incluído um módulo (Spring MVC) que suporta, entre outras funcionalidades, a implementação de API REST.

As vantagens que recorrem de usar esta *framework* são múltiplas, dado que estas surgem da maturidade desta solução, ou seja, é uma *framework* que tem sido aperfeiçoada ao longo dos anos. Mais especificamente, o ciclo de desenvolvimento é acelerado por aspetos como: a abstração de detalhes de comunicação entre componentes; a modularidade oferecida por *templates* pré-definidos; e a fácil integração com múltiplos SGBDs; e até submódulos de segurança. Adicionalmente, as vantagens não surgem apenas no contexto do desenvolvimento, mas também em termos de escalabilidade. Apesar de todas estas caraterísticas, esta *framework* mantém a simplicidade, o que contribui para o excelente suporte que a comunidade oferece.

Akka [123]. Passando à componente de ingestão, de forma a implementar o protocolo de comunicação entre as subcomponentes, recorreu-se à biblioteca Akka. O objetivo desta é suportar aplicações baseadas em mensagens que necessitem de altos níveis de concorrência, distribuição e resiliência. Todas estas são caraterísticas que se pretende que o subsistema de ingestão apresente.

Esta biblioteca é suportada por um modelo de atores. Este trata-se de um modelo de computação concorrente, no qual a unidade fundamental é o ator. Sendo que as operações permitidas por este são: criar outro ator, enviar uma mensagem e especificar como será processada cada uma das mensagens recebidas. Este pode modificar o seu próprio estado, mas qualquer afetação do estado de outro ator é realizada indiretamente, através de uma comunicação assíncrona.

Este paradigma mapeia-se diretamente com o protocolo de ingestão (Secção 3.3.2). Existindo um ator para cada uma das componentes referidas - Orquestrador, Master, Worker. Após implementar o comportamento de cada tipo de ator, criar múltiplas instâncias do mesmo é simples. Adicionalmente, tal como indicado na definição do comportamento de um ator, é possível criar um ator-filho, e é justamente essa a relação existente entre Worker e WorkExecutor. Para além destes atores, existe um outro cuja função é correr um *endpoint* HTTP, que intermedeia a comunicação entre a API REST e o Orquestrador.

No que diz respeito à implementação, recorreu-se a Scala por esta linguagem ter uma melhor integração com o modelo de atores, sendo que cada ator corre isolado numa JVM. Finalizando, notar que esta solução foi baseada numa implementação presente no Lighbend Tech Hub [130].

MongoDB [131]. Este foi o SGBD selecionado para armazenar, indexar e disponibilizar os metadados dos produtos. A escolha recaiu neste SGBD pelos motivos de seguida enumerados, sendo que uma solução recorrendo a PostGIS [132] seria perfeitamente aceitável.

As vantagens do MongoDB incluem: a simplicidade no desenvolvimento; a flexibilidade do esquema, o que é muito importante neste repositório, pois os metamodelos são dinâmicos; expressividade espacial satisfatória; linguagem de interrogação simples e expressiva; integração com JSON, formato no qual grande parte dos metadados é apresentado originalmente; e, por fim, a possibilidade de escalar horizontalmente.

Relativamente ao PostGIS, as operações espaciais que este oferece são bastante mais ricas, incluindo funções de processamento de objetos GIS. No entanto, não oferece a flexibilidade do MongoDB na metamodelação, o que é um ponto vital neste protótipo. No que diz respeito à *performance*, segundo um artigo que compara os dois SGBD nesta vertente [133], o MongoDB em algumas operações espaciais básicas é 10 vezes mais rápido que o PostGIS, sendo que noutras chega às 25 vezes. As operações estudadas foram a interseção de linhas e inclusão de pontos, sendo que foram analisados diversos conjuntos de dados com número de *features* variável. Dado este resultado, os autores concluem que o uso do MongoDB adequa-se mais a sistemas de interrogação multiutilizador, que é o caso do sistema desenvolvido.

Cassandra [134]. Algo que ainda não foi abordado relativamente à instanciação da arquitetura é a presença do Cassandra e a sua relação com o subsistema de ingestão. Como referido na Secção 3.3.2, o Master persiste toda a informação necessária à gestão das

unidades de trabalho. A verdade é que o MongoDB não tem as caraterísticas necessárias para suportar este mecanismo. Nomeadamente, o *plugin* [135] de Cassandra para Akka Persistence, usa este SGBD de uma forma puramente orientada ao *log*, isto é, os dados apenas são inseridos e nunca atualizados. E, por outro lado, as escritas são feitas em *batch*. Ambas as particularidades referidas permitem otimizar o número de operações por segundo (*i.e. throughput*), o que é essencial na componente de ingestão.

4.1.3 Infraestrutura

De modo a suportar a arquitetura definida, é necessário construir uma infraestrutura que se adeque à mesma. Esta deve ter as habituais preocupações de robustez e escalabilidade, mas também de manutenibilidade e portabilidade. Relativamente às duas primeiras caraterísticas indicadas, a arquitetura definida já apresenta tecnologias e mecanismos que contribuem para assegurar estes aspetos. Passando à manutenibilidade da infraestrutura do sistema, é importante que exista automatização do *deploy* de novas versões para os diferentes serviços. Finalmente, a portabilidade está aliada à manutenibilidade, pois permite a instalação de um *cluster* em tempo reduzido, através da conteinerização dos vários serviços que constituem esta plataforma. Ambas as caraterísticas referidas permitem uma redução significativa nos custos de operação.

A tecnologia a que se recorreu de modo a garantir estes requisitos de infraestrutura foi o Docker [33]. Esta trata-se de uma plataforma de *containers* com o objetivo de partilhar e correr qualquer aplicação onde quer que seja. Na verdade, a principal vantagem é o tal empacotamento de *software*, que dá resposta à enorme velocidade da inovação, permitindo o *deploy* automatizado de novas versões. Por outro lado, para a orquestração dos *containers* nos quais correm os serviços desta plataforma, utilizou-se o Docker Swarm [136]. Este modo permite correr aplicações com múltiplos *containers* (*i.e.* serviços) em múltiplas máquinas, juntando as mesmas num *cluster* "dockerizado", designado *swarm*. De notar que estas máquinas podem tanto ser físicas como virtuais.

Na Fig. 4.2 está representado o *cluster* nos quais correm os diferentes *containers*. Uma das máquinas será o Swarm Manager, que é o responsável por gerir os outros nós do *cluster* (*i.e.* Swarm Nodes). Relativamente aos últimos referidos, estes limitam-se a executar um Docker Deamon que comunica com o Swarm Manager e arranca novos *containers* de acordo com a orquestração do nó Manager. Uma nota adicional é o facto de cada um destes nós ter um volume montado que é partilhado entre os mesmos.

Em cima desta infraestrutura correm-se as diferentes componentes da arquitetura em *containers* distintos. Para este efeito, o Swarm Manager lê um ficheiro que descreve toda a arquitetura, designado `docker-compose`. Este indica quais as imagens Docker que serão instanciadas e de que forma se interligam. Estas imagens são essencialmente executavéis de uma certa versão de uma das componentes da arquitetura. De modo a obter detalhes ao nível da implementação, é importante consultar a Secção 4.2.6.

Relativamente à arquitetura de *containers* suportada pelo *swarm cluster*, na Fig. 4.3 é

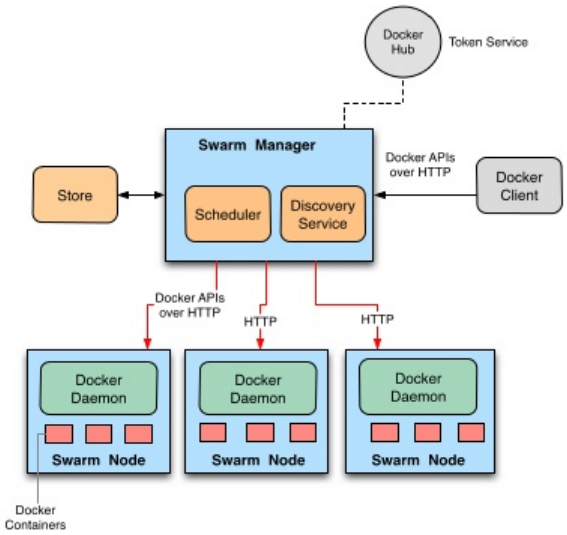


Figura 4.2: Representação de uma possível arquitetura de um *swarm cluster*, com um Swarm Manager e três Swarm Nodes [137]

possível observar as comunicações internas e externas ao sistema. Os *containers* presentes são um mapeamento das componentes da Fig 4.1, exceção feita à ausência da componente de Computação *ad hoc*. A justificação desta ausência passa pela complexidade de automatizar a criação de *containers* isolados, nos quais os diferentes utilizadores possam processar os seus produtos, como referido na Secção 4.1.1. Em alternativa, alguns dos utilizadores têm acesso a máquinas que não estão incluídas no *swarm cluster*, mas nas quais está montado o volume que é partilhado entre todos os nós do mesmo, acedendo deste modo aos dados dos produtos. Outra adaptação passa pela inclusão de um *container* no qual está a executar o ator que intermedia a comunicação entre a API REST e o Orquestrador.

De notar que todos os componentes de ingestão comunicam com o MongoDB. Isto deve-se à natureza dinâmica da configuração, ou seja, o metamodelo e as fontes de dados não são completamente estáticas ao longo do tempo; portanto, existe a necessidade de centralizar esta informação num SGBD.

Nesta representação não estão incluídas as réplicas de cada serviço, pois essas tratam-se apenas de novas instâncias das imagens. A verdade é que o número de réplicas de cada serviço poderá ser configurado no ficheiro que descreve esta infraestrutura (*i.e.* `docker-compose`), sendo que há casos que precisam de mais alguma configuração, como no caso da API REST em que é necessário configurar um *proxy* de acesso aos diferentes *containers*. Relativamente aos componentes de ingestão, a gestão das réplicas é feita ao nível do Akka. Sendo que se configurar duas réplicas Master no `docker-compose`, existem mecanismos que forçam a que apenas exista uma ativa, tornando as outras réplicas passivas.

Passando à relação dos diferentes *containers* com o volume partilhado, na pasta `/data`

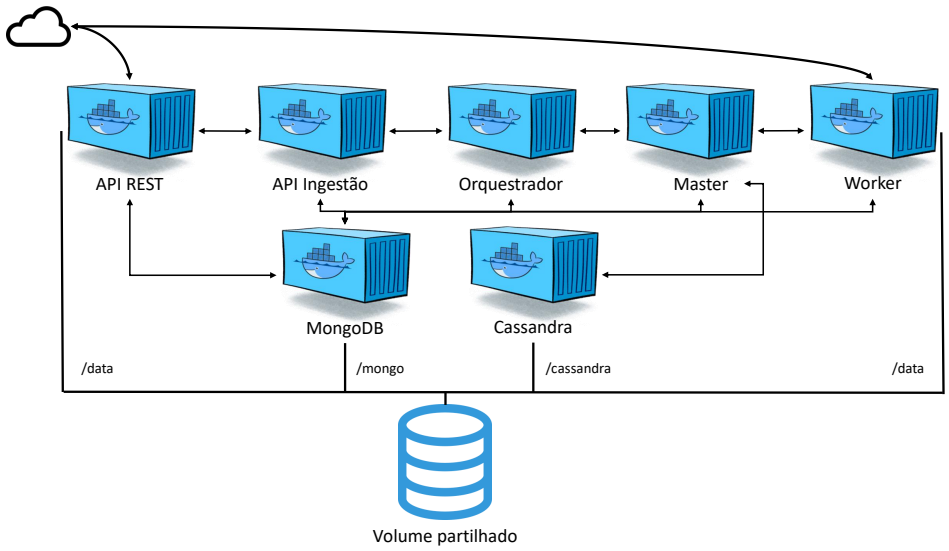


Figura 4.3: Arquitetura de *containers* Docker, com representação das interações entre os mesmos.

estão presentes os dados de observação terrestre. Os apontadores para estes estão armazenados nos metadados presentes no SGBD. É necessário montar esta pasta no *container* da API REST, pois é através desta que os utilizadores podem descarregar os dados dos produtos; e, por outro lado, submeter os seus próprios produtos, armazenando os dados dos mesmos na pasta referida. Relativamente ao Worker, este persiste os dados obtidos nessa mesma pasta. Para além desta área dos dados no volume partilhado, existe também uma área na qual são guardados a configuração e os dados presentes no SGBD.

Finalmente, assinalar que os únicos *containers* que têm interação com redes externas são a API REST e o Worker.

4.2 Detalhes de Implementação

Após especificar os requisitos, apresentar a instanciação da arquitetura e abordar a infraestrutura que a suporta, importa agora analisar alguns detalhes de implementação. Sendo que é importante existir uma clara separação entre a especificação da implementação do protótipo. A primeira deve ser feita *a priori*, já a segunda deve ser analisada *a posteriori* do desenvolvimento do protótipo.

4.2.1 Configuração das Fontes de Dados

As fontes de dados selecionadas estão condicionadas pelos produtos que se pretendem suportar neste repositório. Estes são indicados na Secção 4.2.2 e são disponibilizados através do Copernicus Open Access Hub ou do EarthExplorer.

Focando no caso do Copernicus, na Listagem 4.1 está representado um excerto da configuração da API de acesso na qual está incluída informação como: descrição; url base;

limite de concorrência; credenciais de autenticação; e as extrações para cada um dos tipos de produto (consultar Secção 4.2.2 para mais detalhes).

Listagem 4.1: Excerto da configuração da API de acesso a produtos do Copernicus Open Access Hub. Omissão de elementos representada por **(...)**.

```
"copernicus-oah-odata": {
    "description": "Copernicus Open Access Hub - OData",
    "base-url": "https://scihub.copernicus.eu/dhus/odata/v1/",
    "concurrency-limit": 2,
    "credentials": {
      "username": "<username>",
      "pwd": "<pwd>"
    },
    "extractions": [(...)]
}
```

No caso da API de catalogação, para além da configuração referida na de acesso, inclui também o tamanho das páginas da API REST (Listagem 4.2). Deste modo, quanto maior o tamanho da página, menos vezes se terá de contactar a API em caso de existir mais que uma página, no entanto, os ficheiros obtidos são maiores. Adicionalmente, nesta configuração são especificados os produtos que se pretendem suportar, registando o seu identificador, o *epoch* de recuperação histórica e a frequência com que se verifica a existência de novos produtos deste tipo.

Listagem 4.2: Excerto da configuração da API de catalogação do Copernicus Open Access Hub. Omissão de elementos representada por **(...)**.

```
"copernicus-oah-opensearch": {
    (...)
    "page-size": 10,
    "products": [
        {"program": "sentinel",
         "platform": "sentinel2",
         "product-type": [{
            "name": "S2MSI2A",
            "epoch": 2y,
            "fetching-frequency": 1d
         },(...)
    ],
    "extractions": [(...)]
(...)
```

Resumindo o fluxo de interrogação, existe uma tarefa periódica que interroga a existência de novos produtos de um determinado tipo. Se existirem, são aplicadas as extrações definidas na configuração da API de catalogação. Logo depois, através da API de acesso, descarrega-se o ficheiro de manifesto. Feito este *download*, as extrações presentes na configuração da API de acesso são aplicadas. Nestas extrações, os ficheiros sobre os quais incidem as mesmas são mapeados através do identificador no manifesto. Estando associado a este identificador um URL, descarrega-se o ficheiro e executa-se, finalmente, todas as extrações definidas para aquele contexto. Ilustrando este fluxo, no caso do Exemplo 2 da Tabela 3.1, na extração é necessário especificar o identificador da máscara de nuvens no ficheiro de manifesto. Assim, no ficheiro manifesto, associado a esse identificador está o URL para esse recurso, o que permite descarregá-lo e, finalmente, aplicar as extrações de seguida.

4.2.2 Especificação do Metamodelo

No que diz respeito ao metamodelo deste protótipo, este está representado na Fig 3.5. Sendo este constituído por um conjunto de metamodelos dos diferentes tipos de produto. Nestes estão incluídos alguns dos produtos mais populares de fornecedores americanos (*i.e.* Landsat 7 e 8, MODIS) e europeus (*i.e.* Sentinel-1, -2, -3).

Ao longo da Secção 3.4.1, foi detalhada a técnica com que é feita a especificação do metamodelo, portanto o que importa agora descrever é como essa técnica é mapeada na implementação. A ferramenta a que se recorreu para representar o metamodelo dos diferentes produtos foi JSON Schema [138], sendo que os metadados são armazenados em JSON. Desta forma, é possível anotar e especificar os elementos de metainformação de uma forma muito expressiva. Na Listagem 4.3 está representada parte de um exemplo, no qual é possível observar a tal expressividade que se falou na frase anterior.

A desvantagem do JSON Schema neste contexto passa pela ausência de mecanismos de hierarquia entre metamodelos, sendo que desta forma existe a necessidade de, para cada tipo de produto, definir extensivamente todos os seus elementos, mesmo aqueles que são herdados de níveis de abstração superiores. O que resulta em ficheiros de JSON Schema com grandes porções repetidas entre metamodelos. A verdade é que não se encontrou uma solução prática para este problema.

Adicionalmente, existe a possibilidade de especificar quais os elementos de metainformação que são indexados no MongoDB. Deste modo, quando os utilizadores consultam o metamodelo, também podem analisar quais os elementos que estão indexados, melhorando a *performance* das suas *queries*.

