Uma plataforma de exploração e análise de imagens médicas em nuvem

Igor J. Topcin

Dissertação apresentada
ao
Instituto de Matemática e Estatística
da
Universidade de São Paulo
para
obtenção do título
de
Mestre em Ciências

Programa: Ciência da Computação
Orientador: Prof. Dr. Marcel Parolin Jackowski

São Paulo, julho de 2016

Uma plataforma de exploração e análise de imagens médicas em nuvem

Esta é a versão original da dissertação elaborada pelo candidato Igor J. Topcin, tal como submetida à Comissão Julgadora.

Agradecimentos

Primeiramente, agradeço ao leitor pelo interesse neste trabalho, fruto de muito esforço e dedicação.

Também agradeço aos meus colegas do programa de mestrado do DCC, Rafael e Thiago. Meus agradecimentos também vão ao professor Marcel, pela objetividade, clareza e paciência durante os anos mais recentes da minha vida discente.

Não poderia deixar de agradecer aos meus pais, pessoas que admiro cada vez mais, responsáveis pelos alicerces do ser humano que sou hoje.

Por fim, mas não menos importante, agradeço à Maristela, minha esposa e companheira de estrada há mais de uma década. Sem seu suporte, paciência, dedicação e amor, este trabalho nunca teria sido possível.

Resumo

TOPCIN, I. J. **Uma plataforma de exploração e análise de imagens médicas em nuvem.** 2016. Dissertação de Mestrado - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2016.

O crescente uso de imagens médicas para pesquisas clínicas e biomédicas gera cada vez maiores quantidades de dados que precisam ser processados e analisados. Computadores individuais já não são suficientes para realizar esta tarefa em tempo hábil. Além disso, o vasto espectro de modalidades de imagens, o tamanho dos dados gerados, a natureza dos sinais inerente às imagens médicas tornam difícil tanto a concepção quanto o uso de ferramentas computacionais apropriadas para estudos em grande escala.

Diversas comunidades científicas, como é o caso da Bioinformática e da Astrofísica, têm adotado sistemas de gerenciamento de workflows para coordenação e execução de análises computacionais em grande escala. Tais sistemas se alicerçam na computação de alto desempenho, grades ou nuvens de computadores, redes de alta velocidade e dispositivos de armazenamento de alta capacidade. No entanto, a grande maioria dos sistemas existentes apresentam uma série de funcionalidades que visam atender os mais diversos cenários de uso, mas poucos atendem às especificidades da análise de imagens médicas.

Este trabalho descreve o desenvolvimento e a arquitetura de uma solução para exploração e análise de imagens médicas que responda à demanda de processamento de grandes quantidades de dados. Em síntese, objetivamos: i) identificar os cenários típicos de uso de workflows para imagens médicas em pesquisas científicas; ii) analisar opções de sistemas de gerenciamento de workflows existentes e sua aplicabilidade em imagens médicas; iii) apresentar uma solução baseada em nuvem para promover a exploração e análise de imagens de forma acessível, reprodutível, transparente e escalável; e iv) realizar uma análise do desempenho da solução proposta.

A construção de tal plataforma poderá contribuir para realização de análises estatísticas mais abrangentes entre grupos populacionais. Os resultados obtidos poderão ser usados para o desenvolvimento de programas de prevenção de doenças efetivos e viáveis, na melhoria de ferramentas diagnósticas e em políticas de saúde que causem um impacto positivo na sociedade.

Palavras-chave: workflows científicos, imagens médicas, computação em nuvem

Abstract

TOPCIN, I. J. **A platform for exploration and analysis of medical imaging in the cloud.** 2016. Master's Thesis - Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2016.

The increasing usage of medical imaging in clinical and biomedical research contributes to an ever-larger number of datasets to be processed and analyzed. Personal computers are not able to perform this task in a timely manner any longer. Moreover, it is difficult to conceive and use computational tools for large scale studies due to the vast spectrum of medical imaging modalities, the size of generated datasets and the nature of the signal inherent to medical imaging.

Many scientific communities, such as Bioinformatics and Astrophysics, have been adopting workflow management systems in order to coordinate large scale computational analysis. These systems are based on high performance computing, grid and cloud computing, high speed networks and high performance storage devices. Although most of the existing systems present a wide range of functionalities designed to meet most use case scenarios, only few of them adhere to the requirements of medical imaging analysis.

This work describes the development and architecture of a solution for medical imaging exploration and analysis able to handle large volumes of data. In summary, our main goals are: i) identifying typical workflow usage scenarios in medical imaging research; ii) analyzing existing workflow management systems and their applicability for medical imaging; iii) proposing a cloud based solution that enables accessible, reproducible, transparent and scalable medical imaging analysis and exploration; and iv) running a simple performance analysis for the proposed solution.

The aforementioned platform can leverage broader statistical group analysis and studies. The results of this work may be used to: i) develop effective and viable disease prevention programs; ii) improve diagnosis tools; and iii) guide the creation of health policies that can cause a positive impact on society.

Keywords: scientific workflows, medical imaging, cloud computing

Conteúdo

Lista de Abreviaturas

MRI	Aquisição de Imagem por Ressonância Magnética (*Magnetic Resonance Imaging*)
fMRI	MRI funcional (*Functional Magnetic Resonance Imaging*)
DTI	MRI por Tensor de Difusão (*Diffusion Tensor Imaging*)
CT	Tomografia Computadorizada (*Computed Tomography*)
PET	Tomografia por Emissão de Pósitrons (*Positron Emission Tomography*)
PET-CT	Integração das imagens de PET com CT
DICOM	Comunicação de Imagens Digitais em Medicina (*Digital Imaging and Communications in Medicine*)
PACS	Sistema de Comunicação e Arquivamento de Imagens (*Picture Archiving and Communication System*)
RIS	Sistema de Informação para Radiologia (*Radiology Information System*)
CAD	Sistemas de Auxílio ao Diagnóstico (*Computer-Aided Diagnosis*)
WfMS	Sistema de Gerenciamento de Workflows (*Workflow Management System*)
BPMS	Sistema de Gerenciamento de Processos de Negócio (*Business Process Management System*)
DRMS	Sistema de Gerenciamento de Recursos Distribuídos (*Distributed Resources Management System*)
GUI	Interface Gráfica de Usuário (Graphical User Interface)
CLI	Interface por Linha de Comando (*Command Line Interface*)
API	*Application Programming Interface*
DRMAA	*Distributed Resource Management Application API*
PBS	*Portable Batch System*
TORQUE	*Terascale Open-source Resource and QUEue Manager*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O uso de equipamentos geradores de imagens médicas proporcionou avanços significativos nos campos da medicina diagnóstica e biomedicina. A evolução constante da tecnologia empregada nesses equipamentos é acompanhada do aumento da resolução, complexidade e do volume dessas imagens. Somente na Fundação Instituto de Pesquisa e Estudos de Diagnóstico por Imagem, ou FIDI, foram realizados mais de 4 milhões de exames radiológicos em 2013[1]. Torna-se então necessário o uso de ferramentas e metodologias de processamento e análise de imagens robustas e eficientes para auxiliar o radiologista no diagnóstico do paciente e o cientista a desenvolver pesquisas biomédicas.

A aquisição de imagens médicas de um único paciente pode gerar centenas ou até milhares de imagens tomográficas, como é o caso das modalidades de ressonância magnética (MRI), tomografia computadorizada (CT), tomografia por emissão de pósitrons (PET) e sua combinação com a tomografia computadorizada (PET-CT). Esse grande volume de dados se traduz em alguns grandes desafios para hospitais, laboratórios e centros de pesquisa:

- Como lidar com o vasto espectro de modalidades de imagens e a complexidade inerente a sua natureza?
- Como armazenar, organizar e identificar de forma eficiente as aquisições de imagens de todos os pacientes?
- Como processar de forma robusta e automatizada tamanho volume de dados?
- Como os radiologistas e cientistas acessam as imagens de pacientes?
- Como compartilhar dados, diagnósticos, metodologias e conhecimento?

Existem diversas soluções proprietárias e abertas que respondem, mesmo que parcialmente, essas questões. Por exemplo:

- O padrão DICOM (*Digital Imaging and Communications in Medicine*) surgiu para facilitar a distribuição e visualização de imagens médicas. Hoje em dia, praticamente todos os equipamentos médicos modernos são capazes de transmitir e receber imagens neste padrão.
- Implementações de PACS (*Picture Archiving and Communication System*) e RIS (*Radiology Information System*) surgiram para organizar o acervo de imagens médicas e distribuí-las aos computadores dos radiologistas para realização do diagnóstico, respectivamente.
- Os CADs (*Computer-Aided Diagnosis*) surgiram para auxiliar os radiologistas na tomada de decisão a respeito do diagnóstico do paciente.

[1] http://bit.ly/1uaeLCf

No entanto, parte dessas soluções atendem apenas os requisitos do workflow diagnóstico em clínicas e hospitais, ou seja, não são adequadas para o ambiente de pesquisa científica, tampouco para o processamento de imagens em grandes quantidades ou análises estatísticas de grupos populacionais. Outras partes podem ser reaproveitadas, como veremos nos capítulos a seguir.

Nota-se que muitas das dificuldades acima mencionadas também existem em pesquisas científicas em outras áreas da ciência, como é o caso da astronomia, biomedicina e neurociência. Para superá-las, a computação se tornou essencial, fazendo surgir uma nova vertente computacional em pesquisas, como descrito por Braghetto e Cordeiro (2014). Na neurociência, por exemplo, a análise de imagens funcionais e estruturais do cérebro depende de ferramentas computacionais cada vez mais sofisticadas, capazes de realizar operações como segmentação, reconstrução tridimensional, visualização de volumes e cálculos estatísticos.

Entretanto, há um efeito colateral decorrente dessa dependência: as ferramentas e recursos computacionais estão se tornando cada vez mais complexos e de difícil uso. Cientistas precisam ter habilidades em informática e programação para usar e combinar diversas ferramentas de análise e assim conduzir seus experimentos. Qualificar cientistas no uso dessas ferramentas tem sido uma das estratégias para atacar esse problema. O FMRIB (*Centre for Functional MRI of the Brain*), da universidade de Oxford, oferece material e cursos para dois dos principais softwares para análises de imagens do cérebro: *FSL* e *FreeSurfer* (Courses, b). O *Martinos Center for Biomedical Imaging*, da universidade de Harvard, também oferece cursos e tutoriais para *FreeSurfer* (Courses, a), software por eles desenvolvido.

Em pesquisas computacionais, há ainda o problema da reprodutibilidade: um dado experimento precisa ser documentado de forma clara e objetiva, a fim de permitir que outros cientistas possam reproduzi-lo. A partir dos dados originais, é necessário saber quais transformações foram realizadas e em que ordem. Algumas ferramentas já incorporam meios de se gravar esse tipo de metainformação, como é o caso do pacote AFNI (Cox, 2012), que reúne programas para análise e visualização de imagens por ressonância magnética funcional (fMRI), utiliza um formato proprietário de dados que armazena por quais transformações uma dada imagem passou.

Outros trabalhos propõem resolver o problema da reprodutibilidade tratando experimentos computacionais como uma sequência de operações de transformação de dados. Para compor o experimento, o cientista descreve o fluxo de sua análise de forma estruturada, determinando quais são as passos e em qual ordem eles acontecem. Chama-se essa estrutura de *pipeline* ou *workflow* científico, que pode ser descrito programaticamente em um *script shell*, em XML ou em uma linguagem de domínio específico (em inglês, *Domain Specific Language*, ou simplesmente DSL). No projeto Pegasus (Deelman *et al.*, 2014), por exemplo, workflows são expressos como grafos acíclicos (ou DAG, do inglês *directed acyclic graph*), onde os nós e as arestas representam as tarefas computacionais e suas dependências, respectivamente.

Podemos ainda citar a necessidade de se compartilhar os dados originais, metodologia e ferramentas usadas em um experimento. Dessa forma, as publicações se tornam mais ricas e transparentes, pois oferecem subsídios para outros cientistas obterem os mesmos resultados apresentados no trabalho original. Soluções para esse problema já foram propostas por Goecks *et al.* (2010) e De Roure *et al.* (2009). Ambos propõem métodos para colaboração entre comunidades científicas e publicação e compartilhamento de workflows e experimentos.

Adiciona-se a esse cenário a quantidade imensa de dados e imagens adquiridas em hospitais, clínicas e instituições de pesquisa. A ciência de hoje lida com volumes cada vez maiores de dados, descrito por Hey e Trefethen (2003) como dilúvio de dados. Em *Preserved white matter in unmedicated pediatric bipolar disorder* (Teixeira *et al.*, 2014), por exemplo, usa-se um *workflow* definido como um *script shell* que realiza análises estatísticas a partir de imagens médicas de uma população composta por dezenas de indivíduos. A execução dessas análises pode levar dias se processadas por um único computador.

Felizmente, o surgimento das plataformas de computação de alto desempenho, como aglomerados (*clusters*), grades (*grids*) e nuvens de computadores, nos coloca à disposição uma quantidade virtualmente ilimitada de recursos computacionais para a execução de nossos workflows científi-

cos. A computação em nuvem, por exemplo, permite o aprovisionamento sob demanda de recursos como servidores, armazenamento de dados, redes, dentre outros. Naturalmente, custos recaem sobre o uso desses recursos, mas o modelo de computação em nuvem nos permite alugar apenas os recursos necessários, conforme a demanda por processamento, memória, armazenamento, etc.

1.1 Motivação

Análise e processamento de imagens médicas tem sido um grande desafio para instituições como hospitais e centros de pesquisa. Nesse campo, assim como em outros campos das ciências da vida, ferramentas e recursos computacionais cada vez mais complexos tem se tornado fundamentais para que cientistas realizem suas análises e experimentos. Além disso, muito embora seja crescente a demanda por capacidade de processamento e armazenamento de imagens e dados de pacientes, as instituições estão sendo pressionadas para diminuir seus custos. É necessário então estudar alternativas que sejam viáveis economicamente e escaláveis, ou seja, que comportem esse crescimento da demanda sem ultrapassar orçamentos.

A principal motivação deste trabalho é produto de observações que realizamos no CHU (*Centre Hospitalier Régional Universitaire*) de Nîmes, na França[2], que conduz pesquisas em diversas áreas da medicina. Pesquisas na área de imagens médicas no CHU, por exemplo, tipicalmente envolvem aquisições de dados de diversas modalidades de imagens médicas, como MRI, CT e PET-CT.

Para que o cientista possa conduzir seu experimento no CHU, ele precisa obter e armazenar as imagens adquiridas pelos equipamentos médicos. Em alguns casos, as aquisições são transmitidas ao sistema de arquivamento de imagens (PACS) da instituição, para depois serem recuperadas pelo cientista. Outras vezes o processo de transmissão dessas imagens é manual, por exemplo através de CDs gravados diretamente da estação conectada ao equipamento gerador de imagens médicas. Na primeira opção, o mesmo PACS abrigará tanto acervo de imagens utilizadas em pesquisa, quanto àquelas da rotina de diagnóstico clínico de pacientes reais. Neste caso, pesquisa e rotina clínica competirão pelos mesmos recursos. Já a segunda opção traz consigo alguns problemas: os dados armazenados em CDs podem ser perdidos, seja por problemas no processo de gravação da mídia, ou por simples extravio dos CDs. Além disso, torna-se difícil manter um catálogo unificado com o histórico das imagens de pacientes que foram geradas para fins de pesquisa, pois cada cientista organiza seu próprio catálogo, seja em um dispositivo de armazenamento externo, ou em um sistema de arquivos compartilhado na rede de computadores da instituição em que trabalha.

Uma vez em posse do cientista, as imagens e dados são analisados com ferramentas instaladas em seu próprio computador, ou seja, o cientista está limitado à capacidade de processamento de uma única máquina. Fazer ciência nesses moldes pode ser uma boa opção quando se tem poucos dados para analisar e processar, entretanto o paradigma da ciência mudou nas últimas décadas, e o volume e complexidade dos dados analisados cresceu, como descreve Hey e Trefethen (2003). Em imagens de tensor de difusão, por exemplo, uma única aquisição resulta em uma série de volumes em 3 dimensões, um para cada gradiente de difusão utilizado, além do volume referente à imagem anatômica da parte do corpo estudada. A análise desse tipo de aquisição é complexa e em muitos casos demanda mais poder computacional que uma única estação de trabalho pode suprir.

Em um experimento típico, as atividades de processamento de dados são realizadas através da combinação de ferramentas computacionais, dando origem a uma sequência bem definida de atividades. Chama-se essa sequência de *pipeline* ou workflow, que em muitos casos é implementado como *scripts shell*. O papel desses *scripts* é coordenar a execução das atividades na ordem correta, controlando os dados de entrada e a passagem dos resultados intermediários de uma ferramenta para a outra. Uma atividade pode ser um programa ou até mesmo um outro *script* contendo comandos para outros softwares, como Matlab. Portanto, para compor um workflow usando essa abordagem, o cientista precisa ter habilidades em programação ou a ajuda de um programador. No primeiro caso, temos um problema: o currículo dos cursos relacionados às ciências da vida não contempla disciplinas como Linguagens de Programação e Engenharia de Software. No segundo caso, não é

[2] http://www.chu-nimes.fr

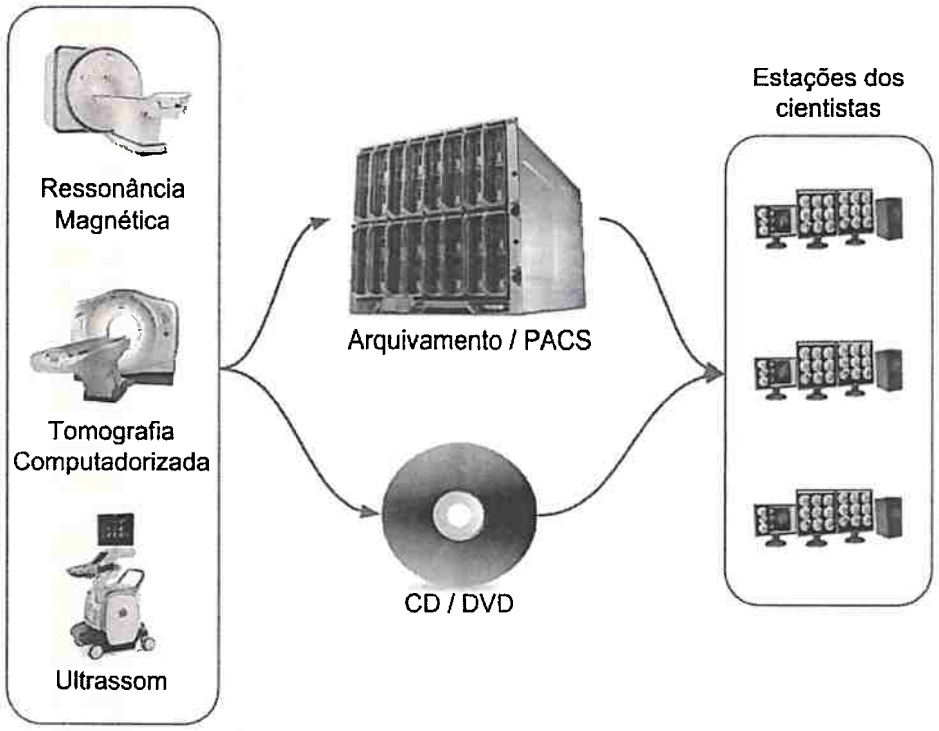

Figura 1.1: *Duas formas de se obter e armazenar as imagens adquiridas pelos equipamentos médicos. A primeira mostra o fluxo de imagens chegando às estações de trabalho dos cientistas através de um PACS. A segunda mostra o mesmo fluxo, porém as imagens são transmitidas via CDs.*

verdade que todo cientista tem a sua disposição um programador sempre que for preciso escrever um novo workflow, ou alterar um existente.