Listagem 4.3: Excerto da especificação do metamodelo dos produtos Landsat 8 nível 1. Omissão de elementos representada por **(...)**.

```
1 {
```

```
  (...)
  "type": "object",
  "title": "The LANDSAT_8_C1 Schema",
  "required": [
    "_id",
    (...)
  ],
  "properties": {
    "_id": {
      "id": "#/properties/_id",
      "type": "string",
      "title": "Product ID",
      "examples": [
        "LC08_L1GT_088231_20190711_20190711_01_RT"
      ],
(...)
```

Listagem 4.4: Excerto da configuração dos índices.

```
{
    "type": "partial"
    "order": "ascending"
    "fields-names": ["cloudCoverPercentage"]
}
```

Na Listagem 4.4, está ilustrada uma entrada desta especificação, dado que neste caso está representado um índice ascendente e parcial sobre a percentagem de cobertura de nuvens. De notar que este é parcial, pois nem todos os tipos de produto têm este elemento de metainformação. Relativamente à ordem (descendente/ascendente) do índice, esta fará com que o resultado da *query* seja apresentado por essa mesma ordem (sem necessidade de reordenações prévias). Por outro lado, importa referir que o campo `field-names` é um *array*, de modo a suportar índices compostos. O que é importante caso se queira otimizar, por exemplo, *queries* espaciotemporais. Já que estas são duas dimensões que são frequentemente interrogadas em conjunto. Em relação à pesquisa recorrendo a este índice, esta é realizada primeiro a nível espacial, o que irá restringir o número de produtos. No entanto, se for uma área geográfica com um elevado número de produtos, a pesquisa temporal deverá também tirar partido da indexação. Finalmente, importa referir que o impacto de cada índice deve ser estudado caso a caso, pois este está dependente das caraterísticas do conjunto de dados (*e.g.* volume, distribuição dos dados).

4.2.3 Especificação hierárquica de ETL

Como referido na Secção 3.4.3, a especificação das extrações a executar é feita de forma hierárquica. Na Listagem 4.5 é justamente esta caraterística que está ilustrada, neste caso trata-se da especificação das extrações na API de catalogação do Copernicus, sendo possível observar os níveis definidos na Fig 3.5: fornecedor - `extractions`; programa - `sentinel`; plataforma - `sentinel-1`, `sentinel-2`, `sentinel-3`; e, por fim, tipo de produto - `S2MSI1C`, `S2MSI2A`. A título de exemplo, os produtos `S2MSI1C` herdam as extrações `sentinel-2`, `sentinel` e as próprias do fornecedor. Finalmente, importa referir que a ausência de alguns níveis deve-se apenas à inexistência de extrações específicas a esses. Exemplificando, o tipo de produto `SLC` herda todas as extrações necessárias dos níveis superiores, sendo possível fazer *overwrite* das mesmas.

Listagem 4.5: Excerto da especificação das extrações da API de catalogação do Copernicus Open Access Hub. Omissão de elementos representada por **(...)**

```
"copernicus-oah-opensearch": {
    (...)
    "extractions": [],
    "sentinel": {(...)},
    "sentinel1": {(...)}
    "sentinel2": {(...)},
    "sentinel3": {(...)},
    "S2MSI1C": {(...)},
    "S2MSI2A": {(...)},
    (...)
}
```

4.2.4 Tradução de XML para JSON

Na Secção 3.4.3 foi referida a necessidade de traduzir os documentos para uma só linguagem de representação. Isto é importante de forma a unificar o pré-processamento, evitando ter de desenvolver mecanismos de extração tanto para XML como para JSON. No protótipo, a decisão acabou por recair no JSON, já que este é naturalmente integrado no MongoDB. Deste modo, os documentos XML descarregados devem ser traduzidos para JSON. No entanto, é importante que esta tradução não leve a perdas de informação. Para este efeito existem duas soluções: guardar os documentos XML originais ou usar uma biblioteca que consiga reverter a tradução a partir do JSON.

Os três desafios principais de uma tradução de XML para JSON sem perdas são os seguintes: o JSON não tem o equivalente a atributos XML; o JSON não suporta *mixed content*, como, por exemplo, misturar elementos de texto com outros; e, por fim, o JSON

não suporta espaços de nomes. Considerando estes problemas, foram encontradas várias convenções [139] que visam standardizar esta tradução de XML para JSON. Dada a quantidade de convenções, focar-se-á a análise no JSONML, pois esta é a que parece ter maiores potencialidades. Esta foi definida de forma a que não existam perdas, visto que suporta atributos XML, *mixed content* e espaços de nomes. À partida, parece uma solução satisfatória; no entanto, os ficheiros JSON resultantes são muito verbosos, tornando-se estes de difícil leitura por parte de um humano. Adicionalmente, não foram encontradas bibliotecas Java que implementem esta convenção de uma forma satisfatória.

Concluindo, optou-se pela alternativa de guardar os documentos XML originais. Ou seja, os documentos XML são traduzidos para JSON; são aplicadas as extrações definidas; e, posteriormente, armazena-se o documento original no sistema de ficheiros e insere-se o resultado da extração no MongoDB. A justificação desta decisão passa pela dificuldade em encontrar uma biblioteca que concilie a possibilidade de reverter a tradução sem perdas com a simplicidade da sua integração no desenvolvimento.

4.2.5 Mapeamento dos *endpoints* na API

Nesta secção são apresentados detalhes relativamente à implementação da API, nomeadamente, mapeando as operações apresentadas na Secção 3.5 em *endpoints* concretos. Este mapeamento está representado na Tabela 4.1.

Tabela 4.1: Tabela com o mapeamento entre as operações descritas na Secção 3.5 em *endpoints* concretos.

Recurso	Operação	Método HTTP	*Endpoint*
Produto	Obter metadados	GET	`/products/{productId}`
	Obter dados	GET	`/products/{productId}/data/{dataId}`
	Pesquisar	POST	`/products/query`
	Adicionar produto	POST	`/products`
	Atualizar metadados	PUT	`/products/{productsId}`
	Pedir obtenção de dados	PUT	`/products/{productsId}/data/{dataId}`
	Remover dados de produto	DELETE	`/products/{productsId}/data/{dataId}`
Metamodelo	Obter metamodelos	GET	`/metamodels`
	Atualizar metamodelo	PUT	`/metamodels/{metamodelId}`
	Adicionar extensão	POST	`/metamodels/{metamodelId}/custom`

Relativamente à operação de pesquisa, nesta é exposta a linguagem de interrogação do MongoDB aos utilizadores. Esta decisão deveu-se à necessidade de obter a máxima expressividade possível. Fornecendo a este SGBD uma caraterística importante para a plataforma desenvolvida, optou-se por não limitar a pesquisa ao implementar, por exemplo, o protocolo ODATA. Caso contrário, o tempo de desenvolvimento teria sido maior e existia o risco da perda de expressividade nas interrogações. A desvantagem desta abordagem é que o acesso não é standardizado (como o ODATA) e os utilizadores têm de assimilar os modelos de dados do MongoDB, juntamente com a sua linguagem de interrogação.

Ainda no contexto da pesquisa, o mapeamento poderá causar alguma apreensão. À

partida esta é semanticamente uma operação de GET. No entanto, as *queries* do MongoDB tendem a ser extensas, o que poderia causar problemas ao nível da limitação de tamanho dos URL. Atendendo a isto, optou-se por definir o *endpoint* como sendo um POST no qual a *query* é encapsulada no corpo do pedido.

Finalmente, é necessário referir que esta API foi documentada recorrendo ao Swagger [140] - que se trata de uma ferramenta de especificação de API REST. Desta forma, existe um *endpoint* adicional - `/swagger-ui.html` - no qual é possível consultar toda a documentação e, adicionalmente, testar os diferentes *endpoints*, recorrendo a *templates* de pedidos.

4.2.6 *Deploy* Automatizado

Na implementação deste protótipo foram desenvolvidos dois projetos. Um deles trata-se da implementação Java da API, recorrendo a Spring, e o outro corresponde à implementação da componente de ingestão em Akka Scala. Ambos os projetos foram configurados de modo a que as imagens Docker fossem geradas automaticamente, durante a compilação dos mesmos. Relativamente ao projeto Java, recorreu-se a um plugin [141] do Maven [142], que gera as mesmas através da especificação de um ficheiro designado Dockerfile [143]. No caso do Scala, o empacotamento é realizado recorrendo a um plugin [144] do SBT [145].

Tanto na Listagem 4.6 como na Listagem 4.7, está descrita a geração da imagem Docker, gerada a partir de uma pré-existente na qual já está instalada o Java 8. Posteriormente, é movido (com as devidas permissões) um *script* para a raiz do sistema de ficheiros da imagem, sendo este usado no `docker-compose`. E, por fim, copia-se o jar gerado pela compilação do projeto para dentro da imagem docker. Tendo estas imagens geradas localmente, existe a possibilidade de as publicar num repositório *online* como o DockerHub [146]. De referir que todo este processo de gerar o executável e publicar a imagem pode ser despoletado com apenas um comando, que no caso do Maven é através de `mvn publish` e no do SBT é pelo `sbt publish`.

Listagem 4.6: Dockerfile a partir do qual é gerada a imagem Docker da API.

```
1 FROM openjdk:8
2 WORKDIR /
3 COPY scripts/wait-for-it.sh   /wait-for-it.sh
4 RUN chmod +x /wait-for-it.sh
5 COPY target/rs-api.jar  /app.jar
```

Listagem 4.7: Especificação do Dockerfile incluída no ficheiro de *build* do SBT.

```
import com.typesafe.sbt.packager.docker._

dockerCommands := Seq(
  Cmd("FROM", "openjdk:8"),
  Cmd("WORKDIR", "/"),
  Cmd("COPY", "opt/docker/lib/wait-for-it.sh", "/wait-for-it.sh"),
  Cmd("RUN", "chmod", "+x", "/wait-for-it.sh"),
  Cmd("COPY", "opt/docker/lib/*.jar", "/app.jar")
)
```

Com as imagens já presentes no Dockerhub, é necessário definir o ficheiro que descreve a arquitetura - `docker-compose` - a partir do qual é feito o *deploy* da aplicação. Na Listagem 4.8 está apresentado parte do `docker-compose` que descreve a arquitetura presente na Fig. 4.3. Realçando a configuração dos volumes, no seguinte exemplo - `/mnt/a/data:/data` - o volume montado na máquina em que corre o Docker (*i.e.* `/mnt/a`) tem uma pasta que será montada no *container*, na seguinte localização - `/data`. Outro detalhe que importa referir é que antes de correr os executáveis dentro do *container*, é executado um *script* (`wait-for-it.sh`) que verifica se todas as componentes das quais esse *container* é dependente já estão disponíveis. Finalmente, deixar a nota de que todas as componentes da ingestão são instâncias da mesma imagem, mas com parametrizações diferentes.

Listagem 4.8: Excerto do `docker-compose` da arquitetura descrita na Fig 4.3. Omissão de elementos representada por **(...)**

```
services:
  mongo:
    (...)
    volumes:
      - /mnt/a/mongo:/data/db
    (...)
  worker:
    image: andrelopes/rs-ingestion:0.1-SNAPSHOT
    command: ["bash", "/wait-for-it.sh", "master:2551", "--",
              "bash", "/wait-for-it.sh", "mongo:27017", "--",
              "java", "-jar", "/app.jar" , '5001', '2']
    volumes:
      - /mnt/a/data:/data
    deploy:
      replicas: 1
    (...)
```

Após a definição do `docker-compose` já é possível fazer o *deploy* em qualquer máquina

que tenha o Docker instalado, sendo que se apenas se quiser executar os diferentes serviços numa só máquina, basta executar o seguinte comando: `docker-compose up`. Se por outro lado, o objetivo for correr os serviços de forma distribuída, é necessário que cada uma delas tenha o Docker instalado, de modo a criar um *swarm cluster* (Secção 4.1.3). Para se proceder ao *deploy*, executa-se no Swarm Manager o seguinte comando: `docker stack deploy -compose-file docker-compose.yml cluster_rs`. Este descarregará todas as imagens necessárias e instanciará os *containers* descritos no ficheiro `docker-compose`, atribuindo um nome à aplicação - `cluster_rs`.

Durante a execução, se for necessário atualizar alguma das componentes, basta publicar a imagem no DockerHub, parar o *container*, atualizá-lo e lançá-lo de novo. Tudo isto através de um processo automatizado. Por outro lado, se a necessidade for aumentar o número de instâncias de Workers, também é muito simples. Ainda no contexto da manutenção, se o problema for os recursos dos Swarm Nodes, é possível acrescentar novos nós sem interrupção do sistema, comunicando com o Swam Manager. No entanto, pode existir a necessidade de adicionar mais algumas configurações à máquina, nomeadamente, expondo portas necessárias à comunicação entre a aplicação e o exterior.

4.3 Conclusão

Ao longo de todo o processo de desenvolvimento deste protótipo, muitas foram as dificuldades de integração das múltiplas fontes de dados. O desafio passou por deixar o sistema genérico o suficiente, sem que não explodisse em complexidade. Outro dado que importa referir é o facto de nas decisões de arquitetura se ter optado pelas tecnologias que davam melhores garantias. Cobrindo desta forma todas as preocupações de escalabilidade e manutenção.

No entanto, esta decisão nem sempre facilitou o desenvolvimento, pois o tempo é limitado para compreender um ecossistema de tecnologias tão diverso. Consequentemente, abdicou-se de algumas funcionalidades, tendo-se sempre como critério de seleção o facto de serem ou não vitais para a prova de conceito. Após o desenvolvimento, poder-se-ia realizar mais algumas iterações sobre este protótipo, servindo este de uma boa base para a construção de um produto.

Finalmente, notar que este protótipo permite comprovar que a solução desenhada é exequível. Sendo que o próximo capítulo (Capítulo 5) visa validar a plataforma proposta e o protótipo, reunindo um conjunto de casos de estudo.

Capítulo 5

Avaliação

Neste capítulo é descrito o processo de avaliação da solução proposta, com especial foco no subconjunto de funcionalidades selecionado para o protótipo, começando pela Secção 5.1, é nesta que são apresentadas as metodologias de avaliação que permitirão validar esta solução.

Neste processo de validação importa apresentar o contexto experimental sobre o qual foi concretizada esta avaliação. Esta descrição é feita na Secção 5.2, na qual se aborda a integração de um conjunto de dissertações elaboradas em paralelo, cujo tema gravita à volta da monitoração de recursos naturais.

Na Secção 5.3, como referido nas metodologias de avaliação (Secção 5.1), será apresentado um conjunto de casos de estudo que servirá de base de sustentação a esta avaliação. Sendo estes casos de estudo reais de dissertações nesta área, começar-se-á por descrever sumariamente cada um dos mesmos, de seguida, apresentar-se-á o processo de desenvolvimento sem recorrer ao repositório desenvolvido e, por fim, será pormenorizada a integração do processo de desenvolvimento na plataforma descrita no Capítulo 3, comparando com os métodos alternativos. Nestes casos de estudo estão incluídos os seguintes temas: Deteção Remota de Parcelas Agrícolas (Secção 5.3.1); Deteção Remota de Estruturas Permanentes (Secção 5.3.2); Deteção Remota do Estado da Vegetação em Faixas de Gestão de Combustível de Incêndios (Secção 5.3.3); Fusão de Imagens de Satélite (Secção 5.3.4).

Por fim, na Secção 5.4, é apresentada uma reflexão relativa aos diferentes pontos de destaque de cada um dos casos de estudo, apresentando-se desta forma uma discussão global da qualidade da solução apresentada na Secção 3.

5.1 Metodologia de Avaliação

No início deste documento, no Capítulo 1.2, foi formulado o problema. Consequentemente, no Capítulo 3, foi apresentada a abordagem para a resolução do mesmo. Finalmente, ao longo deste capítulo, será apresentada a validação da solução retratada. Para este efeito, será traçada uma sistematização da avaliação.

De assinalar que a avaliação incidirá com maior foco sobre o subconjunto de funcionalidades implementadas no protótipo. No entanto, estas são suficientemente representativas da solução para que seja possível generalizar a avaliação. Ainda assim, se o cenário o justificar, estudar-se-á o impacto de uma certa funcionalidade, mesmo não estando esta implementada no protótipo.

Relativamente ao método de avaliação em concreto, este passará por analisar um conjunto de casos de estudo. Nesta análise será confrontado o fluxo de desenvolvimento usado com a alternativa que a plataforma desenvolvida representa. O critério de seleção destes casos de estudo passa por encontrar cenários que, apesar de terem um núcleo comum, apresentam as suas próprias especificidades. Pretende-se que estes sejam significativos o suficiente, de modo a que se possa argumentar a generalização para outras circunstâncias.

Adicionalmente, os casos de estudo estão integrados num contexto real. Esta é uma caraterística importante, pois a avaliação num ambiente simulado não teria em conta as necessidades específicas das partes interessadas nesta plataforma.

Na análise dos casos de estudo, seguir-se-á uma sequência de passos comum. Numa fase inicial, será descrito o caso de estudo. Esta descrição inclui um resumo: da motivação para o mesmo; do problema; da abordagem para o resolver; e dos resultados obtidos. Posteriormente, tendo em conta a abordagem, será detalhado o fluxo de desenvolvimento sem recorrer à solução proposta nesta dissertação, aqui estão incluídos detalhes relativos: à pesquisa de produtos; ao acesso aos mesmos; à computação dos novos produtos de observação terrestre; e, finalmente, à disponibilização dos mesmos, analisando os metamodelos e os formatos que os representam. De notar que, acompanhado da descrição destes passos, estão incluídos os desafios enfrentados por parte dos autores.

De forma a obter todas as nuances relativas ao ao trabalho realizado em cada um dos casos de estudo, os autores foram entrevistados. Durante este processo, as questões seguiram os passos referidos no parágrafo anterior. Adicionalmente, cada um dos entrevistados expôs o seu *feedback* relativamente a esta solução, tendo este sido incorporado na discussão dos resultados.

Após apresentar o fluxo de desenvolvimento a que se recorreu, define-se o fluxo alternativo caso se utilizasse a plataforma proposta, sendo que esta descrição está acompanhada de uma visão crítica relativa às vantagens e desvantagens desta alternativa. Por outro lado, em alguns casos, poderá também ser importante estudar a incorporação deste fluxo em outras plataformas (*e.g.* GEE).

Para finalizar, após apresentar e discutir os vários casos de estudo, são expostas as

conclusões obtidas através da instanciação desta metodologia até aqui descrita. Estas conclusões refletem uma visão geral sobre a avaliação desta plataforma, tendo como base a solução que representa o estado da arte - GEE.

5.2 Contexto Experimental

Na Secção 5.1 foi referido que os casos de estudo selecionados representam aplicações reais. De facto, isto apenas se tornou possível dado o contexto no qual foi elaborada esta dissertação. Este projeto está integrado num conjunto de dissertações resultantes de uma iniciativa relacionada com sistemas inteligentes aplicados à **MO**nitoração de **RE**cursos **NA**turais, designada MORENA.

Nesta iniciativa, os temas das diferentes dissertações subdividem-se em: deteção e monitorização remota e *in-situ* de recursos naturais; e monitorização virtual de recursos naturais. De notar que a presente dissertação está incluída no primeiro. Adicionalmente, nesta mesma área de trabalho foram desenvolvidas dissertações como: monitoração de faixas de gestão de combustível de incêndios; deteção remota de estruturas permanentes; deteção semiautomática de áreas de vegetação permanente; fusão de imagens e produtos de satélite; ou deteção automática de parcelas agrícolas.

Ao longo da elaboração destas dissertações, dada a presença de elementos transversais a todas elas, a cooperação foi notória. Para além da entreajuda diária, ocasionalmente eram organizados *workshops*, com o intuito de: incentivar a aprendizagem nas diferentes fases; partilhar conhecimento, recursos, métodos e técnicas; e, por fim, desenvolver uma equipa com competências diversas. Nestes *workshops*, para além dos autores das dissertações e respetivos orientadores, também estavam representadas entidades externas, como por exemplo, o IPMA ou o Centro de Informação Geoespacial do Exército (CIGeoE).

Neste contexto, a plataforma desenvolvida tem uma clara integração transversal a todas as dissertações englobadas na deteção e monitorização remota e *in-situ* de recursos naturais. A verdade é que qualquer uma destas produz novos dados de observação terrestre. O objetivo é que todos os produtos computados sejam armazenados e disponibilizados recorrendo à plataforma desenvolvida nesta dissertação.

Nesta fase de descrição do contexto em que foi desenvolvida esta dissertação, ficam claras as potencialidades de integrar outras dissertações na avaliação. Tal como descrito na Secção 5.1, a validação da solução consiste na análise de um conjunto de casos de estudo. Portanto, estes incidem sobre trabalho realizado em paralelo. Para este efeito, entrevistou-se os autores das dissertações selecionadas como caso de estudo, sendo que alguns detalhes foram assimilados através do acompanhamento constante.

Concluindo, os casos de estudo selecionados foram:

1. **Deteção Remota de Parcelas Agrícolas** (Secção 5.3.1);

2. **Deteção Remota de Estruturas Permanentes** (Secção 5.3.2);

3. **Avaliação por Deteção Remota do Estado da Vegetação em Faixas de Gestão de Combustível de Incêndios** (Secção 5.3.3);

4. **Estudo Sobre o Efeito de. Estruturas Permanentes na Fusão de Imagens de Satélite** (Secção 5.3.4)

Todos apresentam um fluxo de desenvolvimento com especificidades próprias. Nomeadamente, os métodos usados são distintos, os produtos gerados têm diferentes metamodelos e formatos, e as necessidades de poder de computação são variáveis. Estes foram os principais fatores para a seleção deste conjunto de casos de estudo.

Importa deixar a nota de que existiam outros possíveis casos de estudo neste contexto. No entanto, considerou-se que estes seriam redundantes nesta avaliação. A título de exemplo, um caso de estudo que poderia ter sido selecionado seria a deteção semiautomática de áreas verdes permanentes. Porém, existia uma certa sobreposição do fluxo de desenvolvimento relativamente ao caso de estudo retratado na Secção 5.3.2.

O objetivo final é que os produtos resultantes destes casos de estudo, no final das respetivas dissertações, sejam disponibilizados no protótipo da plataforma.

5.3 Casos de Estudo

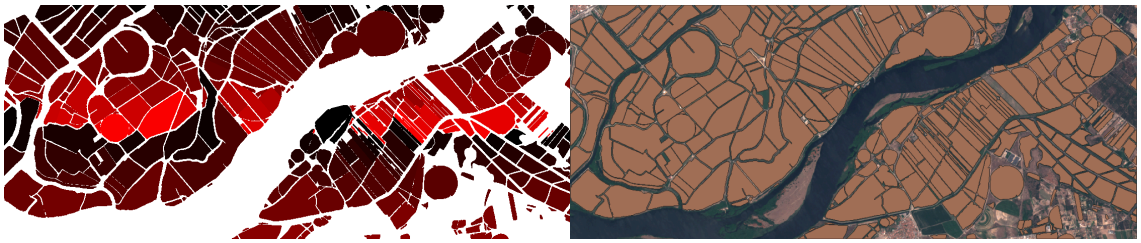
5.3.1 Deteção Remota de Parcelas Agrícolas

Segundo o autor da dissertação [147], correspondente a este caso de estudo, a identificação e monitorização de parcelas agrícolas oferece vantagens tanto ao nível do agricultor como a nível administrativo. No que diz respeito ao agricultor, este poderá monitorizar as suas culturas de forma a otimizar o rendimento das suas próprias colheitas. E, numa perspetiva administrativa, torna-se possível supervisionar os parcelários de maior dimensão destinados, por exemplo, à produção em massa de alimentos. Ambas as atividades necessitam de um acompanhamento periódico, sendo impraticável fazê-lo presencialmente, surgindo assim a deteção remota como resposta.

De realçar que a dissertação em questão não tem como objetivo a monitorização das culturas, mas sim a delimitação das parcelas agrícolas. Neste contexto, recorrendo a imagens obtidas por satélites de média/alta resolução, procurou-se definir os limites reais das parcelas e não os administrativos. A verdade é que estes últimos já se encontram bem identificados pelas entidades responsáveis. Porém, dentro dos limites administrativos, o agricultor pode ter várias subparcelas, demarcando estas os tais limites reais.

Relativamente à abordagem para a resolução deste problema, recorreu-se a séries temporais de produtos Sentinel-2 para a delineação dos limites reais de parcelas agrícolas no solo, baseada em segmentação de imagem sobre índices de vegetação e imagem de cor verdadeira. Tendo-se comparado os resultados obtidos com verdades de terreno disponibilizadas por entidades oficiais ou demarcadas manualmente no terreno via GPS.

No que diz respeito aos resultados obtidos, na Fig 5.1 estão representadas as parcelas agrícolas detetadas na região de Salvaterra de Magos. Esta foi uma de múltiplas regiões estudadas, sendo que, para a região ilustrada, a exatidão do produtor foi de cerca de 80% e a do utilizador rondou os 90%. A primeira exatidão representa quão bem os polígonos da verdade de terreno estão incluídos nos polígonos produzidos pela abordagem. Já a segunda exatidão representa quão bem os polígonos produzidos pela abordagem seguida estão incluídos nos polígonos da verdade de terreno.



(a) Sem fundo. (b) Com fundo.

Figura 5.1: Parcelas agrícolas detetadas na região de Salvaterra de Magos. Imagens fornecidas pelo autor da dissertação.

5.3.1.1 Fluxo de Desenvolvimento

Acesso aos produtos. Durante o desenvolvimento da solução foi necessário aceder a um conjunto de produtos de observação terrestre, estando incluídos:

- Sentinel-2 nível 1C;
- Carta de ocupação de solos (COS) de 2015;
- Parcelário agrícola de 2018.

Relativamente ao primeiro, foi necessário aceder à série temporal de 2018 dos produtos Sentinel-2 de nível 1C (sem correções atmosféricas). No que diz respeito às verdades de terreno, estas baseiam-se nos restantes produtos referidos: o COS de 2015, que é disponibilizado pela DGT; e o parcelário agrícola de 2018, que é fornecido pelo Instituto de Financiamento da Agricultura e Pescas (IFAP).

Passando à fase de acesso a cada um destes produtos, as interfaces disponibilizadas são distintas. Começando pelos produtos Sentinel-2, estes foram acedidos, inicialmente, via interface gráfica do *hub* do Copernicus (Secção 2.4.1). Desta forma, acedia-se a cada um dos produtos da série temporal e obtinha-se as bandas necessárias, tendo em conta uma cobertura máxima de nuvens. Rapidamente se apercebeu que esta abordagem era impraticável, tanto pela quantidade de produtos como pelas limitações de concorrência e latência da API. Consequentemente, optou-se por automatizar este acesso através de uma biblioteca Python denominada sentinelsat [148], não existindo nesta a possibilidade de descarregar algumas bandas do produto, o que representa um desperdício de recursos, já

que apenas algumas destas serão processadas. Adicionalmente, a partir de cada um dos produtos, foi necessário computar imagens que representam índices derivados.

Outro dado importante é o facto de que, apesar de se descarregar a cena toda, apenas se estava a processar uma porção da mesma. Uma vez que quando se filtrava pela cobertura de nuvens, esta metainformação era ao nível da cena e não da região em estudo. De facto, isto poderia enviesar os resultados. De modo a garantir a qualidade dos dados, optou-se por não filtrar pela dimensão de cobertura de nuvens, ou seja, descarregou-se todos os produtos da série temporal. De seguida, tendo os produtos armazenados localmente, recorreu-se às máscaras de nuvens disponibilizadas no produto, de forma a filtrar as cenas que na região de estudo tivessem uma reduzida cobertura de nuvens.

Movendo o foco para o COS 2015, este foi acedido através dos serviços Web Map Service (WMS) disponibilizados nas entidades responsáveis por este produto. Posteriormente, importou-se o serviço no QGIS e exportou-se o *raster* correspondente à região da Lezíria do Tejo, não se conseguindo obter o serviço Web Feature Service (WFS), o que permitiria exportá-lo em formato vetorial. Contrariamente ao caso do parcelário agrícola de 2018, em que o acesso foi semelhante mas através do serviço WFS.

Computação. Após ter todos estes dados armazenados localmente, procedeu-se à computação dos limites das parcelas agrícolas. Neste caso de estudo, para a região da Lezíria do Tejo, o tempo de processamento está na ordem das dezenas de segundos. Para áreas maiores, este será naturalmente maior, não existindo de momento uma versão paralela do algoritmo.

Armazenamento. O produto gerado está num formato vetorial (.shp), no qual cada polígono corresponde a uma parcela agrícola. Paralelamente, cada um destes foi enriquecido com metainformação (*e.g.* identificador único, georreferenciação, área, perímetro). De referir que todos os produtos gerados correspondentes a diferentes regiões de estudo foram armazenados localmente, não existindo um acesso sistemático aos mesmos.

5.3.1.2 Fluxo Alternativo

Acesso aos produtos. Começando pelo acesso aos produtos, os principais problemas da abordagem usada são: ter de se recorrer a múltiplas fontes e existir a necessidade de armazenar um grande volume de dados localmente. O primeiro problema resolve-se através da existência de um repositório comum, no qual podem ser desenvolvidos novos adaptadores para as diferentes fontes. Por outro lado, de modo a resolver o segundo problema referido, este repositório foi construído de forma a que todos os dados descarregados sejam partilhados entre todos os seus utilizadores. Evitando que os múltiplos elementos de uma equipa tenham de descarregar (e pré-processar, em alguns casos) cada um destes produtos.

Relativamente à ingestão de dados do Sentinel-2, esta poderia tanto ser através de uma configuração *a priori* ou recorrendo ao mecanismo de *prefetch*. Através da primeira opção, poder-se-ia configurar o *download* automático de todos os produtos de Sentinel-2 nível 1C correspondentes à região de estudo. Por outro lado, caso essa configuração não estivesse presente, poder-se-ia explorar o mecanismo de *prefetch*, através do qual o utilizador requisitaria a ingestão dos produtos no repositório. Adicionalmente, o fornecedor a que se recorreu não fornece alguns dos índices derivados necessários para o algoritmo. Desta forma, a geração destes é responsabilidade dos utilizadores que necessitem dos mesmos.

Ainda no contexto da pesquisa de produtos do Sentinel-2, importa notar que neste repositório não é possível filtrar pela percentagem de nuvens de uma região específica incluída numa cena. Este é um mecanismo que de facto aumentaria muito a expressividade deste sistema. A verdade é que esta funcionalidade, apesar de não estar implementada no protótipo, poderia ser facilmente integrada na arquitetura definida, como discutido na secção relativa ao trabalho futuro (Secção 6.1).

No que diz respeito ao COS e ao parcelário agrícola, existem duas opções: desenvolver um adaptador para serviços WMS ou WFS, ou deixar que um dos utilizadores fique responsável por submeter estes produtos. A primeira é mais sistemática, pois automatizaria todo o processo de extração das imagens através dos serviços WMS ou WFS. No entanto, o custo desta opção seria muito maior que o da segunda. Pois, na segunda opção, o utilizador ficaria responsável por: fazer esta extração; definir um metamodelo; e, de seguida, submeter o produto. Deste modo, se se pretender utilizar o COS de 2015 ou o parcelário agrícola de 2018, poder-se-á tomar a iniciativa de os integrar no repositório, contribuindo para uma gestão colaborativa do catálogo.

Computação. Após esta fase de identificar e aceder aos produtos, passa-se à computação. Como referido na Secção 5.3.1.1, não existe uma versão paralela do algoritmo. Portanto, o facto do processamento ser realizado numa infraestrutura com maior poder computacional (*e.g.* recorrendo a GPU) não trará grandes benefícios a nível da *performance*. Ainda assim, como os dados estão presentes na infraestrutura, é possível tirar partido da localidade dos dados. Caso contrário é necessário fazer o *download* dos produtos para a máquina pessoal; de seguida, computar o novo produto; e, por fim, ainda ser preciso fazer o *upload* do mesmo para a plataforma. Dado que estes produtos têm um volume significativo, esta opção passa por ser um desperdício de recursos, sendo bastante mais eficiente integrar todo o fluxo de desenvolvimento na plataforma proposta.

Ainda no contexto da computação, neste caso de estudo e nos descritos nas Secções 5.3.2 e 5.3.3, recorreu-se bastante ao QGIS para realizar algumas análises *ad hoc*. Por este motivo, é importante que esta ferramenta seja instalada nas máquinas presentes na área de computação. No entanto, apesar de na arquitetura (Secção 3.2) se ter discutido esta necessidade, no protótipo tal não foi instalado.

Armazenamento. Neste fluxo alternativo, o produto gerado é armazenado e disponibilizado de uma forma sistemática. Para além dos metadados embebidos no ficheiro da imagem, importa definir um metamodelo que permita pesquisar este produto sem aceder a este ficheiro. Para este efeito, estende-se a raiz da hierarquia ilustrada na Fig. 3.5 que reúne os elementos de metainformação transversais a qualquer produto: **id** - `PA-2015-MORENA-LEZIRIATEJO`; **fornecedor** - `DI@FCT-UNL`; **programa** - `MORENA`; **plataforma** - <nome dos autores do algoritmo>; **tipo de produto** - `PARCELAS-AGRICOLAS-MORENA` ; **título** - Parcelas Agrícolas da região da Lezíria do Tejo de 2015 computadas no contexto do MORENA; **cobertura geográfica** - <GeoJSON da área de estudo>; **data de início/fim da aquisição** - <início/fim da série temporal usada na computação>. Nesta enumeração apenas se encontram os principais elementos de metainformação, mas é possível perceber que alguns elementos (*e.g.* plataforma) não se mapeiam perfeitamente neste produto, devido à necessidade de os elementos serem genéricos.

Passando à metainformação específica deste produto, esta divide-se em duas vertentes: proveniência dos dados e agregados. A primeira permite certificar a origem dos dados, estando nesta elementos como: a **matriz de confusão**; a **exatidão do produtor e utilizador**; os **produtos processados**, tendo em conta que no caso do Sentinel-2 indica-se o identificador dos produtos e as bandas e índices a que se recorreu; e, por fim, o **URL para o repositório de código**, no qual é possível obter os *scripts* para a geração deste tipo de produtos. Passando à segunda vertente, os agregados possibilitam um aumento de expressividade nas interrogações, estando incluídos: a **área média/máxima/mínima das parcelas**; o **perímetro médio/máximo/mínimo das parcelas**; o **número de parcelas identificadas**; e a **percentagem de píxeis nos quais estão representadas parcelas agrícolas**.

Após a definição do metamodelo do novo produto, descreve-se este em JSON Schema e submete-se o mesmo através da API. De realçar que esta tarefa apenas é necessária na primeira vez que um produto deste tipo é inserido. Estando o metamodelo registado, é possível agora submeter os produtos gerados, o que inclui enviar um *zip* com o ficheiro vetorial e um JSON com os metadados, cujo esquema será validado através do JSON Schema. Uma grande vantagem desta abordagem é o facto da geração do produto não ficar dependente apenas dos autores da dissertação correspondente a este caso de estudo. A partir do momento em que está registado o metamodelo, qualquer utilizador poderá adaptar os *scripts* disponibilizados, de modo a gerar os produtos que identificam as parcelas agrícolas de outra região.

Enriquecimento do metamodelo. Nos catálogos atuais, o metamodelo dos produtos é relativamente estático. Ou seja, a partir do momento em que cada um dos produtos é submetido, não há praticamente qualquer alteração sobre os seus metadados. No entanto, nesta plataforma é possível que qualquer utilizador registe as suas próprias extensões ao metamodelo. Depois de registadas, qualquer utilizador pode inserir metadados que sigam o esquema definido pelo autor da extensão.

Como foi referido, este caso de estudo foca-se na identificação das parcelas agrícolas e não na monitorização das mesmas. No entanto, após a submissão deste produto, outro utilizador poderia enriquecer a metainformação deste ao nível da monitorização. Como por exemplo, ao classificar as culturas de cada uma das parcelas, tendo associada uma data à qual corresponde essa classificação. Noutra perspetiva, se este produto fosse gerado para áreas correspondentes às das cenas do Sentinel-2, seria possível, por exemplo, pesquisar produtos do Sentinel-2 pela área média das parcelas. Finalizando esta análise, durante a seleção dos produtos Sentinel-2, é calculada a percentagem de nuvens para as regiões de estudo, incluídas na cena. De facto, poder-se-ia enriquecer os metadados destes produtos indexando a percentagem de nuvens associada a uma área específica. Esta solução seria provisória, pois a resolução sistemática deste problema foi remetida para trabalho futuro (Secção 6.1).