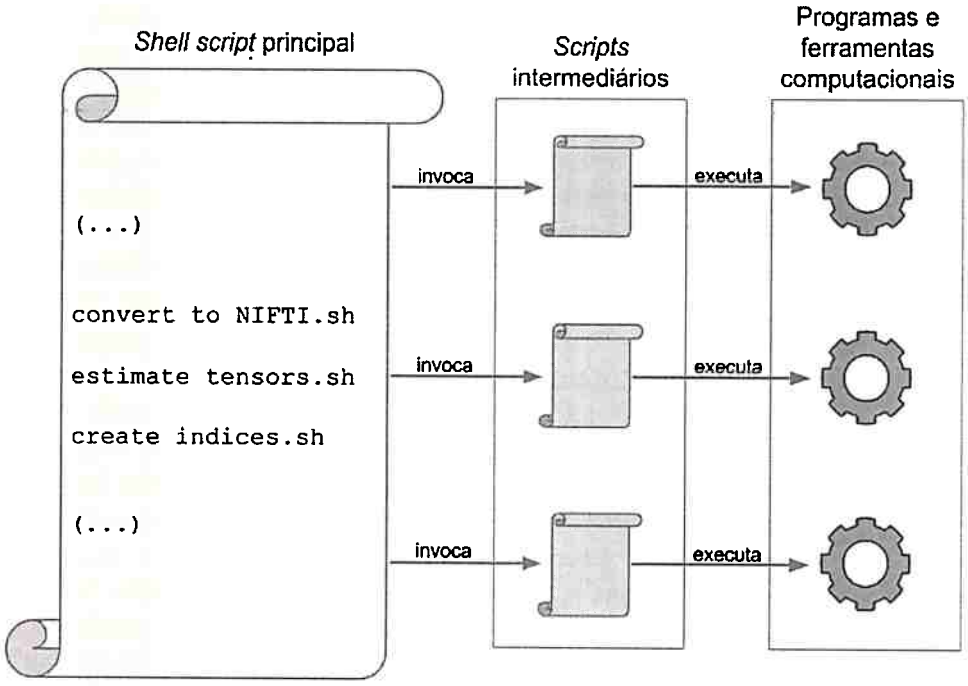

Figura 1.2: *Scripts shell sendo utilizados para controlar a sequência de atividades em um experimento.*

Nesse cenário, cada ferramenta computacional utilizada em um experimento precisa ser insta-

lada, configurada e mantida atualizada localmente, ou seja, no computador do cientista. Além de consumir muito tempo, essa prática acarreta em um problema mais sério: a falta de padronização, que por sua vez culmina no problema da reprodutibilidade, condição fundamental que discerne um estudo científico.

Os resultados das análises e experimentos realizados pelo cientista são armazenados localmente ou em um sistema de arquivos compartilhado na rede da instituição. A primeira opção é a mais preocupante, pois os dados podem ser perdidos se o cientista não realizar *backups* frequentes dos resultados armazenados em seu computador. A segunda opção é um grande avanço, mas a organização dos arquivos na rede ainda está sujeita às preferências pessoais de cada cientista.

Finalmente, a falta de padronização da metodologia de armazenamento e das ferramentas de visualização e processamento de imagens médicas dificultam o uso destas em grande escala, principalmente na rotina clínica de hospitais e laboratórios.

1.2 Objetivos

Este trabalho propõe o desenvolvimento de uma solução para exploração e análise de imagens médicas que responda à demanda de processamento de grandes quantidades de dados. Objetivamos identificar os cenários típicos de uso de workflows de análise de neuroimagem, e apresentar uma solução de software baseado em nuvem para promover a exploração e análise de imagens de forma acessível, reprodutível, transparente e escalável. Como estudo de caso, selecionaremos alguns dos workflows que identificamos para implementá-los a fim de validar nossa proposta.

A solução deve funcionar como uma bancada virtual em que os usuários realizam análises através de ferramentas computacionais. As imagens adquiridas pelos equipamentos médicos serão carregados na plataforma para que se possa usá-las em análises e experimentos científicos. As ferramentas computacionais estarão disponíveis na plataforma, eliminando a necessidade de instalá-las no computador do usuário. As análises também serão mais fáceis de se reproduzir, pois cada passo que as compõe é gravado como um workflow, que poderá ser reexecutado posteriormente e publicado para que outros usuários possam obter os mesmos resultados. Por último, um workflow poderá processar uma única aquisição de imagem ou centenas delas, usando uma infraestrutura de computação em nuvem para escalar recursos computacionais conforme a necessidade

1.3 Trabalhos Relacionados

Trabalhos na área de workflows científicos têm resultado na construção de uma vasta variedade de sistemas de gerenciamento de workflows (WfMS, do inglês *Workflow Management System*), cada qual com características específicas e necessárias para um ou outro campo da ciência. Tais sistemas possuem propriedades comuns, como:

- capacidade de gerir recursos computacionais distribuídos;
- capacidade de monitorar, controlar e distribuir operações computacionais para que sejam executadas nos recursos existentes;
- resiliência a falhas durante a execução dos passos de um workflow;
- interação entre os diversos tipos de plataformas (*e.g. web services*, aglomerados e grades computacionais);
- interfaces de usuário que facilitam a criação de workflows científicos; dentre outras.

Conforme descrito por Olabarriaga *et al.* (2014), quando pensamos em usuários deste tipo de sistema, geralmente temos a imagem de um cientista cujo interesse é pela cadeia completa de gerenciamento de workflows. Na prática, entretanto, esse não é o caso de muitas das comunidades científicas usuárias de WfMS.

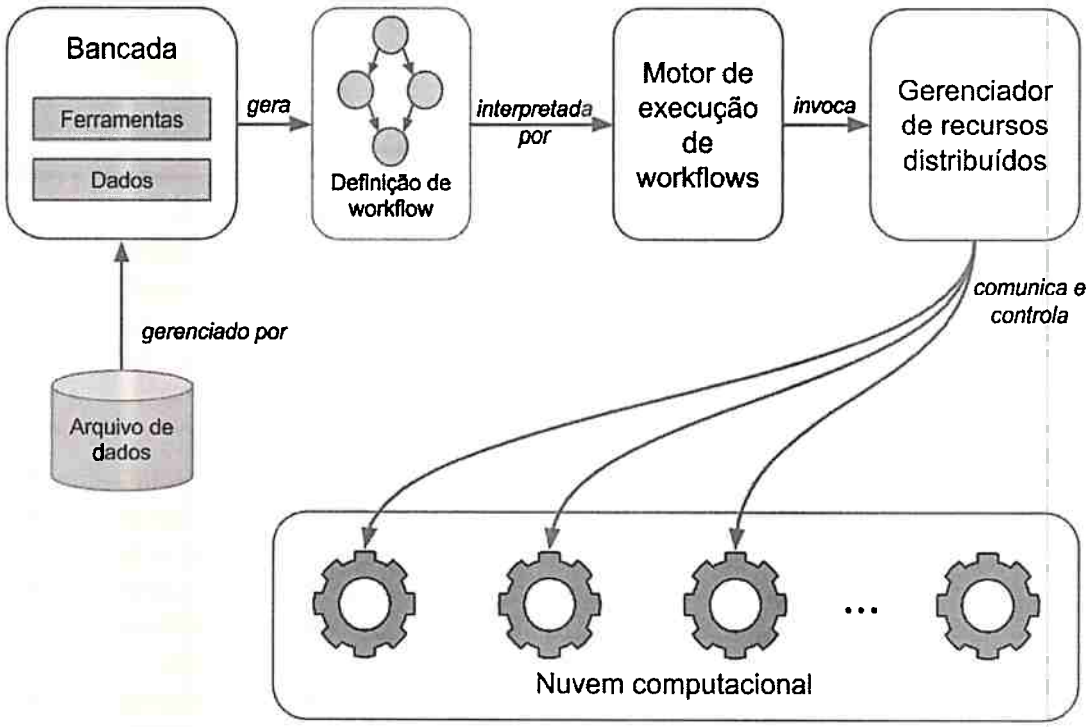

Figura 1.3: *Plataforma para exploração de imagens médicas, mostrando seus diversos componentes e a interação entre eles.*

Alguns trabalhos anteriores a este já propuseram soluções para resolver um ou outro problema isolado. No Projeto Galaxy (Goecks *et al.*, 2010), os autores propõem uma plataforma *web* para pesquisa genômica, mas que também vem ganhando força em outros domínios da bioinformática. Por ser extensível, ferramentas computacionais podem ser incorporadas ao Galaxy, ficando assim disponíveis aos cientistas. Esse é o caso do EMBOSS (Rice *et al.*, 2014), sigla para *The European Molecular Biology Open Software Suite*, um conjunto de ferramentas para biologia molecular disponível para uso no servidor público do *Centre de Génomique Fonctionnelle de Bordeaux* (CGFB). Mais recentemente, Afgan *et al.* (2011) publicaram uma solução que facilita o uso do Galaxy em plataformas de computação em nuvem, como OpenStack e OpenNebula (Moreno-Vozmediano *et al.*, 2012), além de provedores de infraestrutura, plataforma e software como serviço, como a Amazon AWS , Rackspace , dentre outros.

Outro exemplo é o trabalho de Oakley *et al.* (2014). Os autores adaptaram para a plataforma Galaxy uma série de ferramentas computacionais para filogenia. Como resultado, foi criada uma fundação para o desenvolvimento de novas ferramentas para pesquisas filogenéticas.

Os trabalhos citados acima propõem sistemas que promovem a pesquisa computacional acessível, reprodutível e transparente. No entanto, suas aplicações são em campos diferentes do foco desta dissertação, ou seja, processamento e análise de imagens médicas. Neste campo, Wang *et al.* (2013) desenvolveram uma plataforma que integra: (i) o gerenciador de workflows Galaxy; (ii) Hadoop, um framework para processamento de grandes quantidades de dados em ambientes de computação distribuída; e (iii) um conjunto de ferramentas proprietárias para processamento de imagens desenvolvido anteriormente pela agência australiana CSIRO . Detalhes desse projeto também podem ser encontrados em *Galaxy + Hadoop: Toward a Collaborative and Scalable Image Processing Toolbox in Cloud* (Chen *et al.*, 2014). O projeto, no entanto, abrange a integração de ferramentas proprietárias desenvolvidas pela agência, e não compreende o processo de aquisição das imagens produzidas pelos equipamentos médicos.

O projeto LONI Pipeline (Dinov, 2009) também surgiu com o objetivo de facilitar a pesquisa computacional, porém seu foco é em neurociência. A solução proposta pelos autores é composta de um sistema distribuído para execução de workflows, que agrupa ferramentas computacionais comuns na neurociência, e um software cliente capaz de: (i) abstrair a complexidade do uso destas ferramentas e (ii) eliminar a necessidade de *scripts* para coordenar o encadeamento de atividades de análise. Além disso, o LONI Pipeline possibilita a armazenagem e compartilhamento de dados e resultados de análises. Entretanto, grande parte do código fonte é fechado ou distribuído com licenças proprietárias, o que limita as possibilidades de extensão ou investigação de suas funcionalidades e mecanismos internos.

Em *Exploring Workflow Interoperability for Neuroimage Analysis on the SHIWA Platform*, Korkhov *et al.* (2013) discutem a utilização de uma plataforma elaborada pelo projeto SHIWA (Workflow) que facilita o compartilhamento de workflows e sua interoperabilidade. Em linhas gerais, a plataforma SHIWA possui duas abordagens para permitir que diferentes representações de workflows coexistam. A primeira abordagem é a de que um workflow pode ser uma composição de sub-workflows heterogêneos, ou seja, descritos em linguagens diferentes. Já a segunda abordagem permite que um workflow descrito em uma linguagem específica possa ser transformado e posteriormente executado em diferentes sistemas de workflows, como Galaxy, Pegasus, Kepler (Altintas *et al.*, 2004), LONI Pipeline (Dinov, 2009), dentre outros. Propondo a colaboração entre pesquisadores por meio do compartilhamento de workflows, e também a interoperabilidade entre diferentes sistemas de workflows, o projeto SHIWA traz importantes ganhos para a ciência. Entretanto, a plataforma não contempla um método para a composição de workflows científicos. Para compô-los ainda é preciso utilizar um dos sistemas de workflows contemplados pelo projeto SHIWA, como o Kepler e Galaxy, e só então é possível carregá-los na plataforma SHIWA.

Garijo *et al.* (2014) elaboraram um estudo sobre os benefícios do reuso de workflows aplicados em pesquisas de Neurociência envolvendo imagens. Os laboratórios analisados usavam o LONI Pipeline como WfMS. Os autores concluem que os usuários de WfMS estudados percebem claramente os benefícios do reuso de workflows em suas pesquisas e sugerem algumas áreas que ainda devem ser exploradas e que incorporamos ao nosso trabalho, como:

- melhoria na documentação de workflows, facilitando assim o seu uso;
- associação de workflows e seus passos a publicações científicas;
- publicação de workflows em conjunto com os artigos, dissertações ou teses que os originaram.

Além disso, este trabalho considera os cenários de uso dos perfis de usuários de sistemas de workflows científicos, conforme proposto inicialmente por Olabarriaga *et al.* (2014) e Cerezo (2013): especialistas no domínio, especialistas na plataforma e especialistas em workflows. Expandiremos os conceitos apresentados, relacionando-os aos casos de uso que observamos no CHU de Nîmes.

1.4 Contribuições

As principais contribuições deste trabalho são:

- identificar as principais propriedades de uma solução para exploração e análise de imagens médicas que responda à demanda de processamento de grandes quantidades de dados;
- abstrair a complexidade da infraestrutura computacional que torna possível a solução proposta;
- apresentar um arcabouço extensível de ferramentas para realização de análises estatísticas entre grupos populacionais;
- propor uma solução que promova a exploração e análise de imagens de forma acessível, reprodutível, transparente e escalável.

1.5 Organização do Trabalho

No capítulo 2, introduziremos os conceitos que fundamentam este trabalho, tais como: (i) a utilização de workflows na ciência, indústria e medicina; (ii) submissão e controle distribuído de tarefas computacionais; (iii) computação em nuvem; e (iv) aquisição de imagens médicas. No capítulo 3 apresentamos como utilizamos estes conceitos na construção de uma plataforma para exploração de imagens médicas. No capítulo 4, apresentamos os experimentos que realizamos nesta plataforma, os resultados encontrados e os produtos desta pesquisa. Finalmente, no capítulo 5, sumarizamos as principais conclusões e contribuições deste trabalho, discutiremos suas limitações e finalizaremos com algumas sugestões para trabalhos futuros.

Capítulo 2

Conceitos

Neste capítulo apresentaremos os conceitos que alicerçam nossa proposta de plataforma para exploração e análise de imagens médicas em nuvem. Exploraremos i) as características de sistemas de gerenciamento de workflows, ii) a organização da infraestrutura computacional que torna a plataforma possível, iii) técnicas utilizadas para o escalonamento das atividades que compõem um workflow, e iv) um estudo de caso de workflow para processamento de imagens médicas.

2.1 Workflows

Um workflow, segundo o *Workflow Management Coalition* (WfMC), "envolve a automação de procedimentos, onde documentos, informações ou atividades são passadas de um participante a outro de acordo com um conjunto de regras para atingir ou contribuir para um objetivo global de negócio" e é definido como "a automação de um processo de negócio, facilitada computacionalmente, no todo ou em parte" (Hollingsworth, 1993). Essa definição, que chamaremos aqui de **workflows de negócio**, é adequada para o cenário corporativo, onde as preocupações principais são a passagem de informações entre os participantes de um processo de negócio e todas as regras de negócio que regem esse processo.

Aalst e Hee (2002) sugere que a história dos sistemas de workflows começou na década de 1970. Skip Ellis *et al.* trabalharam no projeto *Office Automation Systems*, na Xerox PARC, e já usavam um modelo de workflows baseado em redes de Petri, na época denominado *Information Control Nets*. Zisman (1977) propôs em sua tese de doutorado, intitulada "*Representation, specification, and automation of office procedures*", a ideia de ferramentas e métodos genéricos como alicerce para processos de negócio. Entretanto, somente na década de 1990 que as redes de computadores corporativas se tornaram ubíquas, fato que tornou possível colocar em prática os conceitos de workflows de negócio inicialmente propostos duas décadas antes.

Um sistema de gerenciamento de workflows (WfMS) de negócio controla e gerencia os processos de negócio que ocorrem em uma organização, conectando e orquestrando atividades entre diversos recursos corporativos, como funcionários e sistemas. Em termos práticos, em uma organização cujos processos de negócio são descritos por workflows, percebe-se que quanto mais desacopladas são as funcionalidades de suas aplicações, melhor é a eficiência de seus workflows. Ou seja, componentes de software com escopos bem definidos podem ser facilmente configurados para compor workflows de negócio (Aalst e Hee, 2002). Além disso, esses componentes podem ser reusados em diversos workflows da organização. Em contrapartida, sistemas de informação cujas funções estejam fortemente acopladas são de difícil utilização em workflows.

Sistemas de workflows foram também beneficiados pela evolução da computação distribuída. A arquitetura cliente-servidor era dominante nos primeiros WfMS, mas estes apresentavam problemas de desempenho, confiabilidade e escalabilidade, como descreve Liu *et al.* (2012). Essa arquitetura foi substituída por infraestruturas ponto a ponto, e mais tarde pela computação em grade. O motor de execução (i.e. *enactment engine*) dos sistemas de workflows pôde tirar proveito de arcabouços

de computação em grade, como o Globus Toolkit[1]. Este é o caso de muitos sistemas de workflows, como ASKALON, Kepler, Pegasus, Taverna e Triana.