5.3.2 Deteção Remota de Estruturas Permanentes

O presente caso de estudo [149] está englobado na área da classificação de estruturas permanentes. Esta tem importância, nomeadamente, ao nível do ordenamento do território, de gestão de desastres ou até de alinhamento de imagens. Estas estruturas permanentes representam qualquer estrutura que seja constituída por materiais artificiais e que não tenham sido movidas desde a sua criação, o que engloba, por exemplo, estradas, aglomerados populacionais ou casas individuais.

O problema atual dos produtos de classificação de estruturas permanentes passa pela baixa resolução temporal e espacial para determinadas aplicações. Em termos de resolução temporal, esta é na escala de anos, porém, as mudanças ao longo desse intervalo podem ser várias, dado o rápido desenvolvimento de certas áreas urbanas. Por outro lado, a resolução espacial é reduzida, não sendo adequada para a identificação de pequenas estruturas, como por exemplo pequenos aglomerados ou estradas. Após esta descrição torna-se claro que é necessária a automatização na geração desta classificação, recorrendo a dados com uma resolução espacial e temporal satisfatória.

A abordagem definida para a resolução deste problema passou por explorar dados de séries temporais do Sentinel-1 e -2, com resolução espacial de 10 por 10 metros, recorrendo a algoritmos de aprendizagem automática. Estes foram comparados, de modo a identificar o que melhor se adequa à resolução deste problema.

Quanto aos resultados obtidos, na Fig 5.2 está ilustrada uma comparação entre o classificador XGBoost com a carta Global Human Settlement Layer [150] (GHSL) de 2015. A previsão apresenta maior detalhe tanto a nível de pequenos aglomerados e estruturas isoladas como a nível de corpos de água estreitos. Ao comparar o COS 2015, sem informações relativas a estradas, a GHSL obtém um Kappa de 46% e precisão de 65%. Já a previsão, apresenta 82% em ambas as métricas, o que representa uma melhoria significativa.

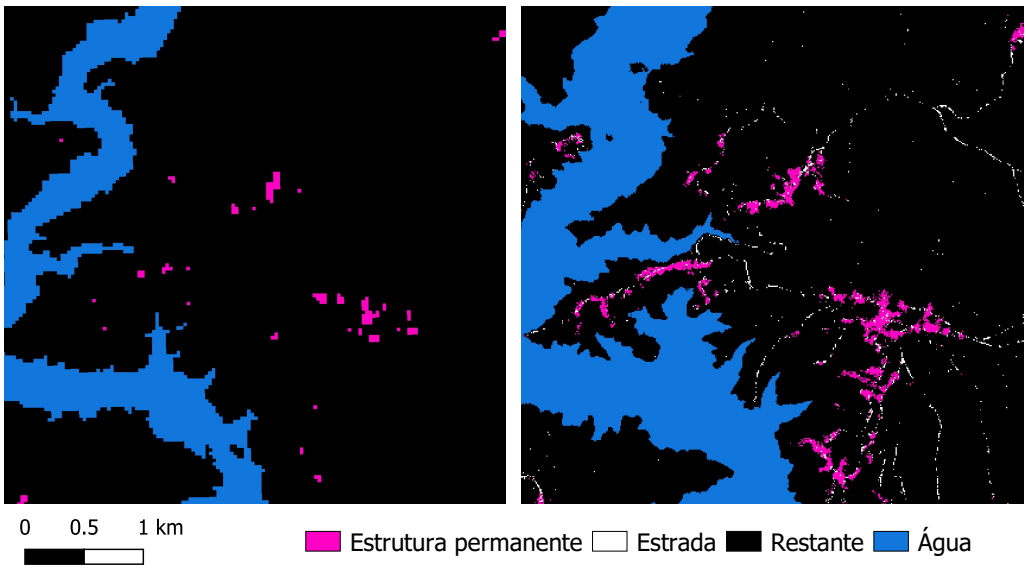


Figura 5.2: Comparação entre GHSL de 2015, à esquerda; e resultado da classificação XGBoost, à direita. Imagem fornecida pelo autor da dissertação.

5.3.2.1 Fluxo de Desenvolvimento

Acesso aos produtos. Durante o desenvolvimento da solução foi necessário aceder a um conjunto de produtos de observação terrestre, estando incluídos os seguintes:

- Sentinel-2 nível 1C;
- Sentinel-1 GRD;
- COS de 2015;
- Open Street Maps - informação relativa às estradas.

Em relação ao primeiro, foi descarregada a série temporal de 2016. O acesso a este tipo de produtos foi semelhante ao do caso de estudo representado na Secção 5.3.1. Sendo que se descarregaram os produtos com uma percentagem de nuvens até 30% em toda a cena. De seguida, removeu-se manualmente as que foram identificadas como estando contaminadas na região de estudo (inserida na cena). O que no final resultou em 18 produtos com 13 bandas cada, totalizando 252 imagens. A partir destes dados ainda foram derivados uma série de índices.

No que diz respeito ao Sentinel-1, a série temporal de 2016 foi recolhida recorrendo ao portal de dados da Alaska Satellite Facility [151]. Optou-se por este portal, pois no Copernicus Open Access Hub os produtos Sentinel-1 de 2016 estão *offline*. Apenas é possível pedir a reconstrução de dois produtos em simultâneo, o que é um problema tendo em conta que é necessário pelo menos um produto em cada mês de 2016. Ao recorrer à plataforma de dados referida, estes produtos estão *online*, e o acesso aos mesmos foi efetuado manualmente via interface gráfica. Adicionalmente, foi necessário pré-processar os produtos Sentinel-1 obtidos, de maneira a aplicar algumas correções.

Os restantes produtos referidos foram utilizados como verdades de terreno. A necessidade de obter a informação referente às estradas do Open Street Maps passa pela ausência de grande parte das estradas no COS. De referir que esta informação é fornecida num formato vetorial. Por outro lado, tal como no caso de estudo da Secção 5.3.1, não estando disponível o serviço WFS para o COS, procedeu-se à extração do *raster* a partir do serviço WMS.

Computação. No que diz respeito à computação, se não se tiver em conta todo o pré-processamento das imagens, o treino dos algoritmos não é especialmente pesado (na ordem dos minutos). Como se trabalhou sempre com as mesmas séries temporais, este pré-processamento apenas foi efetuado uma vez.

Adicionalmente, importa referir que se recorreu ao GEE para realizar algumas análises e visualizações *ad hoc*. Como por exemplo, para analisar variações do NDVI, visualizar o comportamento das séries temporais, ou até navegar no CORINE Land Cover de uma forma mais interativa.

Armazenamento. Finalmente, o produto gerado é armazenado tanto através de um *raster* georreferenciado (GeoTIFF) como em formato vetorial (.shp). De realçar que no caso do ficheiro vetorial para cada polígono existe metainformação relativa à classe de estrutura permanente a que este pertence. Tal como no caso de estudo anterior, os produtos gerados para as diferentes regiões de interesse encontram-se armazenados localmente.

5.3.2.2 Fluxo Alternativo

Acesso aos produtos. No que se refere ao acesso a produtos do Sentinel-2, este é muito semelhante ao do caso de estudo anterior. Por outro lado, relativamente ao Sentinel-1, é interessante refletir sobre a necessidade de recorrer a um catálogo que tenha os produtos de 2016 *online*. Nesta plataforma, a solução para este problema passaria por implementar um adaptador para a API de encomenda do Copernicus Open Access Hub. A grande desvantagem desta opção passa por se ter de esperar por uma notificação para que se possa descarregar os produtos. Alternativamente, poder-se-ia desenvolver um adaptador para a API que suporta o catálogo Alaska Satellite Facility. Em qualquer uma das opções, os mecanismos de ingestão destes produtos são opacos ao utilizador, uma vez que este, quando solicita o *download* de um produto, não necessita de se preocupar com a origem do mesmo.

No que toca ao COS de 2015, existe uma partilha deste produto com o caso de estudo anterior. Desta forma, apenas um dos utilizadores terá de se preocupar com o pré-processamento e submissão desse produto na plataforma. Paralelamente, o produto do Open Street Maps poderia seguir o mesmo mecanismo, já que o caso de estudo descrito na Secção 5.3.4 também necessita deste produto no seu processamento.

Computação. Neste caso de estudo recorreu-se a implementações disponibilizadas dos diferentes algoritmos de aprendizagem automática. Não existindo na biblioteca usada versões paralelas. Desta forma, as principais vantagens relativamente a realizar a computação nesta plataforma passam pelo acesso facilitado aos dados e localidade dos mesmos.

Alternativamente, neste caso de estudo poder-se-ia ter recorrido ao GEE para testar alguns dos algoritmos de aprendizagem automática, dado que a plataforma referida já fornece implementações distribuídas dos mesmos. Deste modo, era possível tanto agilizar o acesso aos dados como tirar partido da distribuição dos mesmos.

Armazenamento. Nesta componente, as semelhanças com o caso de estudo da Secção 5.3.1 são assinaláveis. Por este motivo, importa focar nas especificidades do metamodelo do produto gerado. No que diz respeito à proveniência dos dados, muitas são as métricas geradas no treino dos algoritmos de aprendizagem automática que produzem estes produtos de observação terrestre. Nomeadamente, o **Kappa**, a **precisão**, o ***recall***, o **F1-score**, entre outros. Estas medidas podem ser obtidas tanto a nível global, como a nível de cada uma das classes. Adicionalmente, também poderá ser incluída a **matriz de confusão**.

Passando aos metadados relativos a cada polígono, para além da classe a que pertencem, poder-se-ia incluir a **área** e **perímetro** do mesmo. Visto que a área está numa medida absoluta (*e.g.* m^2), também seria interessante armazenar a percentagem de área daquela estrutura relativamente à área total da imagem.

Finalmente, em relação aos agregados, estariam presentes, para cada classe de estrutura permanente, os seguintes metadados: a **área média/máxima/mínima**; o **perímetro médio/máximo/mínimo**; o **número de polígonos daquela classe**; a **percentagem de píxeis daquela classe em toda a imagem**; e a **percentagem de píxeis relativa apenas às estruturas permanentes**, ou seja, tendo em conta apenas os píxeis nos quais foram identificados estas estruturas. Adicionalmente, também se poderia incluir um **aglomerado global** correspondente à área, perímetro e percentagem de píxeis respeitantes a estruturas permanentes. De modo a ilustrar a expressividade que este metamodelo oferece, caso este produto fosse computado para toda a área do planeta, poder-se-ia identificar as regiões nas quais existe maior densidade de estradas, sem que fosse necessário aceder aos dados, ou seja, recorrendo apenas à metainformação fornecida.

Enriquecimento do metamodelo. Contrariamente ao caso de estudo anterior, neste não se encontrou um exemplo claro de extensão ao metamodelo apresentado. No entanto, os metadados gerados neste produto podem perfeitamente ser integrados no metamodelo de produtos como o Sentinel-1 e -2, nomeadamente, indicando a percentagem de píxeis de uma determinada classe de estrutura permanente. Ao indexar esta informação, poder-se-ia pesquisar por cada um destes atributos, o que facilitaria, por exemplo, na identificação de cenas com grandes aglomerados urbanos.

5.3.3 Deteção Remota do Estado da Vegetação em Faixas de Gestão de Combustível de Incêndios

Esta dissertação [152] surge no contexto dos incêndios florestais, os quais representam uma ameaça para o ecossistema, bens e populações. Com o objetivo de minimizar os danos causados por estes, foi legalmente estipulada a criação de Faixas de Gestão de Combustível de Incêndios (FGCI) a nível nacional, distrital e municipal. Estas são faixas florestais nas quais as árvores e a vegetação curta foram desbastadas ou até completamente removidas, criando desta forma condições adversas à propagação do fogo. Naturalmente, todos os anos as FGCI devem ser intervencionadas antes da época de incêndios, dado que estas são altamente influenciadas por fatores climáticos. No entanto, dada a extensão das mesmas, a monitorização presencial representa um enorme esforço financeiro e de coordenação entre as entidades responsáveis.

Partindo desta motivação, o problema que se pretende resolver passa por monitorizar o estado da vegetação em FGCI, recorrendo a dados de observação terrestre cuja resolução seja suficientemente fina para que seja possível detetar mudanças na vegetação.

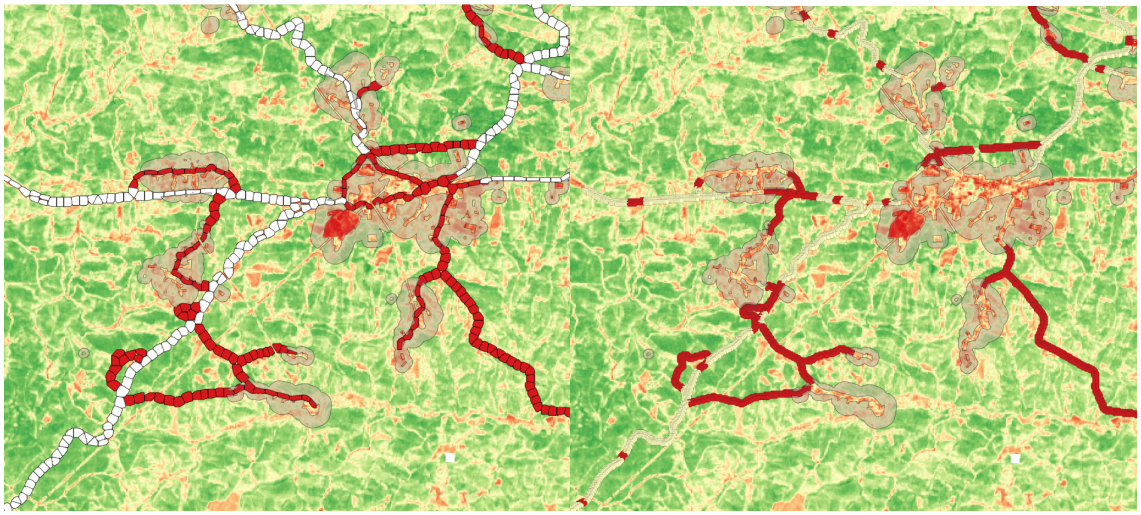
A abordagem usada para a resolução deste problema passou por recorrer a algoritmos de aprendizagem automática de maneira a identificar se uma faixa foi ou não intervencionada. Os dados usados nestes algoritmos incluem: índices de vegetação e bandas específicas de imagens de satélite; e, por outro lado, informação vetorial relativa ao mapeamento destas FGCI.

Aplicando esta abordagem, através do uso de séries temporais de imagens do Sentinel-1 e Sentinel-2, foram analisadas faixas ao longo de linhas de estradas e de linhas elétricas. Os melhores resultados foram obtidos recorrendo ao algoritmo Random Forest que, no caso das faixas ao longo das estradas (representado na Fig. 5.3), obteve um F1-score global de 96% e um Kappa de 92%. Nas linhas elétricas o melhor algoritmo foi também o Random Forest, onde os valores globais foram mais reduzidos (F1-score 87% e kappa de 73%). De referir que estes resultados correspondem apenas às estradas e linhas elétricas porque foram as únicas faixas intervencionadas no ano de 2018 na região de estudo - Mação.

5.3.3.1 Fluxo de Desenvolvimento

Durante o desenvolvimento da solução foi necessário aceder a um conjunto de produtos de observação terrestre, estando incluídos os seguintes:

- Sentinel-2 nível 2A;
- Sentinel-1;
- FGCI da região de Mação;
- FGCI intervencionadas da região de Mação.



(a) Verdade de terreno. (b) Resultado.

Figura 5.3: Análise temporal do estado das FGCI correspondentes a estradas na região de Mação, recorrendo ao algoritmo Random Forest. Vermelho - intervencionadas; branco - não intervencionadas. Imagens fornecidas pelo autor da dissertação.

Acesso aos produtos. Relativamente aos produtos Sentinel-2 nível 2A, procedeu-se à pesquisa de uma série temporal de 2018. Para este efeito acedeu-se à interface gráfica do *hub* do Copernicus. Quando foram identificados os 20 produtos necessários, procedeu-se então ao *download* de cada um deles. De realçar que todo este acesso não está automatizado, pois o acesso é via interface gráfica, existindo a limitação de dois *downloads* concorrentes nesta API. Após descarregar todos os produtos e computar localmente alguns índices necessários, no total armazenou-se localmente 90 *gigabytes*.

No caso do Sentinel-1, pelos mesmos motivos do caso de estudo apresentado na Secção 5.3.2, recorreu-se ao Alaska Satellite Facility. Visto que esta API não está tão limitada quanto ao número de *downloads* de produtos, passou-se a uma fase de pré-processamento destes, armazenando-se aproximadamente 90 *gigabytes*. Porém, antes de remover as imagens que não eram necessárias para o processamento, este armazenamento atingia os 260 *gigabytes*. No entanto, importa referir que estes dados apenas foram usados para alguns testes, não tendo sido usados no produto final.

Quanto às FGCI da região de Mação, foi obtido um ficheiro vetorial que representa as mesmas. Este é disponibilizado pelo Instituto de Conservação da Natureza e Florestas (ICNF). Adicionalmente, foi disponibilizado pelo município de Mação um ficheiro vetorial que tem traçadas as faixas que foram intervencionadas em 2018, que foi usado como verdade de terreno.

Computação. Ao longo da computação, a tarefa mais pesada passa pelo recorte dos polígonos que representam as FGCI nas imagens de satélite, resultando em cerca de cinco mil polígonos recortados nas imagens de diferentes bandas e índices dos 20 produtos que constituem a série temporal. Segundo o autor, o tempo despendido neste processamento

é cerca de quatro a cinco horas, não existindo de momento uma versão paralela para este efeito.

Armazenamento. No final deste fluxo de desenvolvimento, o produto é representado por um ficheiro vetorial no qual um polígono representa uma FGCI, tendo metainformação relativa sobre se esta foi ou não intervencionada. Estes produtos gerados não estão acessíveis através de nenhum catálogo, já que se encontram na máquina na qual foi executado o processamento.

5.3.3.2 Fluxo Alternativo

Acesso aos produtos. O acesso aos produtos Sentinel-1 e -2 é muito semelhante ao do caso de estudo da Secção 5.3.2. Contudo, no presente exemplo, as vantagens seriam ainda mais assinaláveis, pois acedeu-se a todos os produtos via interface gráfica, sem existir uma automatização neste acesso.

No que se refere aos dois ficheiros de FGCI, o facto destes estarem presentes numa plataforma partilhada permite que qualquer outro utilizador que necessite aceder a este produto não precise de pesquisar o mesmo em outro catálogo. De facto, como já se percebeu nos casos de estudo já apresentados, a necessidade de aceder a múltiplos catálogos é um problema recorrente.

Computação. Este caso de estudo, tal como os referidos anteriormente (Secção 5.3.1 e 5.3.2), não tem uma versão paralela, o que limita bastante as possibilidades de escalar o algoritmo recorrendo a uma infraestrutura na qual o processamento é realizado em GPU ou até distribuído.

Armazenamento. Passando ao armazenamento do produto, este seria integrado na plataforma. Para este efeito, o metamodelo precisa de ser melhorado, de modo a acrescentar expressividade na pesquisa deste tipo de produto. Nesta metainformação está incluída a metainformação comum descrita na Secção 5.3.1 e também métricas de avaliação semelhantes às identificadas no caso de estudo apresentado na Secção 5.3.2.

Complementarmente, em termos de agregados, estarão presentes no metamodelo elementos como: **área/perímetro total/médio/máximo/mínimo de FGCI**; **percentagem de FGCI intervencionadas**; e, finalmente, **percentagem da área/perímetro total de FGCI intervencionada**. Com esta metainformação é possível pesquisar a região de Portugal com maior percentagem de FGCI intervencionadas, ou até fazer esta pesquisa em termos absolutos, como, por exemplo, pesquisando a região com maiores áreas de FGCI não intervencionadas.

Enriquecimento do metamodelo. No que diz respeito a este passo, um dos elementos de metainformação que seria interessante integrar neste produto passaria por anotar os polígonos com informação referente à elevação do terreno. Este dado seria importante

pois permitiria identificar as FGCI correspondentes às linhas elétricas que passassem por cima de um vale, não necessitando estas de ser intervencionadas.

Paralelamente, ao longo do fluxo de desenvolvimento, identificou-se uma deslocação de um píxel em algumas imagens do Sentinel-2. Por este motivo, calculou-se o vetor de translação necessário para que essas imagens fossem corrigidas. Este vetor trata-se de metainformação relativa à correção de um produto Sentinel-2, devendo este ser acrescentado ao metamodelo do mesmo. Deste modo, qualquer utilizador que recorrer àquela imagem tem indicação do defeito apresentado pela mesma com um vetor de translação, de modo a conseguir corrigi-la.

5.3.4 Fusão de Imagens de Satélite

A dissertação [153] em questão é um estudo sobre o efeito das estruturas permanentes na fusão de imagens de satélite. A fusão de imagens de satélite emergiu, principalmente, devido à existência de cobertura de nuvens em imagens de satélites com sensores óticos, impossibilitando a análise de alguns píxeis. Os modelos de fusão combinam imagens de satélites de maior resolução espacial com os de maior resolução temporal, permitindo estimar imagens ausentes, e até recuperar as que estejam parcialmente cobertas de nuvens.

No que se refere à abordagem, recorre-se a um algoritmo - ESTAIR - que resulta da junção de outros dois, com as devidas adaptações. Foram ainda incorporados dados cartográficos de modo a estudar o efeito das estruturas permanentes nesta fusão. Finalmente, fez-se um estudo comparativo deste algoritmo com diferentes parametrizações.

Na Fig. 5.4 está representada uma imagem gerada a partir deste algoritmo. Uma vez que foram realizados estudos sobre esta região, que permitirão perceber o impacto de usar séries temporais e de integrar informação no algoritmo relativamente à presença de água. O que se concluiu foi que a inclusão das séries temporais melhora a *performance* deste algoritmo. Por outro lado, no que diz respeito à introdução das máscaras de água, até à data, os resultados são inconclusivos.

5.3.4.1 Fluxo de Desenvolvimento

Durante o desenvolvimento da solução foi necessário aceder a um conjunto de produtos de observação terrestre, estando incluídos os seguintes:

- Landsat 8;
- MODIS;
- Open Street Maps.

Acesso aos produtos. Como referido, estes algoritmos de fusão combinam imagens de satélites de maior resolução espacial com os de maior resolução temporal. Quando o objetivo era uma maior resolução espacial, recorreu-se ao Landsat 8; por sua vez, para

Figura 5.4: Imagem gerada através do algoritmo ESTAIR para a região de Santa Cruz da Trapa e São Cristovão de Lafões, correspondente a 02-08-2017. Imagem fornecida pelo autor da dissertação.

uma maior resolução temporal acedeu-se aos produtos do MODIS. Relativamente ao acesso aos mesmos, todo o desenvolvimento foi integrado no GEE, incluindo, lá está, o acesso aos produtos. As vantagens deste acesso passam pelo facto de não ser necessário descarregar os produtos com a finalidade de computá-los, sendo a computação executada junto aos dados. Com esta abordagem, excluiu-se o tempo de automatizar o *download* para infraestruturas locais. Secundariamente, não existem quaisquer preocupações em recortar as regiões de estudo ou de garantir o alinhamento das imagens, já que o GEE abstrai esses pormenores.

Relativamente ao ficheiro vetorial fornecido pelo Open Street Maps, o GEE fornece um mecanismo que permite importar produtos que não estejam presentes no catálogo apresentado por esta plataforma.

Computação. Passando à computação, no GEE, o modelo é naturalmente distribuído. Deste modo, a dificuldade de escalar com os algoritmos desenvolvidos é encurtada, facilitando a realização de computações *ad hoc* pelo facto do acesso aos dados ser tão simples. No entanto, de maneira a garantir um paradigma que abstraísse a distribuição dos dados e computação, este tornou-se algo complexo. Sendo que, nesta dissertação, o facto de o conjunto de operações estar limitado aos pré-definidos no GEE levou a que existisse uma fase complicada de aprendizagem. Para além do referido, o *debug* dos algoritmos desenvolvidos também se tornou uma tarefa bastante mais complexa do que nos paradigmas típicos de processamento de imagem em série.

Armazenamento. O resultado do processamento subdivide-se em duas componentes: uma imagem em GeoTIFF; e um CSV com as medidas de avaliação associadas. Em relação à imagem gerada, esta poderá representar uma de duas coisas: a substituição de uma imagem ausente ou a correção de uma imagem existente.

No que toca à disponibilização deste produto, o GEE oferece ferramentas que permitem exportar o mesmo, enviando-se, por exemplo, para a Google Drive. Ou, por outro lado, criar uma aplicação *Web* que permita partilhar possíveis visualizações do resultado da computação.

5.3.4.2 Fluxo Alternativo

Acesso aos produtos. Neste fluxo alternativo existem duas opções: ou se computa o produto no GEE e armazena-se o mesmo na plataforma proposta ou inclui-se todo o desenvolvimento nessa mesma plataforma.

Se se optar pela primeira opção, o acesso aos produtos é bastante mais simples que o da plataforma desenvolvida, pois este está incorporado na API para Python e Javascript. No entanto, caso venha a ser necessário um produto que não esteja disponível no catálogo do GEE, não existe a flexibilidade que a arquitetura proposta nesta dissertação fornece.

Quanto à segunda opção, o acesso aos produtos não seria tão simples, pois não há uma integração sistemática com nenhuma linguagem. De outro ponto de vista, o facto de se recorrer à plataforma facilitaria na partilha dos produtos necessários à computação. De forma a ilustrar o referido, o caso de estudo referido na Secção 5.3.2 já tinha integrado o produto do Open Street Maps na plataforma. Por outro lado, com esta abordagem, as séries temporais do Landsat 8 e do MODIS descarregadas para a plataforma seriam úteis para os outros casos de estudo, testando os algoritmos com imagens que não sejam obtidas pelo Sentinel. Contudo, no protótipo já existem adaptadores das fontes necessárias para aceder tanto aos produtos do Landsat como do MODIS. Todos estes pormenores reforçam a ideia de uma gestão partilhada deste repositório de dados e metadados de observação terrestre.

Computação. Como referido nos parágrafos anteriores, caso se decidisse realizar a computação no GEE as vantagens de escalabilidade são múltiplas. O problema desta abordagem passa pela portabilidade da solução. Ou seja, ao recorrer às API do GEE apenas se pode executar a computação nessa mesma plataforma. Se em algum momento se decidir computar estes produtos em outra infraestrutura, o esforço de adaptação é assinalável.

Por outro lado, ao desenvolver uma solução para executar o processamento na plataforma proposta, os pormenores de distribuição da computação não são abstraídos como no caso do GEE.

Armazenamento. Na fase de armazenamento do produto, o GEE é bastante limitado. Este facto resulta de uma ideia referida múltiplas vezes ao longo desta dissertação: o foco do GEE é a computação e não a catalogação dos produtos resultantes. Por este motivo, mesmo que se tire partido das vantagens do GEE (*e.g.* paradigma de computação distribuída), a melhor solução acaba por ser extrair o produto resultante de modo a armazená-lo

no repositório proposto. Apesar desta extração ser muitas vezes uma tarefa demorada, esta solução traz enormes vantagens ao nível da gestão dos produtos e seus metamodelos.

No que toca às especificidades do metamodelo correspondente este inclui: as diferentes **medidas de avaliação** da solução (semelhantes ao caso de estudo da Secção 5.3.2); e também um histórico dos **produtos processados**.

Enriquecimento do metamodelo. Relativamente ao enriquecimento do metamodelo, não se encontrou um exemplo claro. Ainda assim, o que importa refletir é que, na eventualidade de extensão do metamodelo, a integração desses metadados é bastante simples, como demonstrado nos outros casos de estudo apresentados.

Contrariamente ao que acontece no GEE, no qual não existe uma sistematização, nem ao nível da definição do metamodelo, nem ao nível de distribuição pública dos produtos gerados. Pode-se argumentar que as ferramentas do GEE para gerar aplicações *Web* seja uma forma perfeitamente válida de apresentar este tipo de produtos. Todavia, quando se recorre à palavra "sistemática", refere-se a API de acesso a estes produtos bem definidas, permitindo a automatização.

5.4 Conclusão

Após discutir os múltiplos casos de estudos reais, importa refletir globalmente sobre os benefícios da solução apresentada. É assinalável a quantidade de tipos de produtos diferentes em apenas quatro casos de estudo. Sendo que com a integração de todos eles neste repositório, a sua oferta já seria bastante interessante.

Em relação ao acesso aos produtos, nos diferentes casos de estudo foram descritas formas de acesso bastante válidas. No que diz respeito ao acesso via interface gráfica, este é bastante satisfatório em portais como o do Copernicus, no entanto existe a necessidade de automatizar todo este fluxo de pesquisa e acesso. Para este efeito surgiram ferramentas como a API sentinelsat, esta permite automatizar a pesquisa e *download* apenas dos produtos Sentinel, sendo que a granularidade do acesso é realizado ao nível do produto e não banda a banda. Existem também casos em que os produtos estão *offline*, para este problema a arquitetura proposta apresenta dois tipos de solução: desenvolver novos adaptadores para API que permitam solicitar o acesso a esses produtos; ou, por outro lado, introduzir redundância nas fontes, identificando as que têm o produto *online*.

Nos casos de estudo é possível identificar que muitos são os produtos comuns entre os mesmos. O facto destes estarem partilhados numa só plataforma, leva a que se evite o armazenamento local de um grande volume de dados por parte dos utilizadores. Adicionalmente, permite que colaborem para enriquecer o repositório com novos produtos via API, sem que tenham de requisitar aos administradores do sistema a sua integração.

Uma ausência relevante no protótipo é a possibilidade de pesquisar a percentagem de nuvens ao nível da região e não ao nível da cena. Esta é uma funcionalidade que seria transversalmente interessante a todos os casos de estudo apresentados, já que estes

trabalharam sobre uma subárea das cenas. Para este efeito, assinalou-se este mecanismo como trabalho futuro, expondo-se os passos necessários para a sua implementação.

Passando à comparação com o GEE, esta teve o expoente máximo no último caso de estudo apresentado. Nos restantes, referiu-se o GEE como sendo útil em algumas computações e visualizações *ad hoc*, mas não como solução final. Isto porque, em alguns casos, a implementação dos algoritmos seria bastante mais complexa, devido ao paradigma naturalmente distribuído da programação. No entanto, a escalabilidade que o GEE oferece é claramente maior que a da solução proposta. Em algumas situações, se as necessidades de computação forem significativas, aconselha-se que se recorra ao GEE para a computação do produto, exportando-o a seguir para o repositório proposto, dado que é na fase de catalogação dos produtos gerados que o GEE tem maiores insuficiências. Se esta for a decisão tomada, rapidamente se percebe a facilidade com que é possível importar os dados nas API para Javascript e Python.

No que diz respeito ao armazenamento, a presença de metamodelos bem definidos com possibilidade de extensão facilita as interrogações, assegura a qualidade dos dados e ajuda à extensibilidade da solução. Nos casos de estudo foram ilustrados os elementos e mecanismos que garantem estas propriedades. Na metainformação estão incluídos dados: que comprovam a qualidade do produto; agregações que visam facilitar as pesquisas; e também informação transversal a todos os produtos, o que permite realizar interrogações nas dimensões espaço, tempo e banda.

Finalmente, acerca do enriquecimento do metamodelo, este só tem paralelo no CMR, o qual fornece ferramentas de gestão dos metadados. O foco no CMR está na correção dos metadados e na integração de múltiplos *standards* de metamodelação. Nesta plataforma o foco é garantir a extensibilidade dos metamodelos, já que estes são naturalmente dinâmicos, através da contribuição dos múltiplos utilizadores, ilustrados pelos vários exemplos concretos de extensões do metamodelo dos produtos gerados nos diferentes casos de estudo. Desta forma, a solução proposta reúne algumas das vantagens de computação do GEE e as de gestão do metamodelo do CMR, ao agregar tudo numa só plataforma que envolve todo o fluxo de desenvolvimento desde o acesso aos produtos até ao ciclo de enriquecimento dos metadados.

Capítulo 6

Conclusão

A deteção remota tem vindo a tornar-se o método mais importante de recolha de informação sobre a superfície terrestre. Estas tecnologias de observação terrestre tornaram-se um símbolo de capacidade científica e tecnológica, de força económica e de segurança nacional. Isto levou a um crescimento no desenvolvimento de tecnologias de satélite, e por isso a um aumento do número de aplicações que recorrem a estes dados. Os dados gerados por estes revelam caraterísticas de *big data*: volume; velocidade; variedade. Por este motivo, existe a necessidade da criação de repositórios que recorram a técnicas específicas deste domínio.

De facto, o volume e a complexidade da informação gerados pelas diferentes missões e correspondentes satélites são consideráveis. A metainformação é igualmente complexa, pois nesta está incluída a especificação de cada satélite e os dados específicos da imagem. Para além da informação originalmente recolhida pelos satélites, denominados produtos básicos, existem também produtos derivados. Dentro dos derivados surgem: índices diretamente oferecidos pelos fornecedores ou calculados por terceiros, produtos com o objetivo de melhoria espacial ou temporal das imagens originais ou índices derivados e produtos de classificação.

Efetivamente, os múltiplos fornecedores já têm os seus próprios repositórios e catálogos. No entanto, existem algumas insuficiências nestes. Nomeadamente, ao nível: da heterogeneidade dos dados e metadados; da extensibilidade dos metamodelos; da expressividade nas interrogações; e na incorporação de cadeias de processamento local.

Com isto em mente, consultou-se a literatura relativa a este tema. Em relação aos catálogos e plataformas, um dos principais problemas é realmente a expressividade nas pesquisas. Por outro lado, algo que dificulta a pesquisa é o facto de muitas das vezes ter de se consultar vários destes catálogos. Relativamente ao acesso aos produtos, por estes fornecedores servirem necessidades tão amplas, têm de limitar o acesso às suas

API. Imaginando que todos os elementos de uma equipa necessitam do mesmo produto, cada um terá de fazer o *download* do mesmo. Na verdade, este problema não se põe nas plataformas, pois estas incorporam processamento na infraestrutura onde estão armazenados os dados. De assinalar que a plataforma que representa o estado da arte atual é o GEE. Os principais problemas desta passam pela perda de flexibilidade no paradigma de programação imposto; pelo limitado mecanismo de definição do metamodelo dos novos produtos computados; e pela ausência de uma gestão colaborativa da metainformação.

Da multitude de possíveis soluções, a proposta nesta dissertação consiste numa plataforma de desenvolvimento colaborativo de produtos de observação terrestre. Para esse efeito: abordou-se a heterogeneidade das fontes e de que forma esta é integrada na plataforma; definiu-se um claro fluxo dos dados e metadados; apresentou-se a técnica de especificação do metamodelo; referiu-se os possíveis cenários de extensibilidade do mesmo, juntamente com uma linguagem para especificar as extrações necessárias; e, por fim, foram definidas as operações disponibilizadas aos utilizadores via API.

O objetivo desta solução não é substituir sistemas como o GEE ou o Copernicus Open Access Hub. Esta é indicada para equipas que trabalhem num domínio aplicacional comum, pois a flexibilidade da mesma permite que se adicione facilmente produtos específicos do domínio, e, por outro lado, se enriqueça o metamodelo com dados da disciplina em que se está a trabalhar. Adicionalmente, os produtos computados são armazenados e disponibilizados de uma forma sistemática. De realçar que o foco principal da plataforma proposta não é a computação, apesar de existirem componentes para esse fim. O foco é sim criar uma *framework* de gestão colaborativa de produtos de observação terrestre.

Comparando com o GEE, a preocupação deste é ser genérico no seu metamodelo e nos produtos disponibilizados. Não existindo um envolvimento com o domínio em que os utilizadores estão a trabalhar. No entanto, existem vários cenários em que recorrer ao GEE faz todo o sentido, nomeadamente se a computação for pesada. Nesse caso, o GEE abstrai toda a distribuição da computação através do paradigma de programação que oferece nas suas API.

Em relação a catálogos como o Copernicus, esta plataforma não se apresenta como uma alternativa, mas sim como uma extensão, pois os produtos são obtidos nesses mesmos catálogos. De facto, um dos propósitos é evitar que os utilizadores tenham de consultar diferentes catálogos, em que cada um apresenta interfaces, metamodelos e formatos distintos.