2.1.1 Workflows de Negócio, Científicos e Médicos

Workflows de negócio priorizam o fluxo de controle, as regras de negócio e a passagem de informações entre os participantes do processo. Por outro lado, um **workflow científico** "diz respeito à automação de um processo científico, composta por atividades cuja sequência é definida pelo fluxo de controle e pela dependência dos dados manipulados" (Yu e Buyya, 2005). Essa definição, muito embora se assemelhe à definição de workflow de negócio, apresenta uma importante diferença que não pode ser negligenciada: workflows científicos são fortemente voltados à manipulação de dados, o que os caracteriza como workflows orientados à fluxo de dados (i.e. *data-driven workflows*). Além disso, em geral, workflows científicos são computacionalmente intensivos e projetados para manipular e transformar grandes quantidades de dados.

O principal objetivo de workflows científicos é automatizar simulações e experimentos computacionais. No entanto, eles também tem se mostrado eficientes na construção, formalização e comunicação de métodos e processos científicos produzidos por pesquisadores (Cerezo, 2013). Desta forma, os resultados obtidos em um trabalho acadêmico podem ser publicado juntamente com os workflows que os geraram. Buscando esses objetivos, pesquisas tem se dedicado a investigar, modelar e implementar soluções que visam avanços em três aspectos dos sistemas de gerenciamento de workflows científicos, como é caso do trabalho desenvolvido por Goecks *et al.* (2010):

- **Acessibilidade:** facilitar o uso de ferramentas computacionais e workflows, de modo que seja fácil para um cientista criar workflows ou compreender um workflow criado por outros cientistas.
- **Reprodutibilidade:** garantir que o processo sintetizado por um workflow possa ser reproduzido por outras pessoas. Isso pode ser realizado de diversas formas, seja capturando a proveniência dos dados e suas transformações, ou permitindo que os próprios usuários classifiquem e documentem os objetivos das atividades que compõem seus workflows.
- **Transparência:** permitir que os dados originais, ferramentas computacionais e os processos de análise utilizados em um workflow possam ser publicados de forma que outras pessoas possam comparar, reproduzir e analisar os resultados obtidos em um trabalho.

Cerezo (2013) também cita dois outros aspectos importantes, que podem ser considerados como subprodutos da acessibilidade, reprodutibilidade e transparência:

- **Reuso e reaproveitamento:** permitir que os usuários utilizem workflows (ou trechos deles) criados por outras pessoas para atingir objetivos iguais ou diferentes.
- **Comparação:** permitir que workflows diferentes sejam comparados.

Observamos também a presença forte de workflows na indústria da saúde. Os objetivos desses workflows, no entanto, se assemelham mais aos dos workflows de negócio, ou seja, eles visam a automação dos processos que ocorrem dentro da rotina clínica de hospitais, centros de diagnósticos médicos e laboratórios. Estes sistemas, denominados Sistemas de Informação de Saúde (ou HIS - *Health Information Systems*) gerenciam processos como: preparação do paciente, coleta de material, aquisição de imagens, recepção de resultados, ato médico e emissão do laudo clínico. Em "*Strategic Information Management in Hospitals : an Introduction to Hospital Information Systems*", Haux (2003) descreve o papel e as características dos subsistemas e componentes de um HIS em um hospital. Alguns deles são:

[1] www.globus.com/toolkit

- ***Health Information Systems*** - **HIS:** um subsistema sociotécnico que engloba todo o processamento de informações, bem como os participantes que atuam no processamento e transformação dessas informações.
- ***Laboratory Information Systems*** - **LIS:** componente que auxilia nos processos do laboratório de análises clínicas, como coleta e distribuição de amostras e pedidos médicos, recepção de resultados das análises laboratoriais e transmissão dos resultados dos exames para os departamentos ou pessoas interessadas.
- ***Radiology Information Systems*** - ***RIS*:** outro subsistema cujo papel é auxiliar nos processos dos departamentos de radiologia, como preparação do paciente para a execução de um exame radiológico, utilização de insumos, materiais e medicamentos administrados nos pacientes, aquisição e organização de imagens e dados demográficos dos pacientes e composição, revisão e entrega de laudos médicos.

Em geral, sistemas de gerenciamento na saúde não incorporam os conceitos de gerenciamento de processos de negócio ou de workflows científicos que citamos anteriormente. Nestes sistemas, workflows não são formados a partir de componentes de software com escopos bem definidos. As atividades de um workflow estão fortemente acopladas umas com as outras, tornando difícil a tarefa de se reconfigurá-las, caso seja necessário. No entanto, há trabalhos que propõe uma abordagem baseada em workflows para sistemas de radiologia, como em "*Radiology Information System: a Workflow-Based Approach*" (Zhang *et al.*, 2009). Na indústria também encontra-se soluções que permitem maior flexibilidade na modelagem de workflows na rotina clínica, como é o caso do *MotionLIS*[2] e *RIS Carestream*[3].

2.1.2 O Ciclo de Vida de um Workflow

É importante também notar as diferenças entre os ciclos de vida dos workflows de negócio e científicos: a forma como são concebidos, validados, configurados, executados e analisados. Em geral, o ciclo de vida de um workflow de negócio é composto de fases em que atuam diferentes departamentos e áreas operacionais da organização, como mostra a figura 2.1 (Görlach *et al.*, 2011).

Weske (2012) descreve as fases do ciclo de vida de um workflow de negócios de forma similar a Görlach *et al.* (2011). A **modelagem** do workflow é a fase em que um especialista conduz o estudo sobre o processo de negócio. Esse estudo considera aspectos organizacionais e tecnológicos que circundam o processo. Uma vez concluída essa etapa, tem-se o **modelo** (ou **especificação**) de processo de negócio, que consiste em um conjunto de atividades e as regras de execução que regem seu fluxo. Uma **instância** de processo de negócio representa um caso específico dentro da operação da organização.

Uma vez validado o modelo de processo de negócio, deve-se então implementá-lo. Nesta fase, um especialista em Tecnologia da Informação deve incorporar o novo processo de negócio ao sistema de gerenciamento de processos (BPMS). Deste ponto em diante, a **execução** do processo de negócio pode ser realizada pelo BPMS, de acordo com as regras estabelecidas pelo modelo implementado. A cada execução cria-se uma instância do processo cujas transições de estado são controladas pelo BPMS. As instâncias de processo podem ser monitoradas por administradores de sistema através de *logs* de execução e ferramentas administrativas.

Por fim, as instâncias de um processo podem ser analisadas em termos de desempenho computacional e organizacional, por meio de dados estatísticos coletados durante um período de tempo. Como exemplo, esse tipo de análise pode detectar gargalos no processo de negócio, que podem ser evitados melhorando o modelo de processo de negócio. É importante ressaltar que essa melhoria é iterativa, ou seja, após a fase de análise o ciclo de vida se reinicia.

[2] http://www.touchhealth.com.br
[3] http://www.carestream.com

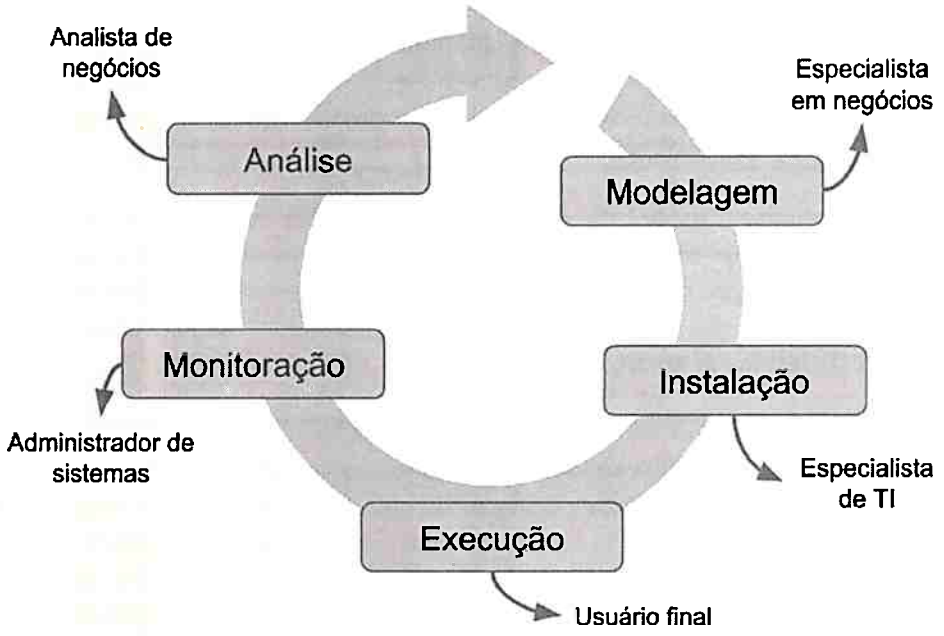

Figura 2.1: *Fases do ciclo de vida de um workflow de negócio e os respectivos departamentos ou papéis tipicamente envolvidos.*

Há uma série de diferenças entre o ciclo de vida de workflows científicos e de negócios, como mostrado em "*Business and Scientific Workflows*" por Barga e Gannon (2007). As diferenças mais relevantes são:

- A fase de modelagem do workflow é realizada em conjunto com a fase de execução, pois geralmente o cientista compõe o workflow por tentativa e erro.
- Apesar de realizar a modelagem, o cientista não necessariamente precisa conhecer os detalhes técnicos de cada atividade, assim como a forma como o workflow é executado.
- O processo de execução e monitoramento são concomitantes, pois do ponto de vista do cientista, os resultados da execução são verificados imediatamente e, caso não sejam satisfatórios, novos parâmetros podem ser utilizados, seguidos de uma nova execução.

Olabarriaga *et al.* (2014) observaram em seu trabalho que sistemas de workflows científicos podem ser utilizados por pessoas com os mais diferentes papéis na cadeia de pesquisa científica. Portanto, o sucesso de um sistema de gerenciamento de workflows em uma comunidade científica está condicionado à implementação dos requisitos dos diferentes perfis de usuário. Abordaremos mais a respeito deste assunto nas seções seguintes.

Além disso, ao contrário do que sugerem Görlach *et al.* (2011), há distinção clara entre um **modelo** e uma **instância** de workflow. Goecks *et al.* (2010) argumenta que um experimento científico precisa ser reprodutível, e para tanto é necessário capturar as atividades de uma dada análise para que se possa repeti-la.

Neste trabalho, defendemos que workflows para exploração de imagens médicas são modelados primeiro, para depois serem utilizados na pesquisa acadêmica ou na rotina clínica. Portanto, consideramos que modelos e instâncias de workflows são conceitos distintos.

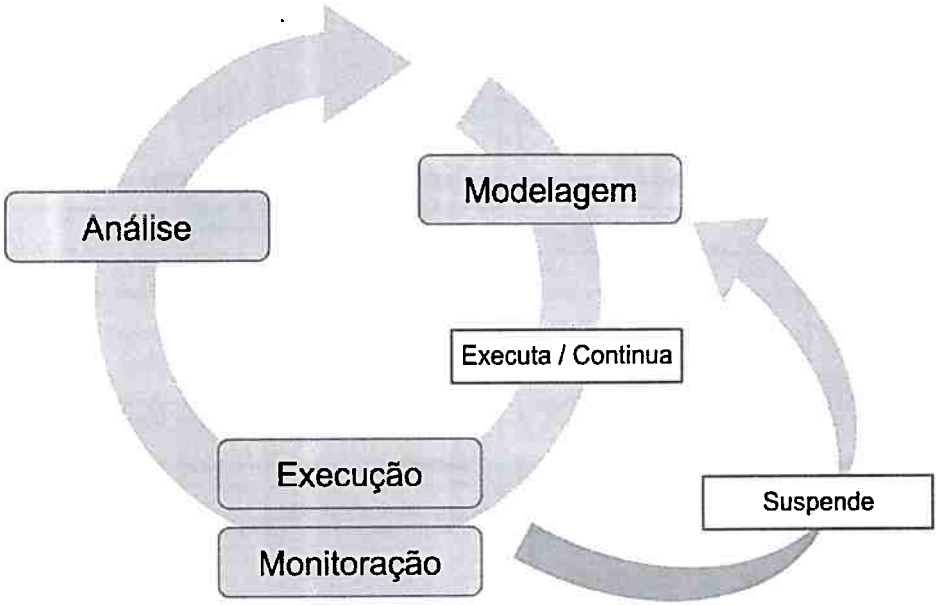

Figura 2.2: *Fases do ciclo de vida de um workflow científico.*

2.1.3 Abstração do Modelo de Workflow

Conforme vimos nas seções anteriores, workflows são utilizados para formalmente modelar análises de modo que estas possam ser posteriormente executadas em uma infraestrutura computacional. O usuários de um WfMS criam modelos de workflows cujo nível de abstração é certamente maior que o modelo de workflow que será executado computacionalmente.Yu e Buyya (2005) classifica esses dois níveis em **abstrato** e **concreto**, respectivamente.

No modelo abstrato, um workflow é descrito de forma abstrata, ou seja, sem referências explícitas aos recursos computacionais (*e.g.* algoritmos distribuídos, tecnologias web e infraestruturas computacionais distribuídas) necessários para a execução das atividades que o compõem. Em contrapartida, o modelo concreto vincula as atividades aos recursos específicos. A grande maioria dos sistemas de workflows trabalham com esses dois níveis de abstração. Os usuários criam modelos abstratos e o sistema de workflows os transformam em modelos concretos imediatamente antes de serem executados, em um processo chamado **escalonamento**.

Em sua tese de doutorado, Cerezo (2013) sugere que há ainda um outro nível de abstração chamado **conceitual**, cujo objetivo é tornar mais próximos o modelo de workflow do domínio da ciência. Os níveis de abstração envolvem diferentes enfoques no decorrer do ciclo de vida do workflow, como mostra a figura 2.3.

No trabalho acima citado, é proposto um método para mapear um modelo conceitual de workflow para um modelo abstrato. Este método é semiautomático, ou seja, existem decisões que devem ser tomadas pelo usuário durante a transformação do modelo conceitual para o abstrato. A interferência do usuário não é necessária no caso da transformação do modelo abstrato para o concreto, pois esse processo é automaticamente realizado pelo WfMS.

No caso dos workflows de negócio, os níveis de abstração permanecem os mesmos: o nível conceitual se preocupa apenas com informações e atividades relevantes para o entendimento do negócio, de modo que possa ser compreendido por profissionais sem conhecimento técnico. A transformação do nível conceitual para o abstrato é geralmente uma tarefa manual, executada por um arquiteto de sistemas ou desenvolvedor, como veremos na seção a seguir. O WfMS se encarrega da transformação para o modelo concreto, assim como ocorre para os workflows científicos.

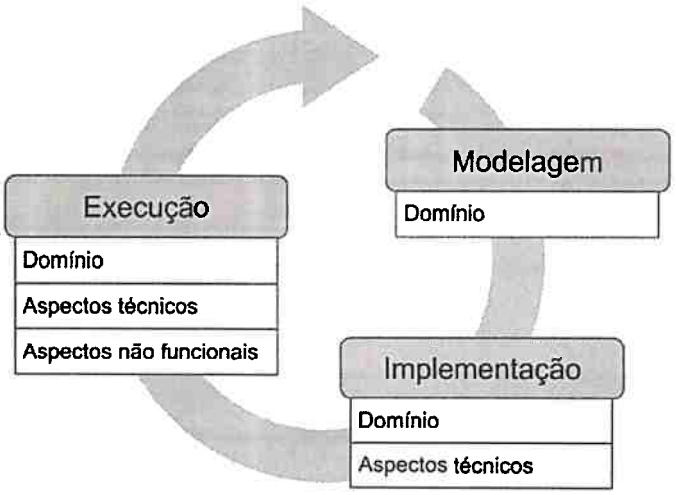

Figura 2.3: *Diferentes enfoques durante a modelagem, desenvolvimento e execução de um workflow.*

Olabarriaga *et al.* (2014) relacionam esses níveis de abstração com o ecosistema de workflows encontrados em alguns domínios da ciência. O nível conceitual, cujo enfoque é no domínio científico, é abrangido pelos portais científicos (*i.e.* interfaces entre o usuário e a infraestrutura computacional). O nível abstrato é contemplado pelo WfMS. Finalmente, o nível concreto é abrangido pelos sistemas escalonadores de atividades e pela infraestrutura computacional distribuída.

Na próxima seção veremos como os diversos perfis de usuário e os níveis de abstração de modelo estão relacionados.

2.1.4 Stakeholders

Para entendermos as partes interessadas em sistemas de workflows, voltemos brevemente aos workflows de negócio. Em "*Business Process Management*", Weske (2012) descreve os principais *stakeholders* de um sistema de workflows de negócios:

- ***Chief Process Officer***: é o curador dos processos da organização, ou seja, sua responsabilidade é padronizar os procesos, cuidar de sua evolução e garantir que haja interação entre eles de forma harmônica.
- **Engenheiro de negócios**: são especialistas de domínio, responsáveis por definir os objetivos estratégicos da organização e seus processos.
- ***Designer* de processos**: projetam os modelos conceituais de processo da organização conforme os requisitos coletados junto aos especialistas de domínio e outros *stakeholders*.
- **Participante do processo**: realizam as atividades de um processo durante a execução de um caso (*i.e.* instância). São um componente importante durante o projeto, execução e análise dos processos em que participam.
- **Dono do processo**: indivíduo designado a cuidar para que um dado processo seja executado de forma correta e eficiente.
- **Desenvolvedores**: também são profissionais com formação em Tecnologia da Informação, que desenvolvem os artefatos necessários para que os modelos conceituais de processos de negócio sejam realizados, incluindo as interfaces de comunicação com sistemas de informação existentes

na organização. São responsáveis, portanto, pela transformação do modelo conceitual para o abstrato e concreto.

- **Arquiteto de sistemas**: são especialistas com formação em Tecnologia da Informação, responsáveis por implementar e configurar os WfMS para que os processos da organização possam ser coordenados por seu motor de execução.

Esses stakeholders colaboram entre si durante o ciclo de vida de um workflow de negócio, como mostramos na seção 2.1.2.

Em "*Scientific Workflow Management - for Whom?*", Olabarriaga *et al.* (2014) estudam três comunidades científicas: Astrofísica, Heliofísica e Biomedicina e abordam para cada uma delas os seguintes aspectos:

- o contexto da área da ciência;
- a infraestrutura computacional adotada pela comunidade;
- as interfaces entre os usuários e a infraestrutura computacional, ou **portais científicos**; e
- as pessoas envolvidas.

Com base nestes aspectos, os usuários foram classificados em três perfis:

Especialistas de Domínio

Especialistas de domínio (ou cientistas), assim como os *Designers* de processos de negócio, modelam essencialmente workflows no nível conceitual, mas podem também contribuir no desenvolvimento de workflows e aplicações. Seu principal objetivo é obter o resultado final gerado pelo workflow. Muitos cientistas utilizam *scripts* em diversas linguagens de programação para orquestrar a execução de programas e assim obter os resultados desejados, como pudemos observar na Neurociência. Programas para fins específicos também são bastante utilizados, por exemplo na geração de visualizações de volumes ou cortes tomográficos.