Relativamente ao protótipo, este representa um subconjunto representativo das funcionalidades apresentadas na abordagem. Ao longo de todo o processo de desenvolvimento do mesmo, muitas foram as dificuldades de integração das múltiplas fontes de dados. O desafio passou por deixar o sistema genérico o suficiente para que não explodisse em complexidade. Interessa também referir que nas decisões de arquitetura se optou pelas tecnologias que davam melhores garantias. Consequentemente, abdicou-se de algumas funcionalidades, tendo-se sempre como critério de seleção o facto de serem ou não vitais para a prova de conceito.

De modo a avaliar este repositório, recorreu-se a casos de estudo reais, o que oferece uma maior credibilidade a este juízo. É de assinalar que apenas com os produtos processados e gerados por este conjunto de casos de estudo o repositório já fica bastante satisfatório. Tendo-se identificado que muitos são os produtos comuns a que recorrem os múltiplos casos de estudo. Desta forma, evita-se o armazenamento local por parte dos utilizadores, permitindo que colaborem entre eles de forma a enriquecer o repositório via API, sem que tenham de requisitar aos administradores do sistema a sua integração.

Comparando com GEE, este é bastante útil para realizar visualizações *ad hoc*, mas nem tanto como solução definitiva. Pois a implementação de alguns algoritmos tornar-se-ia bastante mais complexa, o que resulta do paradigma de programação. Relativamente à escalabilidade, as garantias oferecidas pelo GEE são claramente maiores. Sendo que se aconselha que, para computações pesadas, se recorra ao GEE, exportando de seguida o produto gerado para o repositório proposto, uma vez que é na fase de catalogação dos produtos gerados que o GEE tem maiores insuficiências. Através dos casos de estudo foram comprovadas as vantagens de ter metamodelos bem definidos com possibilidade de extensão, assegurando a qualidade dos dados e a expressividade do sistema.

Em conclusão, apesar do contexto multidisciplinar desta dissertação, pensa-se ter atingido os objetivos deste projeto. Foi proposta uma solução para a resolução do problema formulado que foi validada através de uma comparação com a plataforma que constitui atualmente o estado da arte (*i.e.* GEE), concluindo-se que a arquitetura definida permite aliar a computação à catalogação de uma forma sistemática. Finalmente, o facto desta comparação ter como base um conjunto de casos de estudo diversos permite transmitir maior confiança relativamente às conclusões tiradas.

6.1 Trabalho Futuro

No enquadramento desta dissertação desenvolveu-se um protótipo funcional, baseado na arquitetura proposta. No entanto, não obstante de estarem fora do escopo deste projeto, ainda existem áreas que poderiam ser melhoradas. Por outro lado, há possibilidade de desenvolver funcionalidades completamente novas que enriqueçam o sistema. Para este efeito, de seguida propõe-se uma multitude de aspetos a considerar no trabalho futuro:

Expandir especificação do protótipo. Na Secção 4.1.1 foram indicadas as funcionalidades, que apesar de estarem presentes na solução proposta, eram dispensáveis para a prova de conceito. A especificação destas é feita ao longo do Capítulo 3. Os aspetos que devem ser considerados como prioritários são os seguintes: área isolada de computação; interface gráfica; desenvolver um adaptador para API de encomenda (Secção 3.3.1); desenvolver um adaptador para API com mecanismos de *push*; implementar uma configuração *a priori* (Secção 3.3.2).

Autenticação baseada em papéis. Implementar na API um mecanismo de segurança que permita uma autenticação baseada na função do utilizador. À partida surgem três papéis principais: administrador, programador, convidado. O administrador poderá executar qualquer operação; o programador poderá criar a sua extensão aos metamodelos ou adicionar novos produtos; e, por fim, o convidado apenas tem permissão para operações de consulta sobre os produtos e respetivos metamodelos.

Integrar máscaras de nuvens. Como foi referido na Secção 5.3.1 é necessária a incorporação de um filtro que nos permita pesquisar pela percentagem de nuvens de uma região em específico, e não de toda a cena. Na verdade, este é um problema transversal a todas as máscaras disponibilizadas (*e.g.* nuvens, neve, área ardida). Para a resolução deste problema, as três abordagens encontradas são as seguintes:

- **Máscara integrada nos metadados**: no protótipo desenvolvido, quando se obtém a máscara de nuvens apenas se guarda esta no sistema de ficheiros, não existindo qualquer indexação da mesma. A alternativa seria integrar o ficheiro vetorial no documento de metadados, o que não seria muito complicado dada a existência da linguagem de especificação de ETL (Secção 3.4.3). Deste modo, seria possível operar *queries* geoespaciais sobre essas mesmas máscaras, o que permitiria, entre outras coisas, interrogar se uma dada área da cena tem menos x% de nuvens. Esta abordagem não é tão eficiente como a referida abaixo já que os dados não estão indexados à partida, ou seja, para responder à *query* referida é necessário computar a percentagem de nuvens numa determinada região de estudo.

- **Indexar subáreas**: indexar a percentagem de nuvens para sub-regiões da cena. O problema desta abordagem é que a região selecionada pelo utilizador poderá intersetar múltiplas sub-regiões da cena;

- **PostGIS**: explorar as funcionalidades do PostGIS, no que diz respeito à agregação de píxeis dado um certo polígono, ou seja, computando a percentagem de píxeis com nuvens numa dada região. De notar que esta última hipótese poderá não ser viável, pois foi explorada no contexto desta dissertação.

API Python. O GEE disponibiliza uma API para Python e Javascript, através da qual é possível aceder aos produtos sem preocupações relativas à localização física dos mesmos. Para este repositório seria possível implementar algo semelhante, sendo que a API REST era a componente que suportava a API de uma linguagem específica, ou seja, toda a comunicação com o servidor era ocultada por uma biblioteca. Adicionalmente, aconselha-se que esta seja implementada primeiro para Python, pois é a linguagem mais usada no contexto da deteção remota.

Automatização do processamento. Finalmente, é importante criar mecanismos de automatização do processamento. Um exemplo disto passa por despoletar uma determinada computação sobre as imagens obtidas sempre que se obtém um novo produto do Sentinel-2. Esta computação poderá ser, a título de exemplo, a identificação de áreas ardidas.

Bibliografia

[1] UCS. *UCS Satellite Database*. 2018. URL: https://www.ucsusa.org/nuclear-weapons/space-weapons/satellite-database. Acedido em 11-09-2019.

[2] Z. Chen, Q. Zhou, J. Liu, L. Wang, J. Ren, Q. Huang, H. Deng, L. Zhang e D. Li. "Charms - China Agricultural Remote Sensing Monitoring System". Em: *2011 IEEE International Geoscience and Remote Sensing Symposium*. 2011, pp. 3530–3533. DOI: 10.1109/IGARSS.2011.6049983.

[3] S. Veraverbeke, P. Dennison, I. Gitas, G. Hulley, O. Kalashnikova, T. Katagis, L. Kuai, R. Meng, D. Roberts e N. Stavros. "Hyperspectral remote sensing of fire: State-of-the-art and future perspectives". Em: *Remote Sensing of Environment* 216 (2018), pp. 105 –121. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2018.06.020.

[4] E. Marino, P. Ranz, J. L. Tomé, M. Ángel Noriega, J. Esteban e J. Madrigal. "Generation of high-resolution fuel model maps from discrete airborne laser scanner and Landsat-8 OLI: A low-cost and highly updated methodology for large areas". Em: *Remote Sensing of Environment* 187 (2016), pp. 267 –280. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2016.10.020.

[5] G. H. Mitri e I. Z. Gitas. "Fire type mapping using object-based classification of Ikonos imagery". Em: *International Journal of Wildland Fire* 15.4 (2006), p. 457. DOI: 10.1071/wf05085.

[6] S. H. Peterson, J. Franklin, D. A. Roberts e J. W. van Wagtendonk. "Mapping fuels in Yosemite National Park". Em: *Canadian Journal of Forest Research* 43.1 (2013), pp. 7–17. DOI: 10.1139/cjfr-2012-0213.

[7] D. Roberts, P. Dennison, M. Gardner, Y. Hetzel, S. Ustin e C. Lee. "Evaluation of the potential of hyperion for fire danger assessment by comparison to the airborne visible/infrared imaging spectrometer". Em: *IEEE Transactions on Geoscience and Remote Sensing* 41.6 (2003), pp. 1297–1310. DOI: 10.1109/tgrs.2003.812904.

[8] E. Chuvieco, D. Cocero, D. Riaño, P. Martin, J. Martiñez-Vega, J. de la Riva e F. Pérez. "Combining NDVI and surface temperature for the estimation of live fuel moisture content in forest fire danger rating". Em: *Remote Sensing of Environment* 92.3 (2004), pp. 322–331. DOI: 10.1016/j.rse.2004.01.019.

[9] R. Meng, J. Wu, K. L. Schwager, F. Zhao, P. E. Dennison, B. D. Cook, K. Brewster, T. M. Green e S. P. Serbin. "Using high spatial resolution satellite imagery to map forest burn severity across spatial scales in a Pine Barrens ecosystem". Em: *Remote Sensing of Environment* 191 (2017), pp. 95 –109. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2017.01.016.

[10] B. Yu, F. Chen, B. Li, L. Wang e M. Wu. "Fire Risk Prediction Using Remote Sensed Products: A Case of Cambodia". Em: *Photogrammetric Engineering & Remote Sensing* 83.1 (2017), pp. 19–25. DOI: 10.14358/pers.83.1.19.

[11] L. Giglio, J. Descloitres, C. O. Justice e Y. J. Kaufman. "An Enhanced Contextual Fire Detection Algorithm for MODIS". Em: *Remote Sensing of Environment* 87.2-3 (2003), pp. 273–282. DOI: 10.1016/s0034-4257(03)00184-6.

[12] W. Schroeder, P. Oliva, L. Giglio e I. A. Csiszar. "The New VIIRS 375m active fire detection data product: Algorithm description and initial assessment". Em: *Remote Sensing of Environment* 143 (2014), pp. 85–96. DOI: 10.1016/j.rse.2013.12.008.

[13] P. M. Barbosa, J.-M. Grégoire e J. M. C. Pereira. "An Algorithm for Extracting Burned Areas from Time Series of AVHRR GAC Data Applied at a Continental Scale". Em: *Remote Sensing of Environment* 69.3 (1999), pp. 253–263. DOI: 10.1016/s0034-4257(99)00026-7.

[14] L. Giglio, T. Loboda, D. P. Roy, B. Quayle e C. O. Justice. "An active-fire based burned area mapping algorithm for the MODIS sensor". Em: *Remote Sensing of Environment* 113.2 (2009), pp. 408–420. DOI: 10.1016/j.rse.2008.10.006.

[15] I. Z. Gitas, A. Polychronaki, T. Katagis e G. Mallinis. "Contribution of remote sensing to disaster management activities: A case study of the large fires in the Peloponnese, Greece". Em: *International Journal of Remote Sensing* 29.6 (2008), pp. 1847–1853. DOI: 10.1080/01431160701874553.

[16] T. Katagis, I. Z. Gitas, P. Toukiloglou, S. Veraverbeke e R. Goossens. "Trend analysis of medium- and coarse-resolution time series image data for burned area mapping in a Mediterranean ecosystem". Em: *International Journal of Wildland Fire* 23.5 (2014), p. 668. DOI: 10.1071/wf12055.

[17] N. Koutsias e M. Karteris. "Burned area mapping using logistic regression modeling of a single post-fire Landsat-5 Thematic Mapper image". Em: *International Journal of Remote Sensing* 21.4 (2000), pp. 673–687. DOI: 10.1080/014311600210506.

[18] J. M. C. Pereira. "Remote sensing of burned areas in tropical savannas". Em: *International Journal of Wildland Fire* 12.4 (2003), p. 259. DOI: 10.1071/wf03028.

[19] D. Roy, Y. Jin, P. Lewis e C. Justice. "Prototyping a global algorithm for systematic fire-affected area mapping using MODIS time series data". Em: *Remote Sensing of Environment* 97.2 (2005), pp. 137–162. DOI: 10.1016/j.rse.2005.04.007.

[20] J. Eidenshink, B. Schwind, K. Brewer, Z.-L. Zhu, B. Quayle e S. Howard. "A Project for Monitoring Trends in Burn Severity". Em: *Fire Ecology* 3.1 (2007), pp. 3–21. DOI: 10.4996/fireecology.0301003.

[21] R. Meng, J. Wu, K. L. Schwager, F. Zhao, P. E. Dennison, B. D. Cook, K. Brewster, T. M. Green e S. P. Serbin. "Using high spatial resolution satellite imagery to map forest burn severity across spatial scales in a Pine Barrens ecosystem". Em: *Remote Sensing of Environment* 191 (2017), pp. 95 –109. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2017.01.016.

[22] S. Veraverbeke, S. Lhermitte, W. Verstraeten e R. Goossens. "The temporal dimension of differenced Normalized Burn Ratio (dNBR) fire/burn severity studies: The case of the large 2007 Peloponnese wildfires in Greece". Em: *Remote Sensing of Environment* 114.11 (2010), pp. 2548–2563. DOI: 10.1016/j.rse.2010.05.029.

[23] S. A. Lewis, A. T. Hudak, P. R. Robichaud, P. Morgan, K. L. Satterberg, E. K. Strand, A. M. S. Smith, J. A. Zamudio e L. B. Lentile. "Indicators of burn severity at extended temporal scales: a decade of ecosystem response in mixed-conifer forests of western Montana". Em: *International Journal of Wildland Fire* 26.9 (2017), p. 755. DOI: 10.1071/wf17019.

[24] D Riaño. "Assessment of vegetation regeneration after fire through multitemporal analysis of AVIRIS images in the Santa Monica Mountains". Em: *Remote Sensing of Environment* 79.1 (2002), pp. 60–71. DOI: 10.1016/s0034-4257(01)00239-5.

[25] W. J. D. van Leeuwen, G. M. Casady, D. G. Neary, S. Bautista, J. A. Alloza, Y. Carmel, L. Wittenberg, D. Malkinson e B. J. Orr. "Monitoring post-wildfire vegetation response with remotely sensed time-series data in Spain, USA and Israel". Em: *International Journal of Wildland Fire* 19.1 (2010), p. 75. DOI: 10.1071/wf08078.

[26] S. Veraverbeke, I. Gitas, T. Katagis, A. Polychronaki, B. Somers e R. Goossens. "Assessing post-fire vegetation recovery using red–near infrared vegetation indices: Accounting for background and vegetation variability". Em: *ISPRS Journal of Photogrammetry and Remote Sensing* 68 (2012), pp. 28–39. DOI: 10.1016/j.isprsjprs.2011.12.007.

[27] *100 Earth Shattering Remote Sensing Applications Uses*. 2019. URL: https://gisgeography.com/100-earth-remote-sensing-applications-uses/. Acedido em 11-09-2019.

[28] NASA. *EOSDIS Cloud Evolution*. 2018. URL: https://earthdata.nasa.gov/about/eosdis-cloud-evolution. Acedido em 28-01-2019.

[29] J. W. J. B. Hampapuram Ramapriyan Jennifer Brennan. *Managing Big Data:NASA Tackles Complex Data Challengese*. 2013. URL: https://eijournal.com/print/articles/managing-big-data. Acedido em 28-01-2019.

[30] H. Guo, L. Wang e D. Liang. "Big Earth Data from space: a new engine for Earth science". Em: *Science Bulletin* 61.7 (2016), pp. 505–513. ISSN: 20959281. DOI: 10.1007/s11434-016-1041-y.

[31] NASA. *Daily ingest of data into the EOSDIS archive*. 2018. URL: https://cdn.earthdata.nasa.gov/conduit/upload/6571/estimated_changes_in_5-year_daily_data_volume_ingest_crop.jpg. Acedido em 28-01-2019.

[32] *About Copernicus Land Monitoring Service*. URL: https://land.copernicus.eu/about. Acedido em 06-02-2019.

[33] *Docker - Enterprise Container Platform*. URL: https://www.docker.com/. Acedido em 31-07-2019.

[34] R. H. W. James B. Campbell. *Database System Concepts*. Fifth. Introduction to Remote Sensing, 2011. ISBN: 978-1609181765.

[35] ESA. *What is remote sensing?* 2010. URL: http://www.esa.int/SPECIALS/Eduspace_EN/SEMF9R3Z2OF_0.html. Acedido em 22-01-2019.

[36] NASA. *Landsat Satellite Missions*. URL: https://www.usgs.gov/land-resources/nli/landsat/landsat-satellite-missions?qt-science_support_page_related_con=2#qt-science_support_page_related_con. Acedido em 23-01-2019.

[37] NASA. *About MODIS*. URL: https://modis.gsfc.nasa.gov/about/. Acedido em 23-01-2019.

[38] NASA. *Terra Applications*. URL: https://terra.nasa.gov/science/applications. Acedido em 23-01-2019.

[39] NASA. *Aqua Images + Data*. URL: https://aqua.nasa.gov/content/images-data. Acedido em 23-01-2019.

[40] NOAA. *Satellite Channels Overview*. URL: https://www.goes.noaa.gov/sat-explanation.html. Acedido em 23-01-2019.

[41] NASA. *NPP Mission Overview*. URL: https://www.nasa.gov/mission_pages/NPP/mission_overview/index.html. Acedido em 23-01-2019.

[42] ESA. *Copernicus Overview*. URL: https://www.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Overview4. Acedido em 23-01-2019.

[43] EUMETSAT. *Meteosat Second Generation (MSG) provides images of the full Earth disc, and data for weather forecasts*. URL: https://www.eumetsat.int/website/home/Satellites/CurrentSatellites/Meteosat/index.html. Acedido em 23-01-2019.

[44] JAXA. *About ALOS - Overview and Objectives*. URL: https://www.eorc.jaxa.jp/ALOS/en/about/about_index.htm. Acedido em 23-01-2019.

[45] NASA. *About the ESDS Program*. 2018. URL: https://earthdata.nasa.gov/earth-science-data-systems-program/about-the-esds-program. Acedido em 04-02-2019.