Os especialistas de domínio interagem com os portais científicos, que abstraem a complexidade dos WfMS e da infraestrutura computacional. Tipicamente estes usuários executam workflows preexistentes com parâmetros ou dados diferentes.

Especialistas em Workflows

Especialistas em workflows desenvolvem modelos abstratos de workflows que podem ser usados por eles próprios ou por especialistas de domínio. Apesar de se posicionarem no nível abstrato, o conhecimento básico da infraestrutura computacional também é necessário.

Sua especialidade é o desenvolvimento de workflows utilizando as ferramentas, plataformas e linguagens de workflows disponíveis. Um desenvolvedor de workflows utiliza ferramentas similares a uma IDE (*Integrated Development Environment*) para:

- encontrar componentes, serviços e executáveis;
- criar, visualizar e alterar modelos de workflows;
- executar, testar e analisar os resultados de workflows;

Especialistas em Portais Científicos

Existem dois perfis principais de especialista em portais científicos. O **operador** é responsável pela instalação, manutenção e configuração do portal, enquanto o **desenvolvedor** é responsável por implementar as interfaces gráficas de usuário (GUI, do inglês *Graphical User Interface*) para executar, monitorar e entregar os resultados aos usuários. Para isso, os desenvolvedores utilizam APIs (*Application Programming Interfaces*) ou web services para integrar os portais aos diversos componentes do sistema de workflows (*e.g.* WfMS, escalonador de tarefas, etc).

Portais científicos são aplicações tipicamente complexas, desenvolvidas para se integrar com um ou mais WfMS. Os workflows desenvolvidos pelos especialistas em workflows são incorporados ao portal e abstraídos do restante dos usuários. A execução, monitoramento e coleta dos resultados são feitos através de APIs ou web services disponíveis no WfMS. Dessa forma, o portal consegue entregar ao usuário final o controle da execução do workflow, o seu status e progresso, bem como os resultados finais.

Apresentamos acima as características gerais dos três principais perfis de usuários de um sistema de workflow científico. A linha que separa esses perfis é tênue, ou seja, há intersecção de interesses e responsabilidades entre eles, como mostra a figura 2.4.

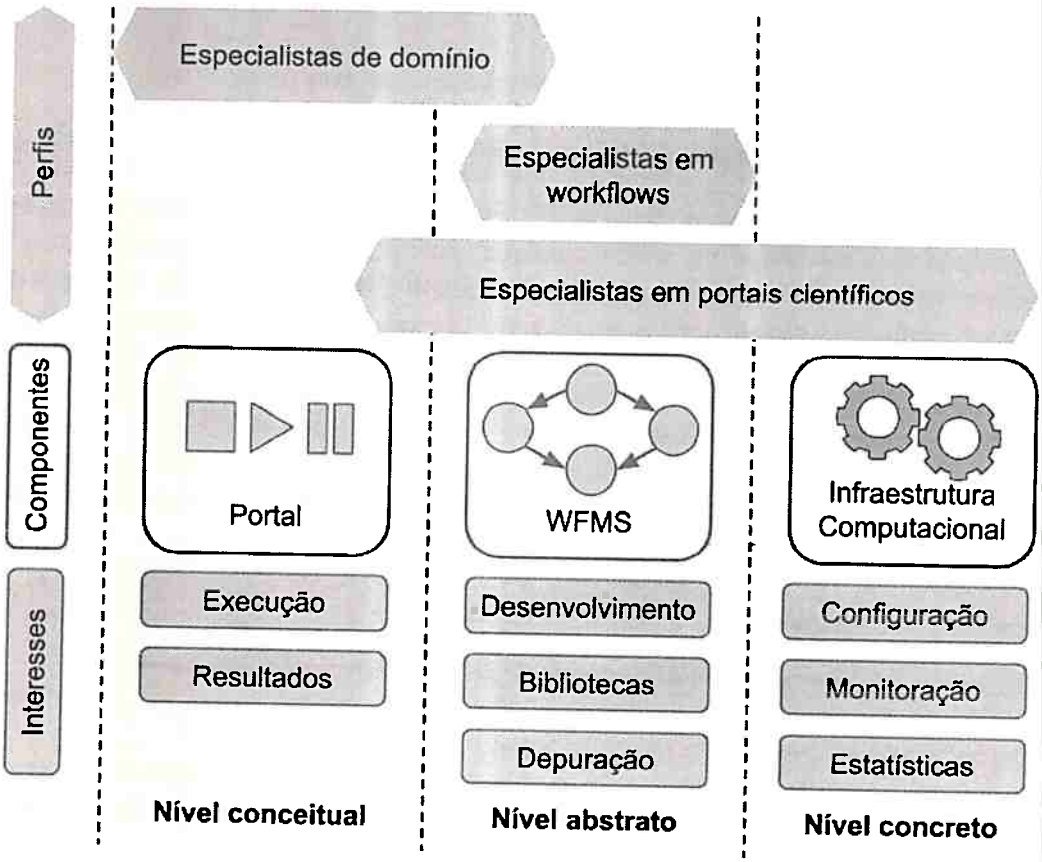

Figura 2.4: *Perfis de usuários de um sistema de workflows científicos, distribuídos nos níveis de informação conceitual, abstrata e concreta.*

2.1.5 Interações com Usuários

Muitos trabalhos se preocupam exclusivamente com a interação dos usuários com sistemas de gerenciamento de workflows, sem se preocupar com as prováveis interações destes usuários com sistemas satélite, que fazem parte do ecossistema de automação de análises, simulações científicas e exploração de grandes quantidade de dados. A lista abaixo reúne as principais interações já identificadas em trabalhos anteriores, bem como algumas que observamos no CHU:

- ***Application Programming Interfaces* (API)**: utilizado por usuários especialistas em portais científicos e workflows. São um conjunto de funções que permitem que outros programas criem, executem e monitorem workflows gerenciados pelo WfMS. APIs podem ser independentes de linguagem, como é o caso de web services baseados em SOAP (*Simple Object Access Protocol*) e WSDL (Web Services *Description Language*) ou APIs RESTful (*Representational State Transfer*).

- **Interfaces por linha de comando (CLI - *Command Line Interfaces*)**: permitem que usuários avançados, sejam eles especialistas em domínio ou em workflows, interajam com o sistema de workflows de forma não gráfica. Em geral, CLIs são mais eficientes que interfaces gráficas, pois além das respostas serem mais rápidas, a gama de comandos possíveis de se executar é maior. Em contrapartida, CLIs são menos acessíveis, pois exigem familiaridade com este tipo de interface, bem como conhecimentos avançados e específicos sobre o WfMS.

- **Interfaces gráficas de usuário (GUI - *Graphical User Interfaces*)**: são interfaces mais amigáveis que as CLIs (*aplicações web, Java Applets, aplicações para* desktop *instaláveis*, etc), sendo portanto mais acessíveis. Permitem criar, executar e monitorar workflows.

- **Linguagens específicas de domínio (DSL)**: são linguagens específicas que permitem que usuários possam criar modelos de workflow que sejam compreendidos e interpretados pelo motor de execução de um WfMS. São tipicamente utilizadas por especialistas em workflows, porém também são úteis para especialistas de domínio com conhecimentos mais avançados.

- **Portais científicos**: aplicações cujo público alvo é o especialista de domínio. Seu objetivo principal é abstrair os detalhes técnicos do WfMS, tornando a execução de workflows e coleta dos resultados mais acessíveis ao especialista de domínio. Este tópico foi abordado com mais detalhes na seção 2.1.4.

- **Repositórios de dados**: são sistemas com o objetivo de armazenar e organizar dados, sejam eles resultantes da execução de workflows ou dados originais que ainda serão processados. Estes repositórios podem ser genéricos, podendo acomodar teoricamente qualquer tipo de dados, ou específicos de um domínio da ciência (*e.g.* biologia, genética e biomedicina).

2.1.6 Representações de Modelos de Workflows

Conforme vimos ateriormente, um modelo de workflow especifica as atividades, suas dependências, restrições e regras que regem seu fluxo. A partir da especificação contida no modelo, o motor de execução do WfMS pode controlar e orquestrar a execução de workflows. Como apresentado na seção 2.1.3, modelos concretos exigem pouca ou nenhuma transformação, porém modelos abstratos ou conceituais necessitam ser transformados em um modelo concreto para então serem executados pelo WfMS.

Organizações têm utilizado workflows para processos de negócio muito antes da ciência utilizá-los para a automação de análises, simulações científicas e exploração de grandes quantidades de dados. Portanto é natural que workflows no domínio de negócios tenham maior grau de maturidade. Tanto o BPMN (*Business Process Model and Notation*, OMG (2011)) quanto o BPEL (Business Process Execution Language, OAS (2007)) são exemplos de iniciativas que se tornaram padrões *de facto* de mercado para representação formal de processos de negócio.

No domínio da ciência, no entanto, não há padrões amplamente estabelecidos, apesar de existirem trabalhos relevantes que objetivam a interoperabilidade entre diferentes motores de execução de workflows, como os publicados por Plankensteiner *et al.* (2011), Plankensteiner *et al.* (2013) e (Korkhov *et al.*, 2013). Este último explora o uso de plataformas para interoperabilidade dentro do domínio de processamento de imagens médicas. Entretanto, não há padronização para a construção de modelos de workflows em uma só linguagem ou representação, como acontece com workflows de negócio.

Na prática, grafos representam muito bem modelos de processos e simulações: os vértices são as atividades ou dados e os arcos são as dependências ou transições. Geralmente utiliza-se grafos direcionados, pois eles representam claramente a direção do fluxo de atividades ou da dependência entre dados e atividades. Nas seções a seguir apresentaremos como os WfMS podem ser classificados quanto ao **tipo de grafo**, **tipo de vértice**, **perspectiva de fluxo** (ou tipo de arco) e **nível de abstração** (Cerezo, 2013).

2.1.7 Tipo de Grafo

Os principais tipos de grafo utilizados em WfMS para representar modelos de workflows são:

- **Grafos Acíclicos Dirigidos (DAG - *Directed Acyclic Graph*)**: as estruturas em workflows baseados em DAGs podem ser classificadas em **sequência** (um conjunto ordenado de atividades em que uma atividade é iniciada após o término da predecessora), **paralelismo** (atividades que são executadas concomitantemente) e **escolhas** (atividades cuja execução está condicionada a uma regra que é avaliada em tempo de execução). Muitos WfMS adotam DAGs como base para seu modelo de workflow, pois eles são representações muito próximas do modo como as atividades são executadas em um WfMS, independente da infraestrutura computacional que sustenta o motor de execução.

- **Grafos Não-Acíclicos Dirigidos (Não-DAG)**: possui as mesmas estruturas dos DAGs, com a adição da **iteração** - estrutura que permite a repetição de atividades em um dado bloco. Essa diferença oferece mais flexibilidade ao modelo, permitindo que atividades complexas e monolíticas sejam substituídas por atividades menores e mais simples, agrupadas dentro de um laço.

- **Redes de Petri**: por definição, uma rede de Petri é um modelo matemático para representar sistemas distribuídos discretos. Elas também são grafos não-DAG bipartidos. Este fato, aliado com à notação gráfica intuitiva das Redes de Petri, fez com elas se tornassem a estrutura central para os modelos de diversos WfMS.

2.1.8 Tipo de Vértice

Em workflows de negócio ou científicos baseados em grafos (DAG ou não-DAG), cada nó representa uma atividade de processamento, como invocação de um programa, script, *web services*, componente de sistema, classe Java, etc. Em workflows de negócio é comum que uma atividade envolva a ação de pessoas, porém isso não impede que partes (ou até mesmo a totalidade) de um processo de negócio ocorra de forma automática, isto é, sem intervenção humana.

Por outro lado, em workflows científicos é incomum a existência de atividades que exigem interação humana. No geral, análises, simulações e exploração de grandes quantidades de dados são compostas de atividades automáticas, ou seja, não há a necessidade de intervenção de um participante do workflow.

Workflows médicos, como aqueles encontrados em RIS e PACS, possuem ambos os tipos de atividades: as que envolvem ações de pessoas (*e.g.* biomédicos, radiologistas, enfermeiras) e as que são executadas automaticamente (*e.g.* compressão de dados, gravação de imagens e metadados).

Os vértices em workflows cujo modelo é baseado em Redes de Petri podem representar **lugares** (elementos estáticos de um workflow, como um estado, dados ou condições) ou **transições** (elementos dinâmicos de um workflow, como a invocação de um programa ou *web services*).

2.1.9 Perspectiva de Fluxo

Em um modelo de workflow representado por um grafo direcionado, os vértices (atividades de processamento) são conectados por arcos, denotando a direção do fluxo entre eles. De acordo com o tipo de fluxo que essas arestas representam, o modelo de workflow pode ser classificado em:

- **orientado a dados**: quando as arestas denotam o **fluxo de dados**, ou seja, a transferência de dados de uma atividade a outra.
- **orientado a controle**: quando as arestas denotam o **fluxo de controle**, ou seja, a ordem de precedência com que as atividades devem ser executadas.
- **híbrido**: quando as arestas denotam tanto o fluxo de dados quanto o fluxo de controle.

Veremos mais a respeito dessas perspectivas nas próximas duas sub-seções.

Modelos orientados a dados

Modelos orientados a dados se adequam muito bem a workflows científicos, pois estes com frequência manipulam e transformam grandes quantidades de dados. Workflow deste tipo são fáceis de criar e compreender. Além disso, são implicitamente paralelizáveis e distribuídos, pois: (i) o motor de execução pode executar atividades assim que os dados estiverem disponíveis para tal e houver recursos computacionais disponíveis na infraestrutura distribuída; (ii) coleções de dados podem ser automaticamente divididas e distribuídas para recursos computacionais de modo a serem processadas paralelamente.

Bharathi *et al.* (2008) citam algumas estruturas utilizadas para modelar o fluxo de dados em workflows científicos:

- **Processo**: é a estrutura básica de um modelo orientado a dados e representa uma atividade que consome dados, os processa e produz como saída dados os transformados.
- ***Pipeline***: é a combinação de um ou mais processos em sequência.
- **Distribuição**: é um processo que produz dados que serão consumidos por múltiplas atividades. Distribuições são geralmente usados para dividir grandes quantidades de dados em porções menores e então distribuí-las para outras atividades. É a estrutura que concede o paralelismo ao workflow.
- **Agregação**: é capaz de consumir dados de múltiplas atividades, agregá-los e finalmente gerar um produto destes dados como saída. É a estrutura responsável por sincronizar as linhas de execução paralelas em uma instância de workflow.
- **Redistribuição**: combina a redução e o particionamento de dados, ou seja, consome dados de múltiplas atividades, os processa e produz como saída uma nova coleção de dados (que por sua vez poderão ser consumidos por outras atividades). Portanto esta estrutura concede ao workflow as habilidade de paralelismo e sincronização.

Infelizmente, estruturas de fluxo de controle como condicionais e laços não podem ser representadas em modelos orientados a dados, a menos que se adote um modelo híbrido ou se recorra a atividades que emulem o comportamento dessas estruturas, como mostra a figura 2.6.

Modelos orientados a controle

Neste modelo o fluxo não depende dos dados, ou seja, não há relação de dependência entre atividades que compõem o workflow e dados. O controle é passado de uma atividade para a outra de acordo com estruturas como sequências, laços e condicionais. Em representações gráficas, como na figura 2.7, retângulos e elipses representam atividades e conectores, respectivamente, enquanto que as setas representam as relações de precedência entre as estruturas do workflow.

Algumas das principais estruturas utilizadas para modelar um fluxo de controle são:

- **Sequência**: é a estrutura mais elementar do modelo orientado a controle. Indica a ordem de precedência de execução de atividades. Por exemplo, no workflow *(a)* da figura 2.7, a atividade F só pode ser executada após o término de E.

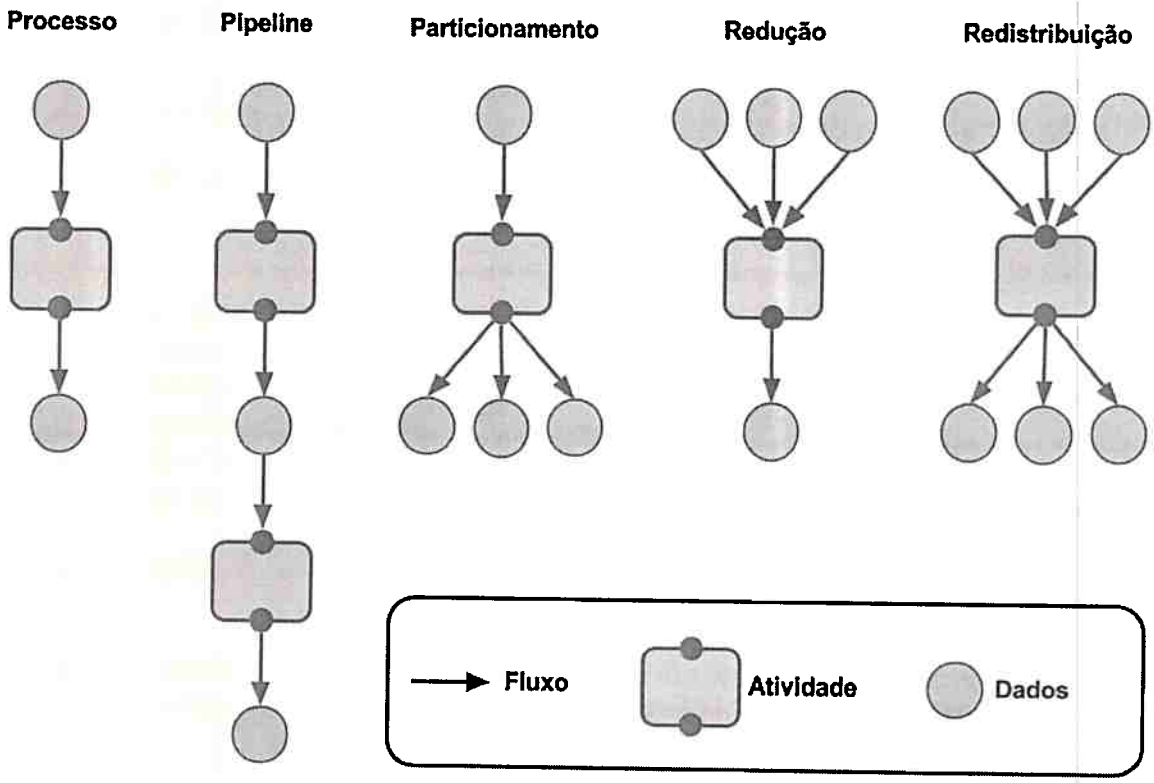

Figura 2.5: *Representação gráfica das estruturas utilizadas em modelos de workflows orientados a dados:* processo, pipeline, *particionamento, redução e redistribuição.*

- **Paralelização** (divisão E): Divide a linha de execução atual em diversas linhas paralelas, como ocorre com as atividades B, C e D do workflow *(a)* (figura 2.7).
- **Sincronização** (junção E): sincroniza diversas linhas de execução paralelas, reduzindo-as a uma única linha de execução. Esse é o caso da atividade E (workflow *(a)* da figura 2.7), que é executada apenas ao término de B, C e D.
- **Escolha exclusiva** (divisão OU exclusivo): é um ponto no workflow onde uma linha de execução é escolhida dentre as alternativas possíveis. Essa escolha é realizada a partir de uma regra cuja avaliação ocorre em tempo de execução. Por exemplo, apenas uma das atividades C e D será escolhida após a execução de B no workflow *(b)* da figura 2.7.
- **Junção** (junção OU exclusivo): é a estrutura que une fluxos de execução alternativos, sem sincronização. No workflow *(b)* da figura 2.7, a atividade B pode ser executada após o término de C ou A.
- **Laço** (iteração): é a estrutura que permite que um conjunto de atividades seja executada repetidas vezes, como é o caso das atividades B e C no workflow *(b)* da 2.7.

O objetivo principal dos workflows científicos é o processamento, transformação e exploração de dados (frequentemente em grandes quantidades). No entanto, workflows orientados a controle não oferecem estruturas para representar a passagem de dados de uma atividade a outra. Para modelar dependências de dados, utiliza-se atividades intermediárias que realizam a transferências de dados e a sincronização entre a atividade que produz os dados e aquela que os consomem, como mostra a figura 2.8.

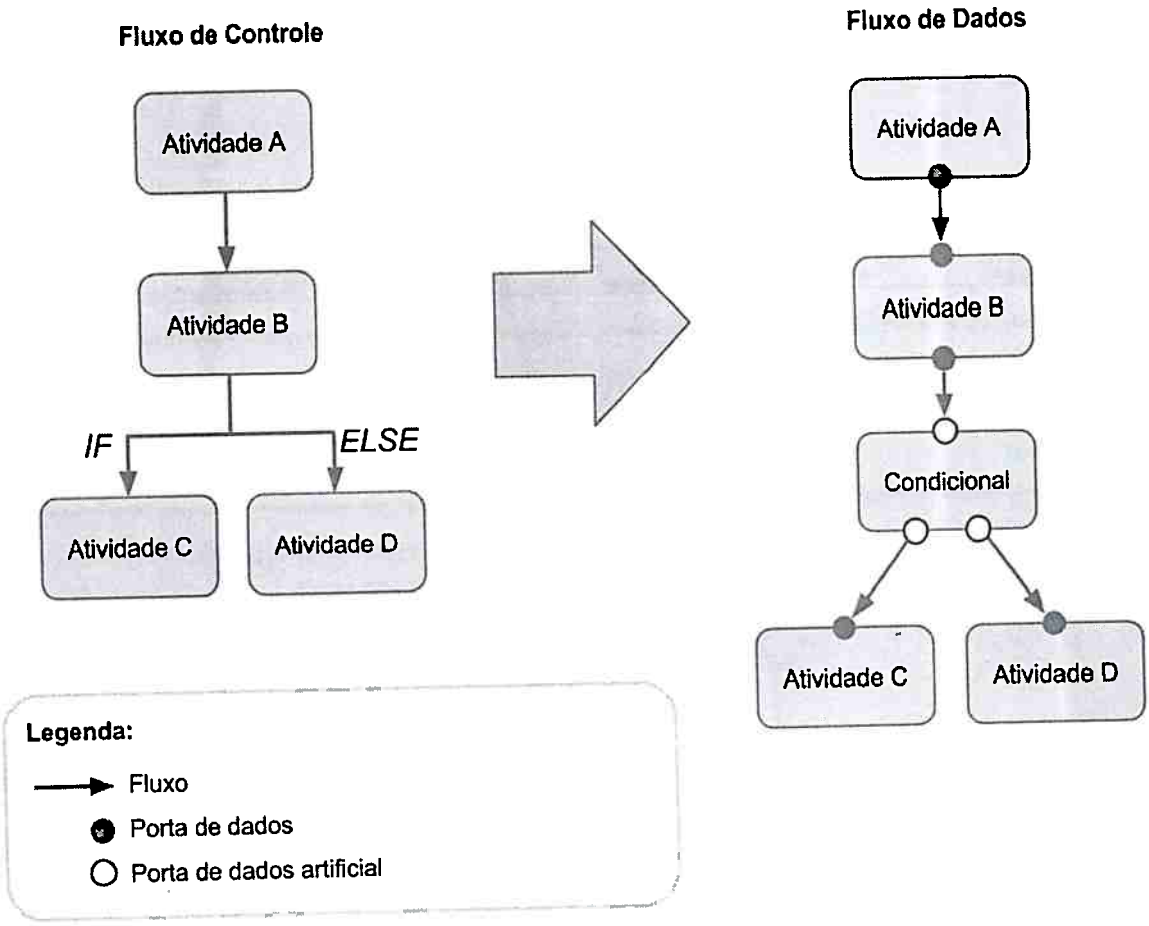

Figura 2.6: *Uso de estruturas de controle em workflows orientados a dados.*

2.1.10 Critérios para Classificar a Abstração em Workflows

Na seção 2.1.3 apresentamos três níveis de abstração para modelos de workflows: conceitual, abstrato e concreto. Também vimos que modelos abstratos são utilizados em grande parte dos WfMS. Entretanto, o grau de abstração oferecido pelos WfMS depende de muitos fatores. Cerezo (2013) propõe os quatro critérios a seguir para classificar quão abstrato é o modelo de workflows (o grau de abstração é diretamente proporcional à quantidade de critérios que o WfMS atende):

1. **Anotações:**
   - O sistema é capaz de gerenciar anotações semânticas em workflows ou em seus componentes?
   - Como os usuários realizam estas anotações (*e.g.* via palavras-chave curadas, etiquetas)?
2. **Criação:**
   - O sistema é capaz de criar workflows automaticamente?
   - Há algum mecanismo que sugere vértices e arcos enquanto o usuário constrói um workflow?
   - O sistema verifica a integridade do workflow durante sua construção (*e.g.* o tipo de dado da uma saída de uma atividade é compatível com a entrada de dados da atividade subsequente)?
3. **Flexibilidade:**

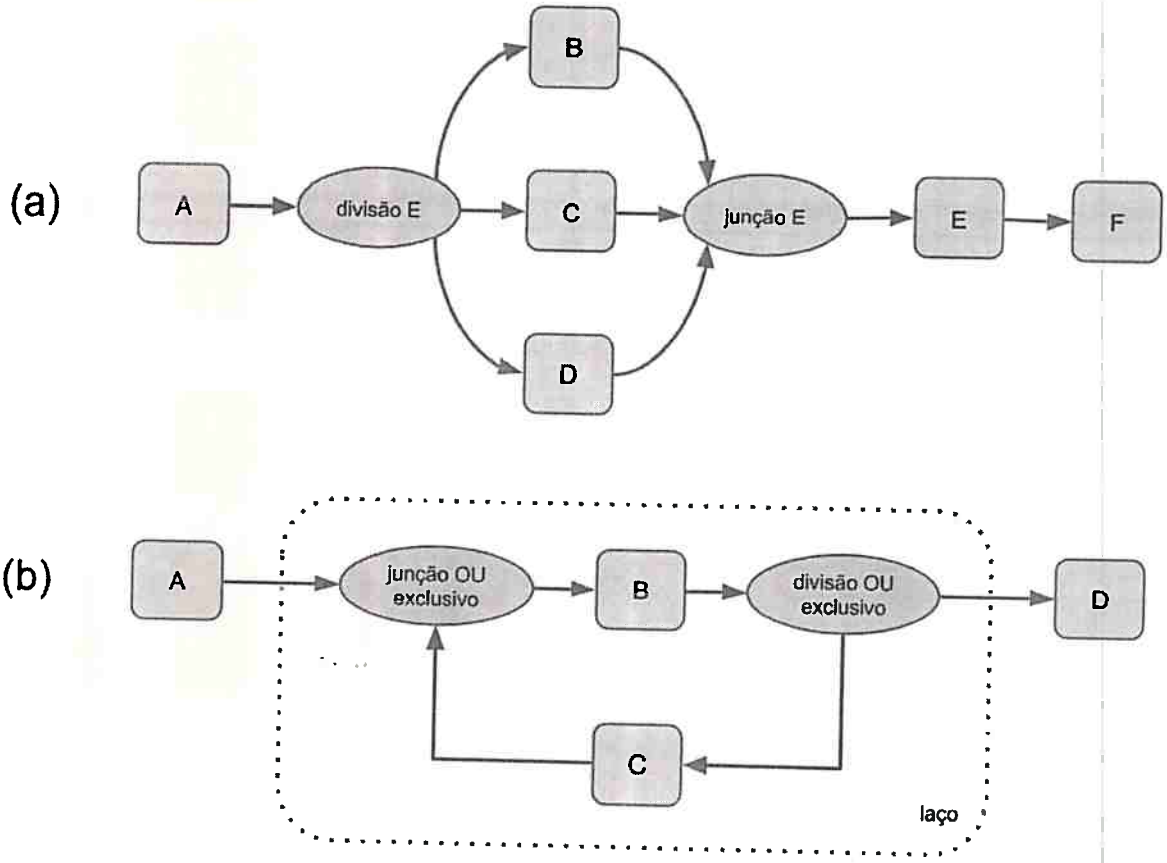

Figura 2.7: *Exemplo de modelo orientado a controle: (a) sequência, paralelização e sincronização, (b) escolha esclusiva, junção e laço.*

- O mesmo modelo de workflows pode se concretizar em diferentes estruturas de execução (*e.g.* dependendo do contexto e da disponibilidade de recursos, duas atividades podem ser executadas paralelamente ou em sequência)?
- Coleções de dados podem ser representadas como um único parâmetro de entrada?
- Mais de um tipo de dados pode ser usado como entrada de uma atividade?

4. **Indireção** (ou desacoplamento):
   - Existe algum nível de indireção entre uma atividade e os detalhes técnicos de sua execução (*e.g.* atividades compostas, subprocessos)?

2.1.11 Comparação entre WfMS

Nesta seção, ***expandimos os critérios de análise*** propostos por Cerezo (2013) para compor uma análise de alguns WfMS por seis perspectivas diferentes:

- **Interações com usuários**: as formas com que os usuários interagem com o sistema para criar e executar workflows.
- **Modelo de workflow**: como o modelo de workflow representa o fluxo de controle e de dados.
- **Nível de abstração**: qual nível de abstração (conceitual, abstrato ou concreto) o sistema utiliza.

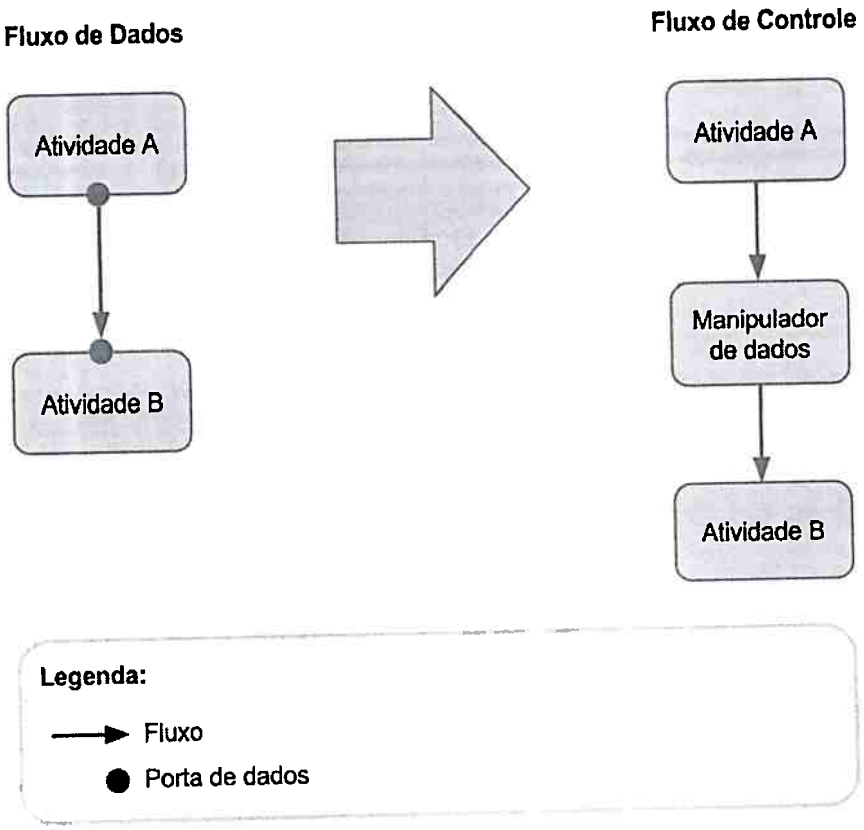

Figura 2.8: *Representação de dependências de dados em workflows orientados a fluxo.*

- **Extensibilidade**: como o sistema pode incorporar requisitos e ferramentas utilizadas em outros domínios da ciência.
- **Abertura e transparência**: o código fonte do sistema pode ser aberto ou fechado. Projetos cujo código é aberto e livre possuem uma maior base de usuários, melhor documentação e maior envolvimento da comunidade científica e de desenvolvedores.
- **Gerenciamento de Dados**: como os dados originais e resultados finais são armazenados.

A tabela 2.1 compara os sistemas de workflows **Pegasus**, **WINGS**, **Galaxy**, **ASKALON** e **Taverna** sobre os critérios de interações com usuários, modelo de workflow, nível de abstração e abertura e transparência do código fonte:

Extensibilidade

Uma atividade é uma representação de um artefato executável, como programas, *scripts*, *web services*, serviços em grade e em nuvem, dentre outros. Na prática, as atividades encapsulam e abstraem os detalhes técnicos relativos à execução dos artefatos subjacentes.

Todo WfMS possui um arcabouço para desenvolvimento de novos tipos de atividades, porém alguns são mais flexíveis que outros. Através desses arcabouços é possível definir modelos de atividades (ou **componentes**), que especificam: (i) como o artefato deve ser executado; (ii) quais parâmetros devem ser passados para o artefato; e (iii) os produtos finais da execução do artefato. Um exemplo de componente é apresentado na figura 2.9: o parâmetro que deve ser passado para o programa *wc* é modelado como uma **porta de entrada** e seu produto como **porta de saída** e o *script shell* especifica como o programa *wc* deve ser executado.

WfMS	Interações	Modelo	Código fonte	*Nível de abstração*			
				An	Cr	Flex	Ind
Pegasus	A, C, D	DAG □	Aberto	○	○	○	◐
WINGS	G◇, R	DAG □	Aberto	●	●	○	●
Galaxy	A, C, G◇ , D, P, R	¬DAG □	Aberto	●	◐	◐	◐
ASKALON	G, D	¬DAG ►□	Fechado	◐	◐	●	○
Taverna	A, G, D	DAG □►	Aberto	◐	◐	●	◐

Interações: A: *application programming interface*, C: *linha de comando*, G: *interface gráfica de usuário*, G◇: *interface gráfica web*, D: *linguagem específica de domínio*, P: *portal científico*, R: *repositório de dados*

Modelo: DAG: *grafo acíclico dirigido*, ¬DAG: *grafo não-acíclico dirigido*, □: *orientado a dados*, ►: *orientado a controle*

Nível de abstração: An: *anotações*, Cr: *criação*, Flex: *flexibilidade*, Ind: *indireção*

Características de abstração: ○: *ausente*, ◐: *parcialmente presente*, ●: *presente*

Tabela 2.1: *Matriz de comparação entre os WfMS estudados, baseada no trabalho de (Cerezo, 2013).*

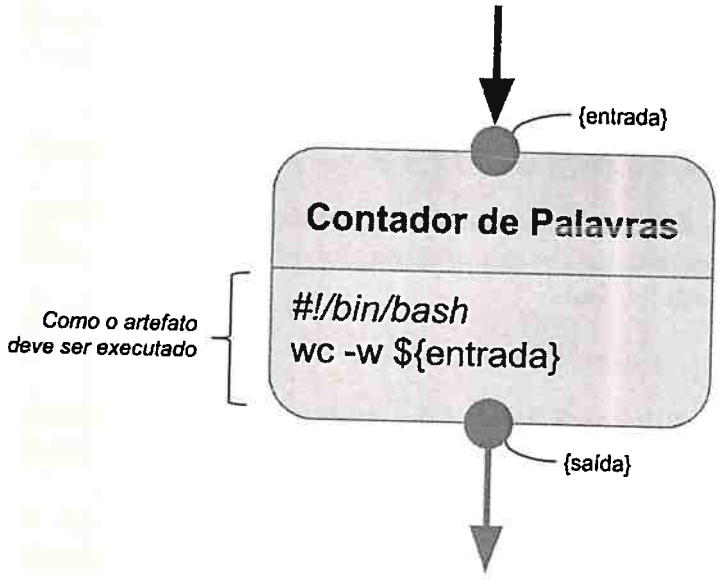

Figura 2.9: *Exemplo de modelo de atividade, ou componente.*

A tabela 2.2 compara alguns sistemas de workflows quanto aos seguintes critérios:

- **Tipos de dados personalizados:** é possível definir novos tipos de dados no sistema?
- **Especificação de componentes**: é a forma com que uma atividade é modelada no sistema (*e.g.* via XML, script, etc).
- **Biblioteca de componentes**: o sistema possui uma biblioteca de componentes? Ou seja, possui um catálogo que pode ser consultado pelos usuários para auxiliar na construção de workflows?
- **Tipos de artefatos:** quais tipos de artefatos executáveis podem ser incorporados ao sistema?

WfMS	Dados	Bib.	Mod. At.	*Tipos de artefatos*			
				CLI	WS	SS	Outros
Pegasus	○	○	XML, Conf	●	●	●	Condor-G, Globus
WINGS	●	●	XML, Conf	●	●	●	Condor-G, Globus
Galaxy	●	●	XML	●	●	●	Docker, serviços em grade
ASKALON	○	●	XML	●	●	●	Globus
Taverna	●	●	XML	●	●	●	serviços em grade

○: *ausente,* ◐: *parcialmente presente,* ●: *presente*

Dados: *é possível definir novos tipos de dados?*

Bib.: *possui repositório de atividades?*

Mod. At. (Modelo de atividade): XML: *eXtensible Markup Language,* Conf: *arquivos de configuração*

Tipos de artefato: CLI: *linha de comando,* WS: *web service,* SS: *script shell*

Tabela 2.2: *Comparação dos WfMS segundo critérios de extensibilidade.*

Gerenciamento de dados

Como mostramos nas seções anteriores, dados são a parte central dos workflows científicos. Entretanto a estratégia para gerenciar coleções de dados difere muito entre os WfMS, tanto na perspectiva de movimentação quanto na de armazenamento:

- **Movimentação de dados**: em WfMS distribuídos, os dados de entrada de uma atividade precisam ser transferidos para o local onde o processamento efetivamente ocorrerá. Além disso, os dados produzidos por essa atividade poderão ser consumidos por outras atividades. Portanto esses dados intermediários também devem ser transferidos para os locais onde serão consumidos. Em alguns WfMS, a movimentação de dados é **manual**, ou seja, precisa ser especificada pelo usuário juntamente com o modelo de workflow. Em outros sistemas a movimentação é **automática**, e pode ser realizada de forma **centralizada**, **intermediada** ou em **pares**. Na forma centralizada, todas as transferências de dados passam por um ponto central. Na forma intermediada, as transferências são gerenciadas por um sistema distribuído de dados. Por fim, as transferências em pares são realizadas diretamente entre os recursos computacionais, sem intermédio de sistemas e sem passar por um ponto central (Yu e Buyya, 2005).