[46] NASA. *About EOSDIS*. 2018. URL: https://earthdata.nasa.gov/about. Acedido em 04-02-2019.

[47] CEOS. *About CEOS*. URL: http://ceos.org/about-ceos/overview/. Acedido em 04-02-2019.

[48] NASA. *Learn About GCMD*. URL: https://gcmd.nasa.gov/learn/index.html. Acedido em 04-02-2019.

[49] R. Simmon. *Ellipse Diagram*. URL: https://earthobservatory.nasa.gov/ContentFeature/OrbitsCatalog/images/ellipse_diagram.png. Acedido em 23-01-2019.

[50] H. Riebeek. *Catalog of Earth Satellite Orbits*. 2009. URL: https://earthobservatory.nasa.gov/features/OrbitsCatalog. Acedido em 23-01-2019.

[51] USGS. *What are the orbit paths of the Landsat satellites?* URL: https://landsat.usgs.gov/what-are-orbit-paths-landsat-satellites. Acedido em 23-01-2019.

[52] GISGeography. *How Universal Transverse Mercator (UTM) Works*. 2018. URL: https://gisgeography.com/utm-universal-transverse-mercator-projection/. Acedido em 25-01-2019.

[53] *Utm-zones*. URL: https://pt.wikipedia.org/wiki/Universal_Transversa_de_Mercator#/media/File:Utm-zones.jpg. Acedido em 25-01-2019.

[54] NASA. *Remote Sensors*. 2018. URL: https://earthdata.nasa.gov/user-resources/remote-sensors. Acedido em 24-01-2019.

[55] NRCan. *Temporal Resolution*. 2014. URL: https://www.nrcan.gc.ca/earth-sciences/geomatics/satellite-imagery-air-photos/satellite-imagery-products/educational-resources/9365. Acedido em 24-01-2019.

[56] NRCan. *Spatial Resolution*. 2015. URL: https://www.nrcan.gc.ca/earth-sciences/geomatics/satellite-imagery-air-photos/satellite-imagery-products/educational-resources/9407. Acedido em 24-01-2019.

[57] NRCan. *Spectral Resolution*. 2015. URL: https://www.nrcan.gc.ca/earth-sciences/geomatics/satellite-imagery-air-photos/satellite-imagery-products/educational-resources/9393. Acedido em 24-01-2019.

[58] NRCan. *Radiometric Resolution*. 2016. URL: https://www.nrcan.gc.ca/earth-sciences/geomatics/satellite-imagery-air-photos/satellite-imagery-products/educational-resources/9379. Acedido em 24-01-2019.

[59] ESA. *Sentinel Resolutions*. URL: https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-2-msi/resolutions. Acedido em 26-01-2019.

[60] K. Zanter. *Landsat 7 (L7) Data Users Handbook*. English. Versão 1.0. USGS. 154 pp. URL: https://landsat.usgs.gov/sites/default/files/documents/LSDS-1927_L7_Data_Users_Handbook.pdf.

[61] NASA. *MODIS*. 2018. URL: https://ladsweb.modaps.eosdis.nasa.gov/missions-and-measurements/modis/. Acedido em 26-01-2019.

[62] NASA. *MODIS Specifications*. URL: https://modis.gsfc.nasa.gov/about/specifications.php. Acedido em 26-01-2019.

[63] NASA. *NASA Base Metadata Requirements*. 2016. URL: https://wiki.earthdata.nasa.gov/display/NASAISO/NASA+Base+Metadata+Requirements. Acedido em 04-02-2019.

[64] NASA. *Sentinel-2 Product Types*. URL: https://science.nasa.gov/earth-science/earth-science-data/data-processing-levels-for-eosdis-data-products. Acedido em 05-02-2019.

[65] ESA. *Sentinel-2 Product Types*. URL: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/product-types. Acedido em 06-02-2019.

[66] *Science Toolbox Exploitation Platform*. URL: http://step.esa.int/main/. Acedido em 20-02-2019.

[67] ESA. *Sentinel 2 Data Formats*. URL: https://earth.esa.int/web/sentinel/user-guides/sentinel-2-msi/data-formats. Acedido em 08-02-2019.

[68] A Gatti, C Naud, C Castellani e F Carriero. *Sentinel-2 Products Specification Document*. 2018.

[69] L. Colaiacomo, J. Masó, E. Devys e E. Hirschorn. *OGC GML in JPEG 2000 (GMLJP2) Encoding Standard*. 2017. URL: http://docs.opengeospatial.org/is/08-085r8/08-085r8.html. Acedido em 08-02-2019.

[70] L.Johnson. *Landsat 7 (L7) Enhanced Thematic Mapper Plus (ETM+) Level 1 (L1) Data Format Control Book (DFCB)*. English. Versão 14.0. USGS. 2009. 125 pp.

[71] M. R. Niles Ritter. *GeoTIFF Format Specification*. Standard. 1995.

[72] E. P. Mike Folk. *HDF5 Data Model, File Format and Library – HDF5 1.6*. English. NASA. 2007. 42 pp.

[73] L. A. Wasser. *Hierarchical Data Formats - What is HDF5?* URL: https://www.neonscience.org/about-hdf5. Acedido em 09-02-2019.

[74] T. Mistelbauer. "Metadata Management of Higher Level Remote Sensing Products". Tese de mestrado. Áustria: Vienna University of Technology, 2012. URL: https://publik.tuwien.ac.at/files/PubDat_208567.pdf. Acedido em 30-01-2019.

[75] R. B. Ben White. *NASA Base Metadata Requirements*. 2016. URL: https://wiki.earthdata.nasa.gov/display/NASAISO/NASA+Base+Metadata+Requirements. Acedido em 30-01-2019.

[76] ISO. *ISO 19115:2003 Geographic information – Metadata*. 2003. URL: https://www.iso.org/standard/26020.html. Acedido em 30-01-2019.

[77] NASA. *ISO 191** Metadata Family*. 2016. URL: https://wiki.earthdata.nasa.gov/display/NASAISO/ISO+191**+Metadata+Family. Acedido em 31-01-2019.

[78] ISO. *ISO 19115-2:2009*. 2009. URL: https://www.iso.org/standard/39229.html. Acedido em 30-01-2019.

[79] ISO. *ISO/TS 19115-3:2016*. 2016. URL: https://www.iso.org/standard/32579.html. Acedido em 31-01-2019.

[80] ISO. *ISO/TS 19139:2007*. 2007. URL: https://www.iso.org/standard/32557.html. Acedido em 31-01-2019.

[81] ISO. *ISO 19157:2013*. 2013. URL: https://www.iso.org/standard/32575.html. Acedido em 31-01-2019.

[82] ISO. *Geographic information – Observations and measurements*. 2011. URL: https://www.iso.org/standard/32574.html. Acedido em 14-02-2019.

[83] J. Gasperi, F. Houbie, A. Woolf e S. Smolders. *OGC Earth Observation Metadata profile of Observations Measurements*. 2014. URL: http://docs.opengeospatial.org/is/10-157r4/10-157r4.html.

[84] INSPIRE. *About INSPIRE*. URL: https://inspire.ec.europa.eu/about-inspire/563. Acedido em 31-01-2019.

[85] Drafting Team Metadata and European Commission Joint Research Centre. "INSPIRE Metadata Implementing Rules : Technical Guidelines based on EN ISO 19115 and EN ISO 19119 - V.1.3." Em: (2013), p. 99. URL: https://inspire.ec.europa.eu/documents/inspire-metadata-implementing-rules-technical-guidelines-based-en-iso-19115-and-en-iso-1.

[86] J. Cetl, Vlado.; Nunes De Lima, Vanda; Tomas, Robert.; Lutz, Michael. ;D 'eugenio, Joachim; Nagy, Adam; Robbrecht, Joeri; De Lima, Nunes; Tomas, Robert; Lutz, Michael; D 'eugenio, J; Nagy, Adam; Robbrecht. *Summary Report on Status of implementation of the INSPIRE Directive in EU*. 2017. URL: http://publications.jrc.ec.europa.eu/repository/bitstream/JRC109035/jrc109035_jrc109035_jrc_inspire_eu_summaryreport_online.pdf.

[87] NASA. *Directory Interchange Format (DIF) Standard*. 2019. URL: https://earthdata.nasa.gov/user-resources/standards-and-references/directory-interchange-format-dif-standard. Acedido em 04-02-2019.

[88] NASA. *What is a SERF?* 2019. URL: https://gcmd.gsfc.nasa.gov/add/serfguide/whatisaserf.html. Acedido em 04-02-2019.

[89] A. M. Kathleen Baynes. *Unified Metadata Model (UMM)*. 2018. URL: https://earthdata.nasa.gov/about/science-system-description/eosdis-components/common-metadata-repository/unified-metadata-model-umm. Acedido em 03-02-2019.

[90] NASA. *UMM Scalability*. 2018. URL: https://cdn.earthdata.nasa.gov/conduit/upload/3969/UMM\%20model\%20benefits\%20-\%20final.png. Acedido em 03-02-2019.

[91] NASA. *UMM Schedules and Milestones*. 2018. URL: https://wiki.earthdata.nasa.gov/display/CMR/UMM+Schedules+and+Milestones. Acedido em 03-02-2019.

[92] NASA. *UMM Profiles Content*. 2018. URL: https://cdn.earthdata.nasa.gov/conduit/upload/4876/UMM%20Page%20-%20Table%203%20-%20Component%20model%20Metadata%20explanations%20and%20progress.png. Acedido em 03-02-2019.

[93] *EOS Land Viewer*. URL: https://eos.com/landviewer/. Acedido em 16-02-2019.

[94] ESA. *Sentinel Data Access Annual Report 2017*. 2018. URL: https://scihub.copernicus.eu/twiki/pub/SciHubWebPortal/AnnualReport2017/COPE-SERCO-RP-17-0186_-_Sentinel_Data_Access_Annual_Report_2017-Final_v1.4.1.pdf. Acedido em 17-02-2019.

[95] *IPSentinel*. URL: https://ipsentinel.pt/. Acedido em 11-02-2019.

[96] *Copernicus Open Access Hub*. URL: https://scihub.copernicus.eu/dhus/. Acedido em 12-02-2019.

[97] ESA. *Copernicus Open Access Hub User Guide*. URL: https://scihub.copernicus.eu/userguide/. Acedido em 13-02-2019.

[98] *OData - the best way to REST*. URL: https://www.odata.org/. Acedido em 11-09-2019.

[99] *Solr is the popular, blazing-fast, open source enterprise search platform built on Apache Lucene*. URL: https://lucene.apache.org/solr/. Acedido em 11-09-2019.

[100] *CREODIAS*. URL: http://www.creodias.eu/. Acedido em 13-02-2019.

[101] *ONDA-DIAS*. URL: http://www.onda-dias.eu/. Acedido em 13-02-2019.

[102] *Mundi*. URL: http://www.mundiwebservices.com/. Acedido em 13-02-2019.

[103] *WEkEO*. URL: http://www.wekeo.eu/. Acedido em 13-02-2019.

[104] *Sobloo*. URL: http://www.sobloo.eu/. Acedido em 13-02-2019.

[105] CREODIAS. *Data Offer - CREODIAS*. URL: https://creodias.eu/data-offer. Acedido em 14-02-2019.

[106] CREODIAS. *Data Access Interfaces - CREODIAS*. URL: https://creodias.eu/data-access-interfaces. Acedido em 14-02-2019.

[107] ONDA-DIAS. *Upcoming data - ONDA DIAS*. URL: https://www.onda-dias.eu/cms/data/catalogue/upcoming-data/. Acedido em 17-02-2019.

[108] G. Systems. *DIAS Metadata: Sentinel-1*. URL: https://www.gael-systems.com/dias-metadata-sentinel-1/. Acedido em 14-02-2019.

[109] *Earth Explorer*. URL: https://earthexplorer.usgs.gov/. Acedido em 16-02-2019.

[110] *API for the ESPA ordering system*. URL: https://github.com/USGS-EROS/espa-api. Acedido em 02-08-2019.

[111] FDGC. *Geospatial Metadata*. URL: https://www.fgdc.gov/csdgmgraphical/index.html. Acedido em 17-02-2019.

[112] *Earthdata Search*. URL: https://search.earthdata.nasa.gov/. Acedido em 18-02-2019.

[113] A. M. Kathleen Baynes. *The Common Metadata Repository: The Foundation of NASA's Earth Observation Data*. 2018. URL: https://earthdata.nasa.gov/the-common-metadata-repository. Acedido em 03-02-2019.

[114] NASA. *Common Metadata Repository (CMR)*. 2018. URL: https://earthdata.nasa.gov/about/science-system-description/eosdis-components/common-metadata-repository. Acedido em 04-02-2019.

[115] NASA. *ECHO Metadata Standard*. 2019. URL: https://earthdata.nasa.gov/user-resources/standards-and-references/echo-metadata-standard. Acedido em 04-02-2019.

[116] *A planetary-scale platform for Earth science data analysis*. URL: https://earthengine.google.com/. Acedido em 02-08-2019.

[117] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau e R. Moore. "Google Earth Engine: Planetary-scale geospatial analysis for everyone". Em: *Remote Sensing of Environment* 202 (2017). Big Remotely Sensed Data: tools, applications and experiences, pp. 18 –27. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2017.06.031. URL: http://www.sciencedirect.com/science/article/pii/S0034425717302900.

[118] *Earth Engine Google Editor*. URL: https://developers.google.com/earth-engine/playground. Acedido em 05-08-2019.

[119] C. Chambers, A. Raniwala, F. Perry, S. Adams, R. Henry, R. Bradshaw e Nathan. "FlumeJava: Easy, Efficient Data-Parallel Pipelines". Em: *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. 2 Penn Plaza, Suite 701 New York, NY 10121-0701, 2010, pp. 363–375. URL: http://dl.acm.org/citation.cfm?id=1806638.

[120] H. Gonzalez, A. Y. Halevy, C. S. Jensen, A. Langen, J. Madhavan, R. Shapley, W. Shen e J. Goldberg-Kidon. "Google Fusion Tables: Web-centered Data Management and Collaboration". Em: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*. SIGMOD '10. Indianapolis, Indiana, USA: ACM, 2010, pp. 1061–1066. ISBN: 978-1-4503-0032-2. DOI: 10.1145/1807167.1807286. URL: http://doi.acm.org/10.1145/1807167.1807286.

[121] *QGIS*. URL: https://qgis.org/. Acedido em 13-09-2019.

[122] *Balancing Workload Across Nodes with Akka*. 2012. URL: https://letitcrash.com/post/29044669086/balancing-workload-across-nodes-with-akka-2. Acedido em 16-08-2019.

[123] Lightbend. *Build powerful reactive, concurrent, and distributed applications more easily*. URL: https://akka.io/. Acedido em 08-08-2019.

[124] EUMETSTAT. *LSA SAF - Land Surface Analysis*. URL: https://landsaf.ipma.pt/en/. Acedido em 12-08-2019.

[125] EUMETSTAT. *Land Surface Temperature*. URL: https://landsaf.ipma.pt/en/products/land-surface-temperature/lst/. Acedido em 12-08-2019.

[126] ESA. *Sentinel 2 Acquisition Plans*. URL: https://sentinel.esa.int/web/sentinel/missions/sentinel-2/acquisition-plans. Acedido em 12-08-2019.

[127] *Apache Hadoop*. URL: https://hadoop.apache.org/.

[128] R. Lott. *Geographic information - Well-known text representation of coordinate reference systems*. 2013. URL: http://docs.opengeospatial.org/is/12-063r5/12-063r5.html. Acedido em 14-08-2019.

[129] *spring.io*. URL: https://spring.io/. Acedido em 16-08-2019.

[130] Lighbend. *Akka Distributed Workers with Scala Guide*. URL: https://developer.lightbend.com/guides/akka-distributed-workers-scala/index.html.

[131] *The most popular database for modern apps*. URL: https://www.mongodb.com/. Acedido em 16-08-2019.

[132] *PostGIS - Spatial and Geographic Objects for PostgreSQL*. URL: https://postgis.net/. Acedido em 16-08-2019.

[133] S. Agarwal e K. S. Rajan. "Performance analysis of MongoDB versus PostGIS/PostGreSQL databases for line intersection and point containment spatial queries". Em: *Spatial Information Research* 24.6 (2016), pp. 671–677. ISSN: 2366-3294. DOI: 10.1007/s41324-016-0059-1. URL: https://doi.org/10.1007/s41324-016-0059-1.

[134] *Manage massive amounts of data, fast, without losing sleep*. URL: http://cassandra.apache.org/. Acedido em 16-08-2019.

[135] *Akka Persistence Cassandra*. URL: https://doc.akka.io/docs/akka-persistence-cassandra/. Acedido em 16-08-2019.

[136] *Swarm mode overview*. URL: https://docs.docker.com/engine/swarm/. Acedido em 18-08-2019.

[137] R. Dua. *Docker Swarm Architecture*. 2015. URL: https://image.slidesharecdn.com/dockerswarmv1-150401123157-conversion-gate01/95/docker-swarm-introduction-3-638.jpg?cb=1427891574. Acedido em 18-08-2019.

[138] *JSON Schema*. URL: https://json-schema.org/. Acedido em 19-08-2019.

[139] Open311. *XML to JSON Conventions*. URL: http://wiki.open311.org/JSON_and_XML_Conversion/. Acedido em 19-08-2019.

[140] *API Development for Everyone*. URL: https://swagger.io/. Acedido em 18-08-2019.

[141] Spotify. *spotify/dockerfile-maven*. URL: https://github.com/spotify/dockerfile-maven. Acedido em 19-08-2019.

[142] B. Porter, J. v. Zyl e O. Lamy. *Welcome to Apache Maven*. URL: https://maven.apache.org/. Acedido em 19-08-2019.

[143] *Dockerfile reference*. URL: https://docs.docker.com/engine/reference/builder/. Acedido em 19-08-2019.

[144] *Docker Plugin*. URL: https://www.scala-sbt.org/sbt-native-packager/formats/docker.html. Acedido em 19-08-2019.

[145] *The interactive build tool*. URL: https://www.scala-sbt.org/. Acedido em 19-08-2019.