- **Armazenamento de dados**: alguns WfMS são capazes de armazenar e gerenciar os dados de entrada e de saída de workflows. Os dados de entrada podem ter diversas origens, como bases de genomas, arquivos de imagens médicas, dentre outras. Já os dados de saída são aqueles gerados durante a execução dos workflows, que podem ser tanto os resultados finais, quanto os intermediários. Classificamos o armazenamento de dados em **local** ou **centralizado**. No armazenamento local, os dados são gravados no computador do usuário. No armazenamento centralizado, os dados são armazenados em um sistema remoto, podendo ser compartilhados com outros usuários.

A tabela 2.3 compara alguns WfMS com relação aos aspectos de gerenciamento de dados acima descritos.

WfMS	Movimentação	Armazenamento
Pegasus	Intermediada	Local
WINGS	Intermediada	Centralizado, com compartilhamento
Galaxy	Centralizada, Intermediada*	Centralizado, com compartilhamento
ASKALON	Manual, Centralizada	Local
Taverna	Centralizada	Local

* *via Pulsar (https://github.com/galaxyproject/pulsar)*

Tabela 2.3: *Comparação dos WfMS segundo aspectos de gerenciamento de dados.*

2.2 Submissão e Controle Distribuído de Atividades

Em geral, sistemas de gerenciamento de workflows possuem duas estratégias para executar as atividades de um workflow: **local** e **distribuída**. Quando o motor de execução e as atividades que ele coordena são executados no mesmo recurso computacional (ou mesmo nó de processamento), diz-se que a execução é local. Já quando as atividades são executadas em outros recursos computacionais (ou nós), a execução é distribuída.

O processamento local de atividades oferece uma barreira para a escalabilidade. O poder computacional neste caso está confinado aos recursos locais como memória, dispositivos de armazenamento e CPUs. Portanto há somente uma alternativa de crescimento: o **vertical**. Ou seja, o aumento da capacidade de processamento só pode ser atingido através da adição de recursos a um único nó.

Por outro lado, quando a execução de atividades é distribuída, pode-se escaloná-la **horizontalmente**, ou seja, consegue-se aumentar o poder computacional através da adição de novos nós ao sistema. Neste caso, o motor de execução deve gerenciar:

- o escalonamento das atividades;
- os nós de execução;
- o monitoramento do ambiente distribuído.

É comum que os WfMS deleguem parte dessas tarefas a um sistema de gerenciamento de recursos distribuídos (DRMS, do inglês *Distributed Resource Management System*, como HTCondor (Thain *et al.*, 2005), Apache Hadoop YARN (Vavilapalli *et al.*, 2013), Apache Mesos (Hindman *et al.* , 2011) e TORQUE/PBS (Staples, 2006). Estes sistemas são capazes de abstrair CPU, memória, e outros recursos computacionais em um aglomerado (*cluster*), grade (*grid*) ou nuvem computacional, provendo APIs para gerenciamento de recursos e escalonamento às aplicações usuárias (por exemplo, um WfMS).

A figura 2.10 mostra como o escalonamento das atividades de um workflow ocorre em um WfMS. O motor de execução envia um sinal ao DRMS indicando que há uma atividade que precisa

ser executada. Os requisitos específicos da atividade também são informados ao DRMS, como a quantidade de memória e CPUs necessários, quais artefatos executáveis serão requeridos, etc. Em seguida, o DRMS decide quais dos recursos computacionais disponíveis são compatíveis com esses requisitos. Por fim, a ordem de execução é submetida ao recurso escolhido e os resultados obtidos são retornados ao motor de execução.

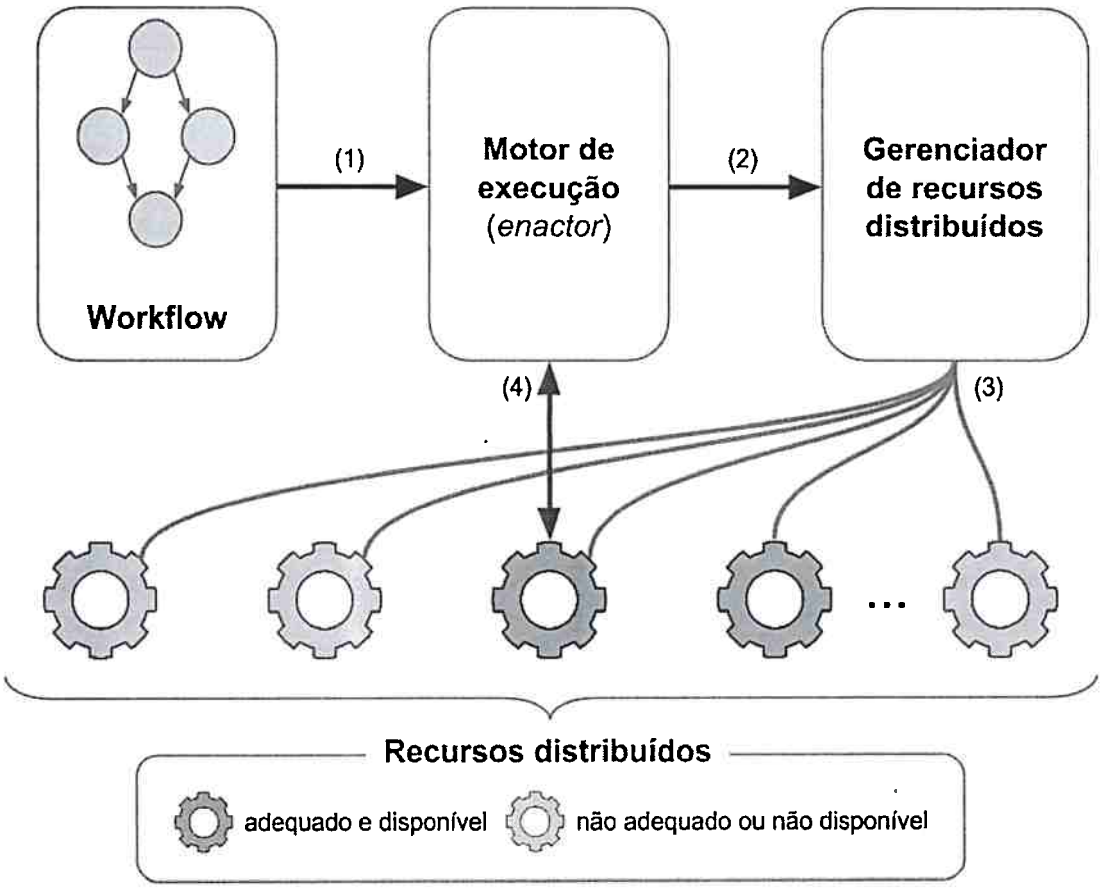

Figura 2.10: *Interação entre o WfMS e o DRMS durante a execução de um workflow.*

(1) o workflow é interpretado pelo motor de execução; (2) o motor de execução envia um sinal ao DRMS indicando que há uma atividade que precisa ser executada; (3) cada recurso computacional existente na infraestrutura distribuída reporta constantemente seu estado ao DRMS; através dessas informações o DRMS decide qual dos recursos disponíveis são compatíveis com os requisitos da atividade; (4) a ordem de execução é submetida ao recurso computacional escolhido e os resultados obtidos são retornados ao motor de execução

2.2.1 Processamento em Lote em Recursos Distribuídos

A transformação do *hardware* distribuído em comodato possibilitou o surgimento dos **aglomerados** de computadores, ou seja, conjuntos de computadores que colaboram entre si para um atingir um mesmo objetivo. O paradigma de sistemas distribuídos trouxe muitos benefícios para o processamento em lote (*batch processing*), porém também aumentou consideravelmente a complexidade dos programas. Surgiram então os sistemas de gerenciamento de aglomerados, que controlam o escalonamento de atividades, gerenciam os nós de execução e monitoram o ecossistema distribuído (*e.g.* GridEngine, TORQUE/PBS e HTCondor), como descrevem Tröger e Merzky (2014).

A computação paralela e distribuída em grande escala, utilizada principalmente pelas comunidades de computação de alto desempenho (HPC, do inglês *High Performance Computing*), motivou o nascimento de novas estruturas distribuídas, como as **grades computacionais**. Grades são plataformas cooperativas formadas por recursos computacionais heterogêneos mantidos por diferentes

organizações e possivelmente geograficamente distribuídos (Foster e Kesselman, 2004). Já em meados dos anos 2000, um nova modalidade de computação distribuída surgiu, desta vez impulsionada pelas necessidades da indústria e imbuída de um novo modelo de negócio - a **computação em nuvem**. Desde então, as nuvens computacionais têm se tornado uma importante modalidade não só na indústria, mas também nas comunidades acadêmicas. Abordaremos mais sobre esse assunto seção 2.3.

Tröger e Merzky (2014) afirmam que o uso do processamento em lote não é exclusivo da computação de alto desempenho. Arcabouços que suportam o modelo de programação *MapReduce*, como o Apache Hadoop (White, 2009), também realizam processamento de dados em lote, coordenando a distribuição de atividades e gerenciando os nós de execução pertencentes à grade. A execução de workflows de negócio, científicos e médicos também é um exemplo típico de uso de processamento em lote, tanto em grades quanto em nuvens computacionais.

Para todos os casos de uso de processamento em lote, os DRMSs são importantes facilitadores para a utilização dos aglomerados, grades e nuvens, pois eles: (i) abstraem do programador a complexidade da computação distribuída; e (ii) são capazes de utilizar os mais variados tipos de infraestrutura, independente do *hardware* ou *middleware* subjacente.

2.2.2 Sistemas de Gerenciamento de Recursos Distribuídos

Como veremos mais adiante na seção 2.3, a computação em nuvem oferece um modelo de negócio mais viável, tanto para quem provê o serviço, quanto para quem o consome. São três as principais vantagens: (i) consegue-se maximizar a utilização de recursos computacionais disponíveis; (ii) facilita-se a gestão da infraestrutura computacional (e.g. nós de computação, armazenamento e redes de dados); e (iii) torna mais justo a distribuição dos custos de utilização dos recursos computacionais da nuvem. Diante desse novo paradigma de computação distribuída, a relevância da submissão e controle de atividades em nuvens computacionais tem aumentado.

Alguns trabalhos já propuseram uma evolução dos padrões existentes de DRMS para a utilização em nuvens computacionais, como é o caso de Tröger e Merzky (2014). A indústria também tem encontrado soluções para a execução de processos em lote em plataformas de nuvem, como é o caso dos arcabouços Apache Mesos e Apache Hadoop YARN.

Emprestamos de Tröger e Merzky (2014) a seguinte definição de DRMS, que utilizaremos no restante deste trabalho: "Um Sistema de Gerenciamento de Recursos Distribuídos é qualquer sistema que implementa a execução de tarefas computacionais em recursos distribuídos. Alguns exemplos são: um sistema multi-processado controlado pelo escalonador de um sistema operacional, um sistema em um aglomerado de computadores que é controlado por um escalonador central, um sistema em grade, ou um serviço em nuvem para execução de atividades computacionais".

Como vimos, existem diversas implementações de DRMS (*e.g.* GridEngine, TORQUE/PBS e HTCondor), algumas das quais adotam interfaces de programação padronizadas ou seguem algum padrão na forma como a submissão de atividades é realizada. Um dos principais padrões é o *Distributed Resource Management Application API* (**DRMAA**), concebido logo após o surgimento da computação em grade. Seu principal objetivo é definir uma API padrão para interação com diversos DRMS, priorizando a portabilidade entre diferentes implementações.

Neste contexto, define-se que uma **aplicação** é um artefato de software que utiliza um DRMS por intermédio de APIs. Algumas dessas APIs são proprietárias, ou seja, são interfaces específicas para um determinado DRMS, enquanto que outras APIs são padronizadas, ou seja, são interfaces comuns a diversas implementações de DRMS, como é o caso do padrão DRMAA. A vantagem da segunda com relação a primeira é o desacoplamento da aplicação à implementação do DRMS, possibilitando assim a portabilidade.

Por fim, terminaremos esta seção com três importantes conceitos de um DRMS:

- **Nó de submissão**: é um recurso computacional que executa uma aplicação, ou seja, é o nó responsável por enviar atividades a uma implementação de DRMS.

- **Nó de execução**: é um recurso computacional capaz de executar atividades enviadas a uma implementação de DRMS. Um mesmo recurso computacional pode acumular a responsabilidade de nó de execução e de submissão.
- **Atividade**: é um tarefa computacional enviada por uma aplicação a uma implementação de DRMS. Cada atividade se traduz em um ou mais processos de um sistema operacional, e pode ser executada por um ou mais nós de execução.

Na seção a seguir abordaremos a computação em nuvem e visitaremos algumas estratégias de uso de um DRMS neste novo tipo de plataforma computacional.

2.3 Computação em Nuvem

A computação em nuvem é, antes de tudo, uma evolução de tecnologias, paradigmas e modelos de negócio em torno da Tecnologia da Informação e Comunicação. Este novo paradigma, juntamente com o termo "*cloud computing*", foi popularizado em 2006 com o lançamento do *Amazon Elastic Compute Cloud* (Amazon EC2)[4]. Nascia ali uma nova modalidade de serviço que possibilitava o provisionamento e configuração de recursos computacionais de forma simplificada e sob demanda, podendo-se aumentar ou diminuir a quantidade de servidores conforme necessário. Conforme anunciado ao público pela Amazon, a computação em nuvem "muda o modelo econômico da computação, fazendo com que se pague apenas pela capacidade [computacional] que é efetivamente utilizada". Portanto, pode-se dizer que o surgimento da computação em nuvem foi impulsionado pela necessidade de um modelo de computação economicamente viável para a indústria.

Na prática, uma nuvem computacional é composta por um conjunto heterogêneo de *hardware* interconectados por uma rede de computadores, controlado através de um arcabouço de gerenciamento de nuvens (**e.g.** OpenNebula[5] e OpenStack[6]). Este arcabouço é responsável por combinar tecnologias de virtualização (*e.g.* KVM, Xen, VMware, LXC) com funcionalidades específicas da nuvem, como elasticidade e provisionamento automático de recursos, locação múltipla (*multi-tenancy*), faturamento do consumo de serviços, dentre outras.

Liu *et al.* (2012) cita que a eficiência da computação em nuvem tornou possível a redução dos custos operacionais de Tecnologia da Informação, mesmo para os casos de sistemas distribuídos sofisticados. Além disso, computação em nuvem confere maior agilidade às organizações, pois: (i) pode-se redimensionar rapidamente os recursos computacionais necessários para uma aplicação, de acordo com a demanda e (ii) pode-se prever com maior facilidade e transparência os custos de utilização dos serviços em nuvem. É importante ressaltar que as vantagens da computação em nuvem impulsionam o surgimento de inovações no projeto, arquitetura e desenvolvimento de software.

2.3.1 Tipos de Nuvens Computacionais

Neste trabalho, consideramos três os principais tipos de nuvens computacionais: públicas, privadas e híbridas. Embora todas as premissas da computação em nuvem estejam presentes em todas, cada uma delas possui características próprias.

Nuvens Públicas

Nuvens públicas, como é o caso da Amazon WS, Microsoft Azure e Rackspace, oferecem recursos computacionais praticamente ilimitados, economicamente viáveis e facilmente acessíveis, na forma de serviços via Internet. Em conjunto com a evolução da infraestrutura de rede de alta velocidade e da alta disponibilidade de Internet de banda larga, a computação em nuvem permitiu que as organizações migrassem seus *datacenters* (ou parte deles) para a Internet.

[4] https://aws.amazon.com/about-aws/whats-new/2006/08/24/announcing-amazon-elastic-compute-cloud-amazon-ec2—beta/

[5] http://opennebula.org

[6] http://openstack.org

Nuvens Privadas

Nuvens privadas também vêm ganhando popularidade tanto na indústria quanto na academia. Neste caso, os recursos computacionais que compõem a nuvem são de propriedade da organização, porém as premissas da computação em nuvem se mantém: entrega de recursos sob demanda, elasticidade, facilidade de acesso. As nuvens privadas oferecem às organizações uma forma de se consolidar seus ativos tecnológicos, como servidores, dispositivos de armazenamento e rede de dados. Além disso, alguns arcabouços de gerenciamento de nuvens também permitem a configuração de **datacenters** federados, abstraindo o uso de recursos computacionais em múltiplos *datacenters*. Uma outra vantagem das nuvens privadas é o isolamento que elas conferem às suas organizações. Por exemplo, as informações médicas de pacientes de uma clínica podem ser consideradas muito sensíveis para serem armazenadas em uma nuvem pública, portanto pode-se utilizar uma nuvem privada para armazená-las.

Nuvens Híbridas

Também existem as **nuvens híbridas**, que são a combinação de nuvens públicas e privadas. Este tipo de configuração confere às organizações as vantagens das duas modalidades de nuvens. Por exemplo, uma organização pode utilizar recursos de uma nuvem pública caso a capacidade de sua nuvem privada se esgote. Desta forma, uma aplicação executada em um **datacenter** local pode escalonar horizontalmente além das fronteiras da organização (*cloudbursting*).

2.3.2 Modelos de Serviço

Os provedores de computação em nuvem, sejam eles privados ou públicos, têm como objetivo final oferecer serviços computacionais a seus clientes internos ou externos. Owens (2010) classifica esses serviços em três modalidades distintas: Infraestrutura como Serviço (IaaS), Plataforma como Serviço (PaaS) e Software como Serviço (SaaS).

Infraestrutura como Serviço (IaaS)

No modelo de Infraestrutura como Serviço (*Infrastructure as a Service* - IaaS), os provedores de serviço oferecem computadores físicos ou virtuais, acessíveis remotamente via Internet. Além disso, é comum que ofereçam serviços adicionais, como: catálogo de imagens de máquinas virtuais, armazenamento de dados em bloco, armazenamento de arquivos e objetos, *firewalls*, balanceadores de carga, gerenciamento de IPs públicos e privados, redes de computadores virtuais (VLANs), GUI e CLI para o provisionamento de recursos e ferramentas para escalonamento automático.