[146] *Docker Hub*. URL: https://hub.docker.com/. Acedido em 19-08-2019.

[147] I. Ruivo. "Deteção Remota de Parcelas Agrícolas". Por publicar. Tese de mestrado. FCT-UNL, 2019.

[148] *Search and download Copernicus Sentinel satellite images*. URL: https://github.com/sentinelsat/sentinelsat. Acedido em 26-08-2019.

[149] A. Neves. "Deteção Remota de Estruturas Permanentes". Por publicar. Tese de mestrado. FCT-UNL, 2019.

[150] T. Kemper, C. Corban, D. Ehrliche, A. Florczyk, S. M. Carneiro Freire, L. Maffenini, M. Melchiorri, P. Politis, M. Schiavina, M. Pesaresi, P. Soille e V. Syrris. "GHS built-up grid, derived from Landsat, multitemporal (1975, 1990, 2000, 2014)". Em: (2017).

[151] *ASF Data Search*. URL: https://search.asf.alaska.edu/. Acedido em 29-08-2019.

[152] R. Afonso. "Avaliação por Deteção Remota do Estado da Vegetação em Faixas de Gestão de Combustível de Incêndio". Por publicar. Tese de mestrado. FCT-UNL, 2019.

[153] J. Ferreira. "Estudo Sobre o Efeito de Estruturas Permanentes na Fusão de Imagens de Satélite". Por publicar. Tese de mestrado. FCT-UNL, 2019.

ANEXO I

ANEXOS

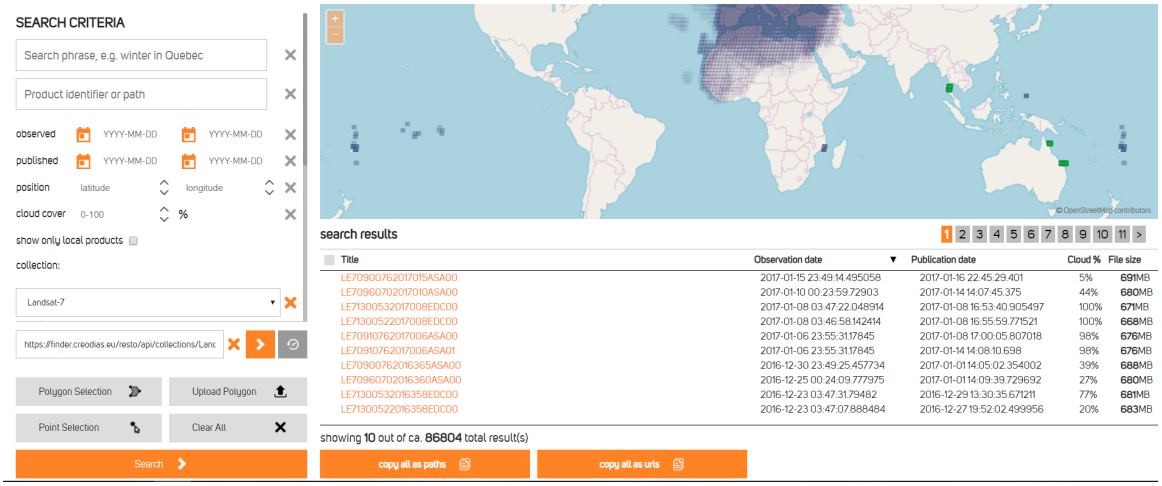


Figura I.1: Página inicial do Data Finder.

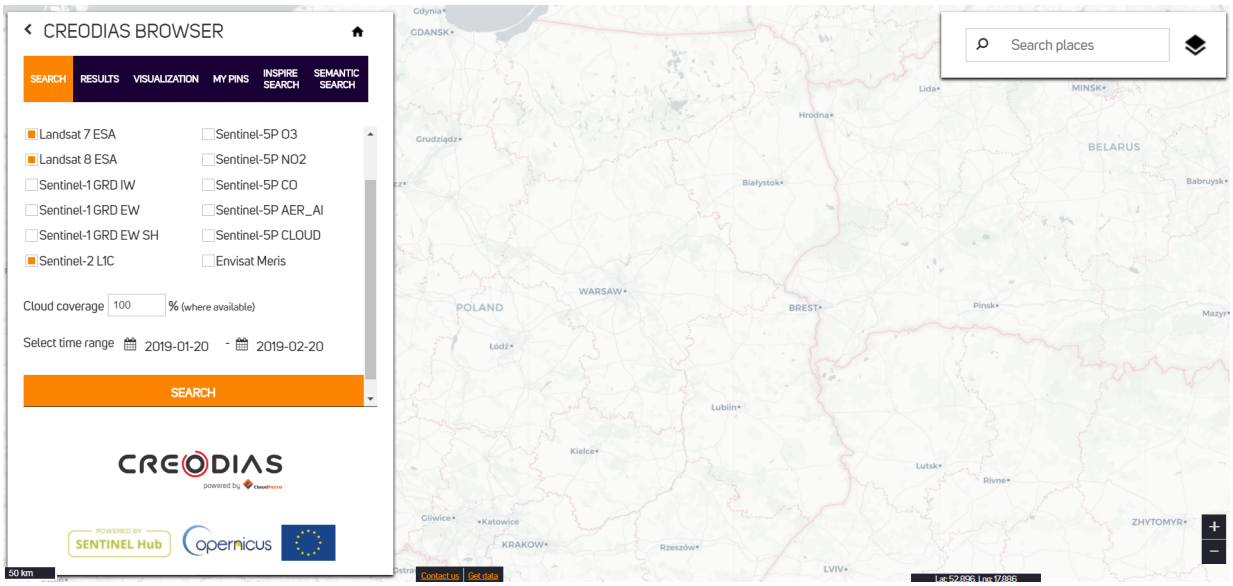


Figura I.2: Página inicial do CREODIAS Browser.

Figura I.3: Visualização de produtos no CREODIAS Browser.

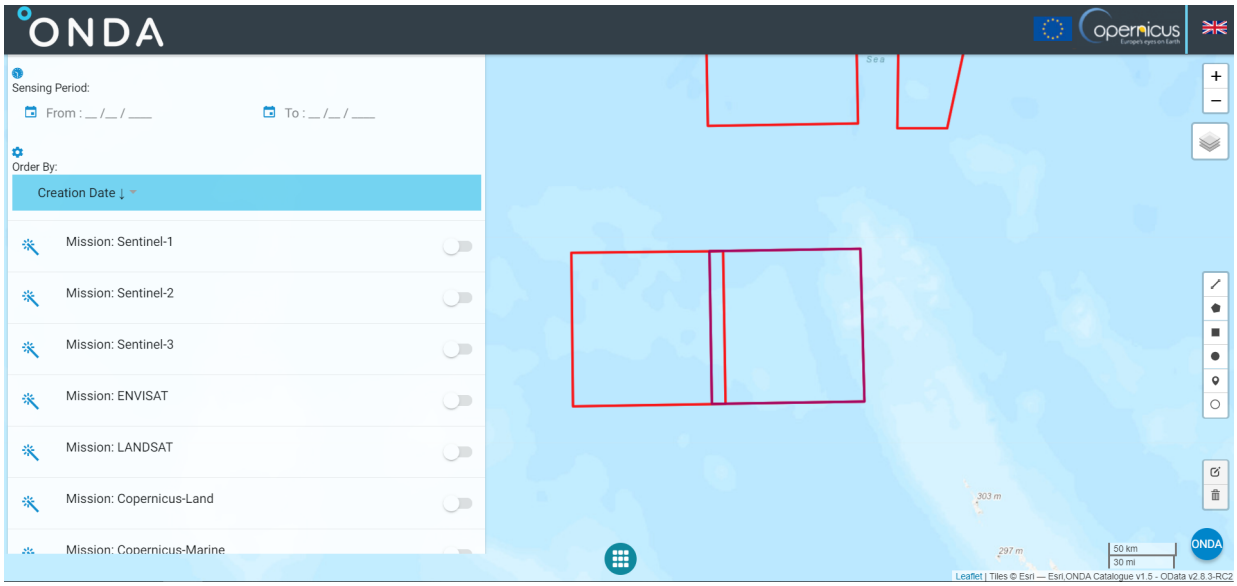


Figura I.4: Página inicial no catálogo Onda DIAS.

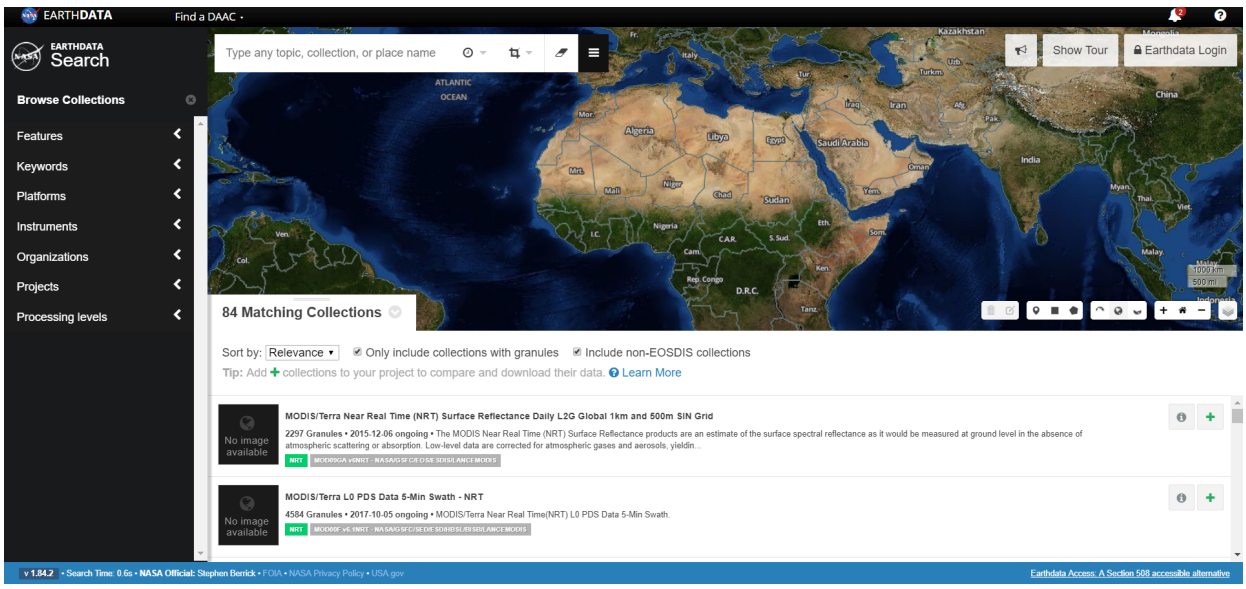


Figura I.5: Página inicial no catálogo Earthdata.

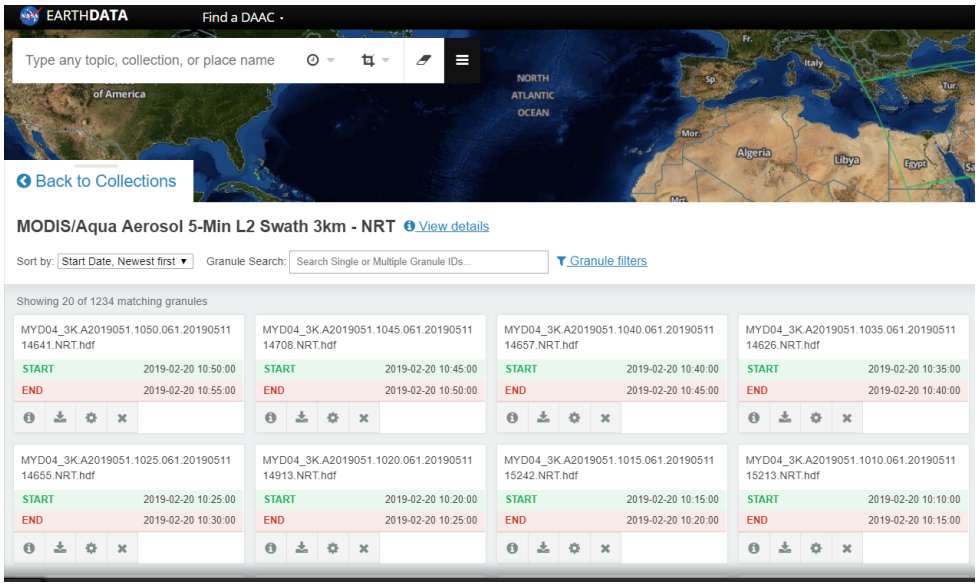


Figura I.6: Apresentação dos resultados no catálogo Earthdata.

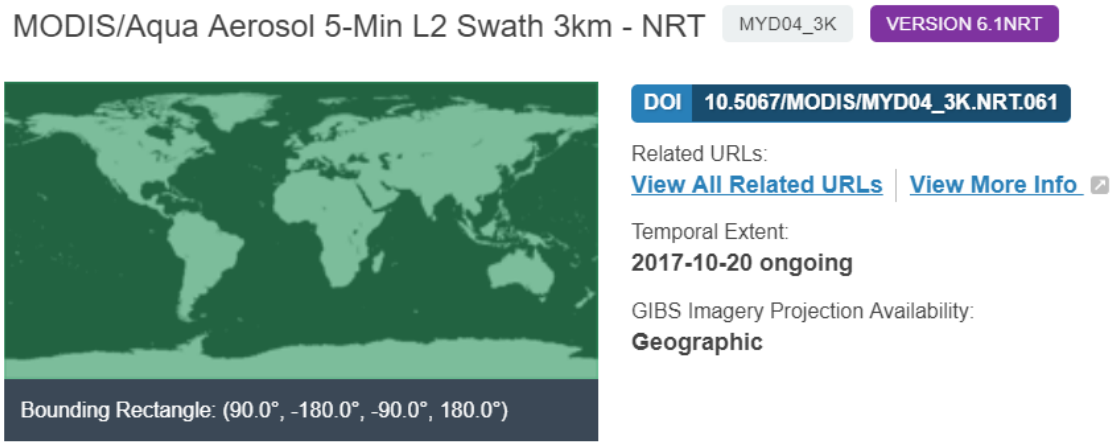


Figura I.7: Apresentação parcial dos metadados no catálogo Earthdata.

Dataset	Nominal resolution	Temporal granularity	Temporal coverage	Spatial coverage
Landsat				
Landsat 8 OLI/TIRS	30 m	16 day	2013–Now	Global
Landsat 7 ETM +	30 m	16 day	2000–Now	Global
Landsat 5 TM	30 m	16 day	1984–2012	Global
Landsat 4–8 surface reflectance	30 m	16 day	1984–Now	Global
Sentinel				
Sentinel 1 A/B ground range detected	10 m	6 day	2014–Now	Global
Sentinel 2A MSI	10/20 m	10 day	2015–Now	Global
MODIS				
MOD08 atmosphere	1°	Daily	2000–Now	Global
MOD09 surface reflectance	500 m	1 day/8 day	2000–Now	Global
MOD10 snow cover	500 m	1 day	2000–Now	Global
MOD11 temperature and emissivity	1000 m	1 day/8 day	2000–Now	Global
MCD12 Land cover	500 m	Annual	2000–Now	Global
MOD13 Vegetation indices	500/250 m	16 day	2000–Now	Global
MOD14 Thermal anomalies & fire	1000 m	8 day	2000–Now	Global
MCD15 Leaf area index/FPAR	500 m	4 day	2000–Now	Global
MOD17 Gross primary productivity	500 m	8 day	2000–Now	Global
MCD43 BRDF-adjusted reflectance	1000/500 m	8 day/16 day	2000–Now	Global
MOD44 veg. cover conversion	250 m	Annual	2000–Now	Global
MCD45 thermal anomalies and fire	500 m	30 day	2000–Now	Global
ASTER				
L1 T radiance	15/30/90 m	1 day	2000–Now	Global
Global emissivity	100 m	Once	2000–2010	Global
Other imagery				
PROBA-V top of canopy reflectance	100/300 m	2 day	2013–Now	Global
EO-1 hyperion hyperspectral radiance	30 m	Targeted	2001–Now	Global
DMSP-OLS nighttime lights	1 km	Annual	1992–2013	Global
USDA NAIP aerial imagery	1 m	Sub-annual	2003–2015	CONUS
Topography				
Shuttle Radar Topography Mission	30 m	Single	2000	60°N–54°S
USGS National Elevation Dataset	10 m	Single	Multiple	United States
USGS GMTED2010	7.5″	Single	Multiple	83°N–57°S
GTOPO30	30″	Single	Multiple	Global
ETOPO1	1′	Single	Multiple	Global
Landcover				
GlobCover	300 m	Non-periodic	2009	90°N–65°S
USGS National Landcover Database	30 m	Non-periodic	1992–2011	CONUS
UMD global forest change	30 m	Annual	2000–2014	80°N–57°S
JRC global surface water	30 m	Monthly	1984–2015	78°N–60°S
GLCF tree cover	30 m	5 year	2000–2010	Global
USDA NASS cropland data layer	30 m	Annual	1997–2015	CONUS
Weather, precipitation & atmosphere				
Global precipitation measurement	6′	3 h	2014–Now	Global
TRMM 3B42 precipitation	15′	3 h	1998–2015	50°N–50°S
CHIRPS precipitation	3′	5 day	1981–Now	50°N–50°S
NLDAS-2	7.5′	1 h	1979–Now	North America
GLDAS-2	15′	3 h	1948–2010	Global
NCEP reanalysis	2.5°	6 h	1948–Now	Global
ORNL DAYMET weather	1 km	Annual	1980–Now	North America
GRIDMET	4 km	1 day	1979–Now	CONUS
NCEP global forecast system	15′	6 h	2015–Now	Global
NCEP climate forecast system	12′	6 h	1979–Now	Global
WorldClim	30″	12 images	1960–1990	Global
NEX downscaled climate projections	1 km	1 day	1950–2099	North America
Population				
WorldPop	100 m	5 year	Multiple	2010–2015
GPWv4	30″	5 year	2000–2020	85°N–60°S

Figura I.8: Conjuntos de dados frequentemente acedidos no catálogo do GEE. [117]