No modelo de IaaS, o provedor de serviço entrega a infraestrutura computacional, deixando a cargo do usuário a instalação e administração dos sistemas operacionais e suas aplicações. O custo desta modalidade de serviço é proporcional a quantidade de recursos alocados e utilizados.

A Amazon é pioneira em IaaS públicas, porém desde o seu surgimento, outras companhias também começaram a oferecer este serviço (*e.g.* Google e Rackspace). Muitas organizações também vêm implantando nuvens privadas de IaaS. Nestes casos, o departamento de Tecnologia da Informação oferece IaaS para os outros departamentos da organização.

A infraestrutura computacional utilizada em nossa proposta é baseada em IaaS, ou seja, a plataforma de exploração de imagens médicas é inteiramente alicerçada em recursos computacionais virtualizados, como máquinas virtuais e dispositivos de armazenamentos.

Plataforma como Serviço (PaaS)

Plataforma como Serviço (*Platform as a Service* - PaaS) é uma modalidade que entrega ao usuário uma plataforma de execução de aplicações completa, que inclui o sistema operacional, ambiente de execução para uma ou mais linguagens de programação, banco de dados e servidores web.

A plataforma provê o gerenciamento automático da infraestrutura computacional e do ambiente de execução (máquinas virtuais, *firewalls*, servidores de aplicação, etc). No entanto, os usuários da nuvem precisam desenvolver suas aplicações conforme especificado pelo provedor de PaaS. Algumas plataformas também são capazes de redimensionar automaticamente os recursos utilizados, de acordo com a demanda do usuário.

Google, Heroku e Microsoft estão entre os principais provedores de PaaS.

Software como Serviço (SaaS)

Os provedores de Software como Serviço (*Software as a Service* - SaaS) oferecem acesso remoto a aplicações, abstraindo de seus usuários o gerenciamento da infraestrutura subjacentes. A escalabilidade das aplicações em nuvem é diferente dos outros tipos de aplicações, pois geralmente precisam suportar múltiplas locações (*multi-tenancy*) e grandes variações de demanda, ou seja, elas devem possuir um comportamento elástico: aumentar ou diminuir os recursos computacionais que alicerçam a aplicação conforme a necessidade.

Google Apps, Dropbox e Netflix são alguns dos exemplos mais conhecidos de SaaS para o público em geral. Para a comunidade de desenvolvedores, exemplos importantes são: New Relic, GitHub e Splunk.

As diferentes modalidades de serviços aqui apresentadas podem estar diretamente relacionadas. Por exemplo, um SaaS pode ser construído a partir de um IaaS, como é o caso do Netflix. Usuários também podem consumir serviços em nuvem em diversas modalidades ao mesmo tempo. Por exemplo, pode-se utilizar a Amazon AWS (IaaS) para provisionar máquinas virtuais que serão monitoradas por serviços como New Relic e Splunk (ambos SaaS).

Nossa proposta de plataforma de exploração de imagens médicas pode ser vista como um serviço de software na modalidade SaaS. A complexidade do WfMS, do gerenciamento de recursos distribuídos e da infraestrutura computacional subjacente é abstraída do usuário final.

2.3.3 Arquitetura da Nuvem

Com base no modelo tradicional de cinco camadas da arquitetura em grade, Foster *et al.* (2008) propuseram um modelo de arquitetura para computação em nuvem composto por quatro camadas: aplicação, plataforma, recursos unificados e malha computacional, como mostra a figura 2.11.

A camada de **malha computacional** (*fabric*) consiste de um conjunto heterogêneo de *hardware*, como servidores, equipamentos de rede e unidades de disco, interconectados por uma rede de computadores.

A camada de **recursos unificados** é composta por diversos tipos de recursos virtualizados, como máquinas virtuais, sistemas de arquivos e redes de computadores virtuais. Utilizando tecnologias de virtualização, esta camada abstrai os recursos da malha computacional subjacente e os expõe às camadas superiores como recursos virtuais.

A camada de **plataforma** é formada por ferramentas de gerenciamento de recursos e serviços de *middleware*, construída sobre a camada de recursos unificados. Bancos de dados, servidores web e DRMS são exemplos típicos de serviços nesta camada.

Por fim, a camada de aplicação representa artefatos de software que utilizam as camadas subjacentes para serem executados. Exemplos: sistemas de workflows, portais científicos, aplicações web, etc.

2.3.4 Gerenciamento de Recursos Distribuídos em Nuvem

Na seção 2.2, exploramos o papel dos DRMS em aglomerados de grades computacionais. Braghetto e Cordeiro (2014) argumentam que as versões mais recentes de muitas implementações de WfMS possuem algum tipo de adaptação para a computação em nuvem, porém com algumas limitações importantes. Segundo os autores, a prática mais comum é o provisionamento de uma

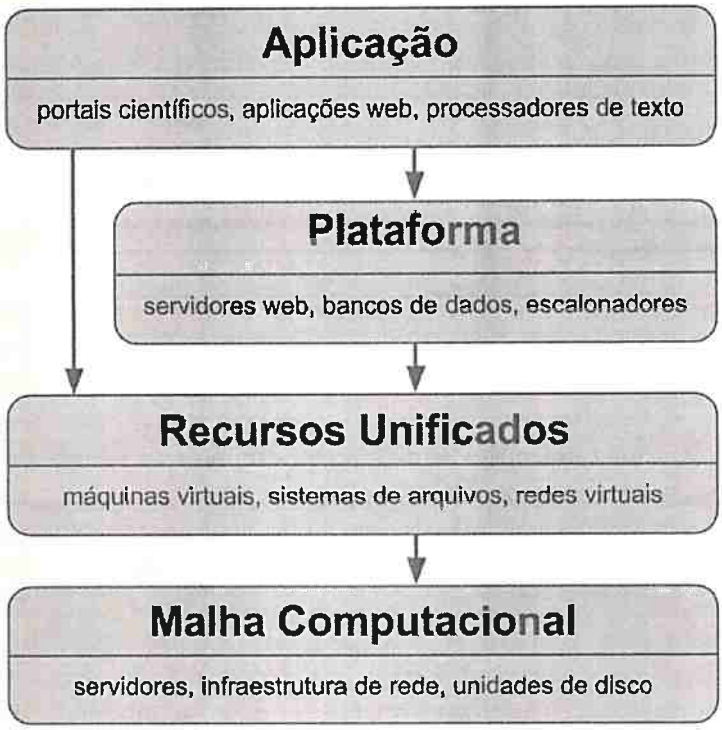

Figura 2.11: *Modelo de arquitetura para computação em nuvem.*

quantidade fixa de recursos computacionais de um provedor de IaaS, que serão gerenciados pelo DRMS como se pertencessem a um aglomerado ou grade computacional.

Tröger e Merzky (2014) chamam esta abordagem *VM-internal* (Interna à Máquina Virtual). Nela, o DRMS é instalado em máquinas virtuais em um provedor de IaaS, que passam a se comunicar da mesma forma como fariam em um aglomerado. Os autores indicam que essa prática, muito embora seja de simples implementação, apresenta um grande problema: o controle sobre os recursos computacionais é disputado pelo gerenciador de recursos da nuvem e pelo DRMS.

No entanto, defendemos que este problema não se aplica à maioria dos DRMS modernos, como Apache Mesos e HTCondor. Isso porque os gerenciadores de recursos da nuvem e os DRMS atuam em camadas arquiteturais distintas. O primeiro opera na camada de recursos unificados, e sua responsabilidade é controlar o provisionamento de máquinas virtuais, sistemas de arquivos e outros recursos virtuais. Já o DRMS atua na camada de plataforma, e sua responsabilidade é negociar e controlar o uso dos recursos provisionados.

Na abordagem *VM-internal*, também é possível explorar a capacidade elástica da nuvem, ou seja, pode-se aumentar ou diminuir a quantidade de recursos computacionais conforme a necessidade. Por exemplo, cada nó de execução pode ser provisionado como uma máquina virtual. Durante o processo de inicialização, o nó de execução se registra no DRMS, tornando-se parte do conjunto de recursos disponíveis. Esta estratégia, aliada a uma política de provisionamento automático de recursos, concede ao DRMS a elasticidade da nuvem. Vöckler *et al.* (2011) mostraram que essa abordagem é possível utilizando o gerenciador de recursos HTCondor juntamente com diversos provedores de IaaS.

A outra abordagem proposta por Tröger e Merzky (2014) é a chamada *VM-external* (Externa à Máquina Virtual), onde o processamento em lote se torna parte do conjunto de serviços disponíveis na nuvem computacional. Neste caso, as funcionalidades do negociador de recursos computacionais (mestre) são oferecidas como serviços pelo provedor de nuvem (SaaS). Os autores propõem uma ampliação do padrão OCCI (*Open Cloud Computing Interface*) para incorporar elementos da especificação DRMAA. O resultado final é um protocolo e API para controle e gerenciamento de processamento em lote, compatível com o padrão OCCI.

Em nosso trabalho, vamos adotar a abordagem* VM-internal, *pois além de tornar

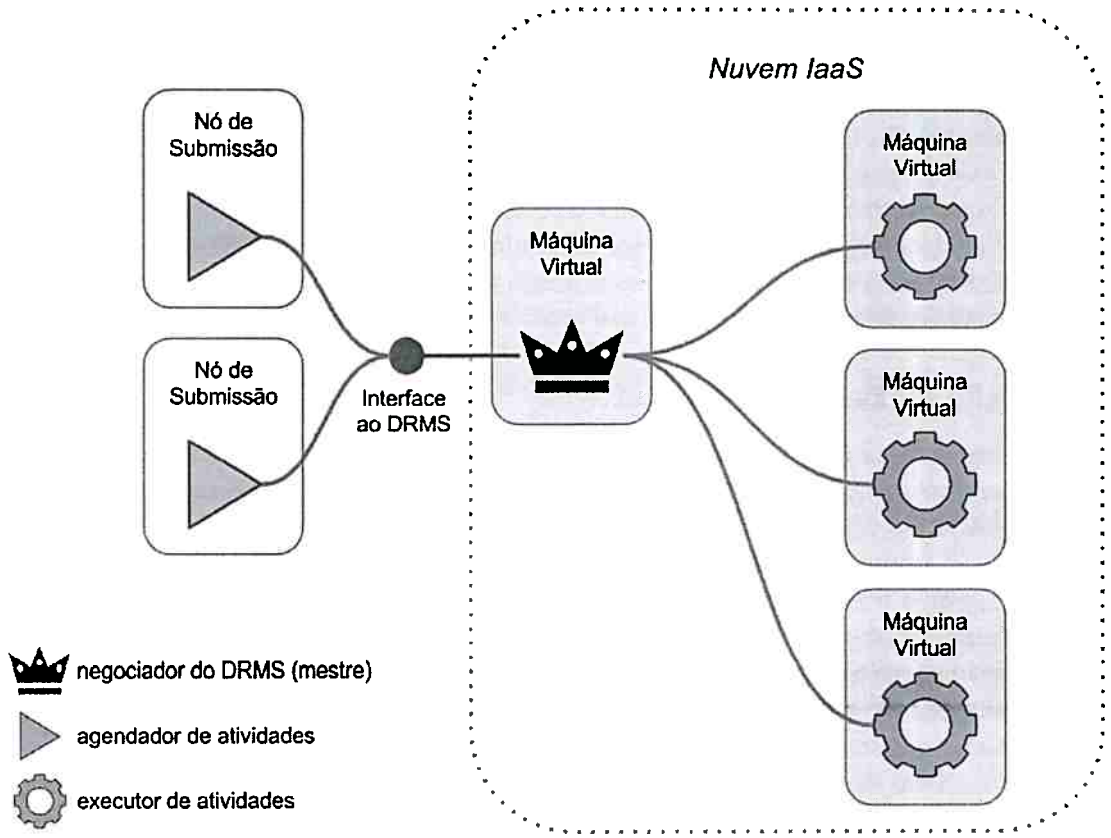

Figura 2.12: *Abordagem* VM-internal, *mostrando a comunicação entre os nós de execução e os executores de atividades, através de um negociador (mestre). Novas máquinas virtuais podem ser incorporadas ao DRMS, aumentando a capacidade de execução de atividades do conjunto. Os nós de submissão também podem ser máquinas virtuais dentro da nuvem IaaS.*

possível o processamento em lote em nuvens computacionais, não há necessidade de se alterar o protocolo e a API do arcabouço de gerenciamento de nuvem utilizado. Na prática, visamos desacoplar o DRMS do provedor de IaaS, possibilitando a portabilidade entre diferentes provedores (OpenNebula, OpenStack, Amazon AWS, Rackspace, etc).

2.4 Workflows para Processamento de Neuroimagem

Conforme vimos no capítulo 1, nossos objetivos são: i) propor uma solução baseada em nuvem para exploração de imagens médicas; e ii) identificar os cenários típicos de uso de workflows de análise dessas imagens para fins de pesquisa e rotina clínica.

Para atingir estes objetivos, examinaremos alguns exemplos de workflows aplicados a técnicas de imageamento em medicina. Mais especificamente, nos concentraremos nas aplicações em neuroimagem, uma disciplina da medicina e neurociência que objetiva a aquisição de imagens do sistema nervoso através de diversas modalidades de imageamento, como ressonância magnética (MRI), tomografia computadorizada (CT) e tomografia por emissão de pósitrons (PET).

Nesta seção visitaremos brevemente algumas das principais técnicas de aquisição e processamento de imagens médicas, e examinaremos um exemplo de workflow de análise estatística baseado neste tipo de imagem. Referenciamos Semmlow e Griffel (2014), Rangayyan (2005) e Smith e Webb (2010) para uma introdução mais abrangente à análise e processamento de imagens médicas.

2.4.1 Aquisição de Imagens Médicas

A aquisição de imagens médicas é a criação de representações gráficas do interior do corpo de um indivíduo para fins clínicos ou de pesquisa. Existem diversas modalidades de imagens médicas, cada uma envolvendo um método de aquisição diferente.

Imagens médicas podem ser classificadas em: i) **anatômicas**, que identificam as estruturas dos tecidos que compõe o corpo; e ii) **funcionais**, que permitem a investigação do metabolismo associado à anatomia (Maintz e Viergever, 1998).

A formação de imagens médicas parte da aquisição e processamento de sinais observados em uma sessão de imageamento. No caso da **ultrassonografia**, por exemplo, o sinal capturado é a reflexão das ondas mecânicas que incidem sobre os tecidos do corpo. Outro exemplo são as **imagens por projeção**, como é o caso dos raios-x, cujo sinal é produto da irradiação de feixes de fótons sobre o corpo, que sofrem atenuações devido à absorção total ou parcial nos diferentes tecidos que formam as estruturas do corpo humano. Os feixes atenuados são projetados em um plano, dando origem à imagem por raios-x.

Também há a tomografia computadorizada, que é capaz de produzir imagens tomográficas do corpo, ou seja, imagens que representam cortes axiais (planos perpendiculares ao maior eixo do corpo), coronais (planos longitudinais que dividem o corpo em partes posterior ou anterior) e sagitais (planos longitudinais que dividem o corpo em partes direita e esquerda). A figura 2.13 mostra exemplos de imagens nestes três planos. Para produzir estas imagens tomográficas, sinais são capturados ao redor da região do corpo estudada, em diferentes ângulos. Em seguida, utiliza-se algum método de reconstrução tomográfica para processar os sinais capturados, como as transformadas de Fourier e Radon.

Na tomografia computadorizada, o estudo ou análise de uma região do corpo requer a aquisição de múltiplas imagens tomográficas nos eixos axial, coronal e sagital. Em aplicações clínicas, estas imagens podem ser pré-processadas antes de serem analisadas pelo médico. Já em pesquisas científicas, por exemplo, as imagens de múltiplos pacientes podem ser processadas para a comparação e análise estatística de um dado grupo de indivíduos.

O foco deste trabalho são as modalidades derivadas da tomografia, mais especificamente a ressonância magnética (MRI), tomografia computadorizada (CT) e tomografia por emissão de pósitrons

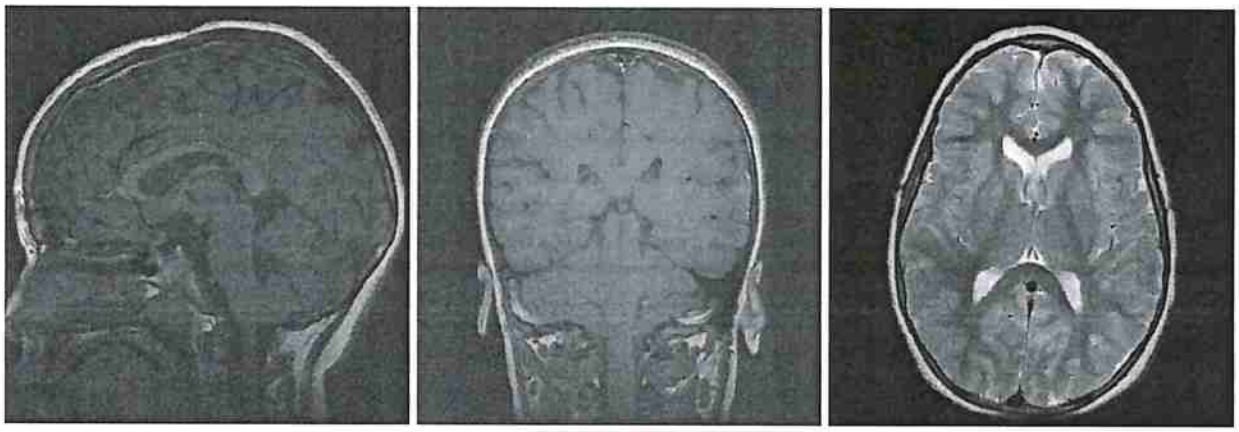

Figura 2.13: *Imagens por MRI nos planos (a) sagital, (b) coronal e (c) axial da cabeça de um paciente Rangayyan (2005).*

(PET). Cada uma destas modalidades podem ser utilizadas no imageamento de diferentes partes ou órgãos do corpo, como o cérebro e coração. As imagens geradas podem ser submetidas à diversas técnicas de processamento de imagens, com o objetivo de facilitar o diagnóstico médico ou extrair informações de forma automatizada. A figura 2.14 ilustra o uso de processamento de imagens às diversas modalidades aplicadas no estudo de diferentes órgãos e regiões do corpo.

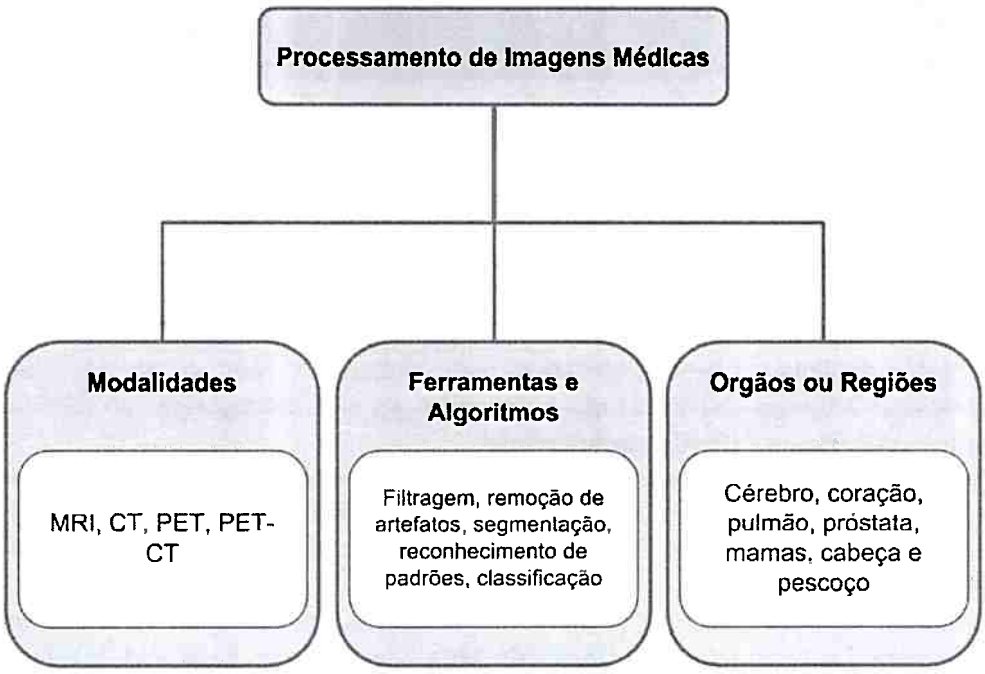

Figura 2.14: *Diferentes ferramentas e algoritmos de processamento de imagens médicas utilizadas em múltiplas modalidades aplicadas no estudo de vários órgãos.*

Em todas estas modalidades, o resultado do exame de uma região são uma série de cortes nos planos axial, coronal e sagital, ordenados pela posição do corte em relação ao volume. Além das imagens em si, um estudo é geralmente acompanhado de informações demográficas do paciente (*e.g.* sexo, idade, nome), além de informações sobre a sessão de aquisição (*e.g.* equipamento médico utilizado, data em que o estudo foi conduzido).

Ressonância Magnética

A ressonância magnética utiliza campos magnéticos de alta intensidade para polarizar os núcleos do átomo de hidrogênio na água presente nos tecidos do corpo humano, produzindo assim um sinal elétrico que pode ser medido. Os dados coletados são posteriormente processados para se reconstruir a imagem em duas ou três dimensões através da transformada de Fourier. MRI é uma modalidade intrinsecamente 3D, podendo-se adquirir imagens em qualquer plano axial, sagital, coronal ou oblíquo.

Para enfatizar estruturas anatômicas específicas ou anomalias presentes nos tecidos analisados, pode-se ponderar o contraste das imagens por MRI através da sequência dos pulsos de radiofrequência emitidos. Existem diversos parâmetros que podem ser utilizados para ponderar uma imagem por MRI, entretanto os mais relevantes para fins diagnósticos são os tempos de relaxação *T1* e *T2*, e PD (*densidade de prótons*).

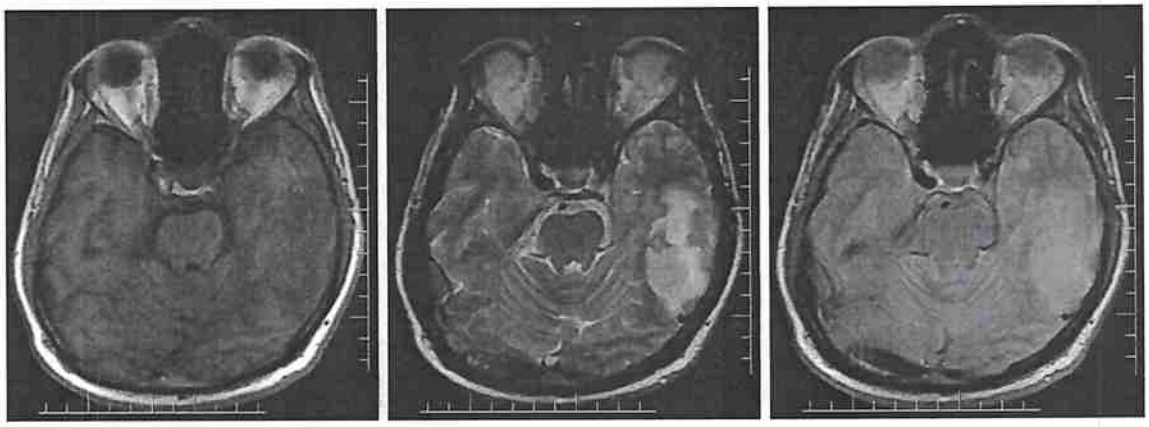

Figura 2.15: *Imagens de MRI ponderadas por T1 (à esquerda), T2 (centro) e PD (à direita). Imagens via Wikimedia Commons, por Nevit Dilmen.*

Com a ressonância magnética, também pode-se medir a difusão da água, ou seja, o movimento de translação das moléculas da água em qualquer direção. Porém quando não há restrição a este movimento, a difusão da água não possui nenhum significado especial, pois as medições terão o mesmo resultado em todas as direções. Neste caso temos a **difusão isotrópica**. Entretanto, quando analisamos a difusão da água em tecidos biológicos como músculos e cérebro, as moléculas de água tendem a se propagar ao longo das fibras que os compõe, ou seja, há restrição ao movimento das moléculas, o que caracteriza a **difusão anisotrópica**.

Através da medição da difusão anisotrópica, aliada a um modelo matemático fundamentado no cálculo de tensores, pode-se determinar a orientação das fibras do tecido estudado. O MRI de difusão baseado neste modelo matemático é conhecido como **aquisição de imagem por tensor de difusão** ou DTI.

Em termos práticos, uma aquisição por DTI dá origem a uma imagem onde **cada voxel** é uma matriz composta dos parâmetros de difusão da água naquele local. Essa matriz é formada por vetores e escalares, que podem ser utilizados para criar visualizações clinicamente úteis que nos permitam a exploração da anatomia ou função do órgão estudado (Mori e Tournier, 2013).

As formas mais comuns para representar imagens de DTI em escala de cinza são o **coeficiente de difusão aparente** (*Apparent Diffusion Constant*, ou ADC) e os **mapas de anisotropia**, dos quais os mais utilizados são a anisotropia fracional (FA), a difusividade média (MD) e a difusividade radial (RD) (Mori, 2007a,b). A figura 2.16 mostra alguns exemplos de mapas de difusão obtidos por DTI.

Além do mapas de cores, como mostra o exemplo (f) da figura 2.16, também há uma outra importante maneira de se representar a orientação da difusão: a **tratografia**. Nesta técnica, regiões com alto índice de anisotropia, como os tratos, são concatenados, permitindo o rastreamento das vias que ligam duas regiões (Jackowski *et al.*, 2005). A figura 2.17 mostra um exemplo de tratografia.

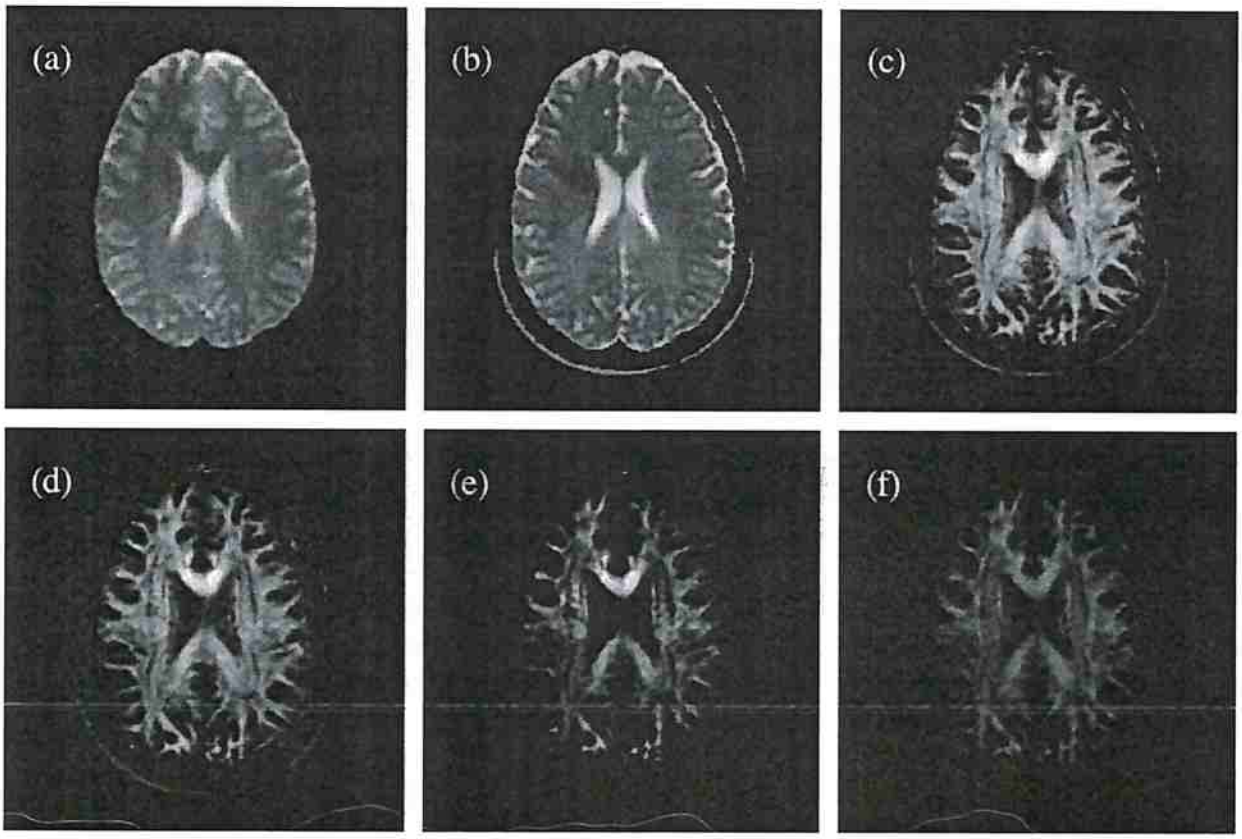

Figura 2.16: *Exemplos de contrastes diversos obtidos por MRI: (a) imagem ponderada por T2; (b) mapa de ADC; (c) mapa de FA; (d) mapa de anisotropia relativa; (e) mapa de relação de volume; (f) mapa de cores representando a orientação das fibras - vermelho representa a direção esquerda-direita, verde é a direção anterior-posterior e azul é a direção superior-inferior (Mori e Tournier, 2013).*

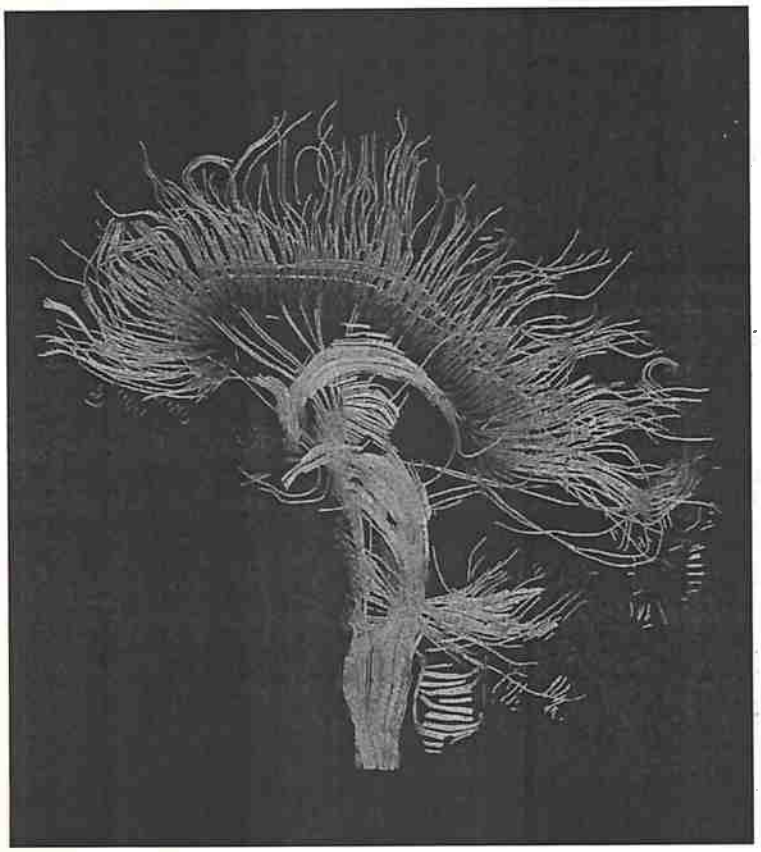

Figura 2.17: *Imagem de tratografia, mostrando os tratos do plano sagital médio do cérebro humano - imagens via Wikimedia Commons, por Thomas Schultz.*

Tomografia Computadorizada

Podemos também obter imagens tomográficas a partir da aquisição de raios-X transversais ao redor do órgão ou região examinada. O resultado são projeções por raios-x de múltiplos ângulos, que são computacionalmente processados através de uma técnica de reconstrução tomográfica, como vimos nas subseções anteriores. Após a reconstrução, obtém-se uma série de seções axiais do órgão estudado. Seções em outros planos (coronal, sagital ou oblíquo), bem como a reconstrução em três dimensões são possíveis apenas através da computação das imagens axiais do volume examinado. Esta é a principal diferença em termos de reconstrução tomográfica com relação à modalidade de MRI (Rangayyan, 2005).

A figura 2.18 mostra exemplos de visualizações produzidas a partir de CT, com os usuais planos axial, coronal e sagital, bem como a reconstrução do volume em três dimensões. Os planos coronal e sagital são reconstruídos a partir do plano axial. A renderização do volume em três dimensões requer grande poder computacional, e geralmente não é utilizado para fins diagnósticos. Para este fim, as imagens nos planos axial, coronal, sagital e oblíquos são geralmente utilizadas pelos radiologistas.

Além das reconstruções em duas ou três dimensões, também são comuns outros tipos de processamento para auxílio no diagnóstico ou para fins de pesquisa. Alguns exemplos incluem segmentação de regiões de interesse, registro ou alinhamento de imagens e volumes, detecção automática de características e classificação.

Tomografia por Emissão de Pósitrons

Outra importante modalidade de imagens médicas é a tomografia por emissão de pósitrons, ou PET, uma técnica de imageamento por medicina nuclear. A grande vantagem de imagens por medicina nuclear é sua capacidade de capturar aspectos funcionais do corpo humano, muito embora as imagens capturadas apresentem baixa resolução espacial e alta susceptibilidade à ruídos. A

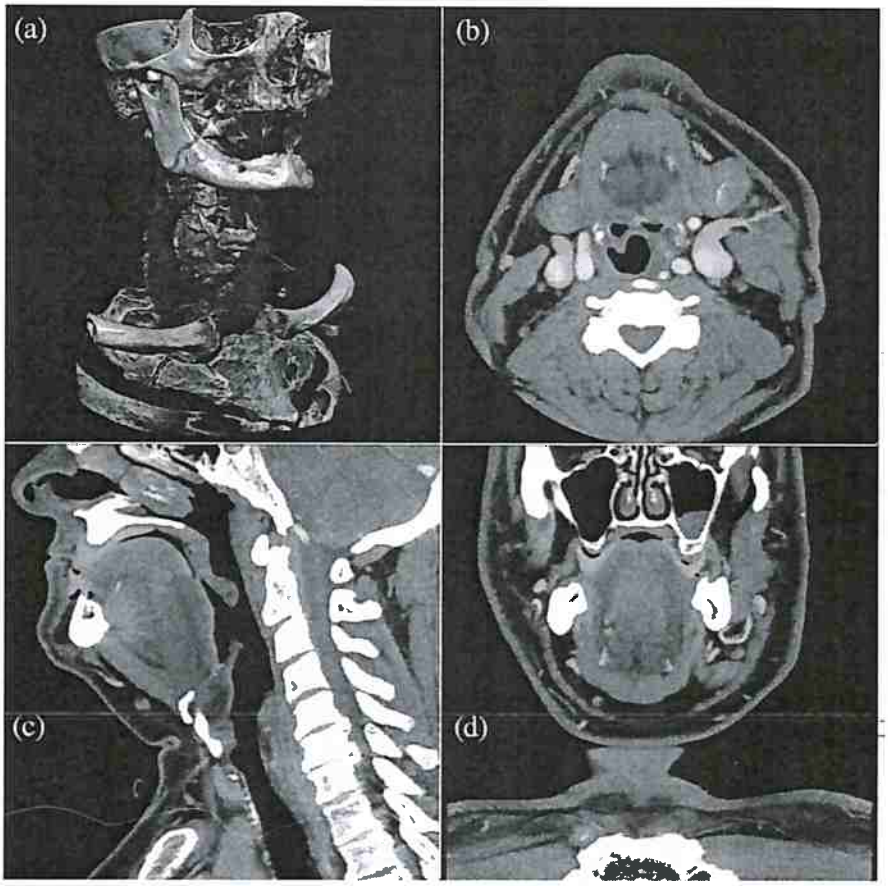

Figura 2.18: *Imagens por CT: (a) representação do volume reconstruído a partir da aquisição original; (b), (c) e (d) imagens de cortes nos planos axial, coronal e sagital, respectivamente - imagens via Wikimedia Commons.*

aquisição deste tipo de imagem envolve o uso de radiofármacos que são projetados para serem absorvidos e concentrados nos órgãos ou regiões de interesse do corpo humano (Rangayyan, 2005).

O radiofármaco utilizado em sessões de aquisição por PET é um isótopo que gera pósitrons como resultado de seu decaimento radioativo. Quando emitido, um pósitron interage com um elétron, causando a aniquilação de ambos e a emissão de exatos dois fótons de radiação gama em direções opostas. Estes fótons são coletados pelo equipamento médico, transformados em sinais elétricos e processados por um algoritmo de reconstrução tomográfica objetivando a obtenção de imagens em duas e três dimensões.

O PET também é comumente utilizado em conjunto com CT ou MRI, objetivando a fusão das duas modalidades em uma única imagem. A operação de fusão de imagens de duas modalidades diferentes tem grande utilidade na medicina diagnóstica, e geralmente envolve algoritmos de processamento de imagens (*e.g.* registro de imagens, operadores morfológicos, filtros *wavelet*, dentre outros) (James e Dasarathy, 2014). A figura 2.19 mostra a imagem resultante da fusão de PET e MRI. Há também equipamentos híbridos capazes de capturar imagens por PET e por CT em uma única sessão de aquisição.

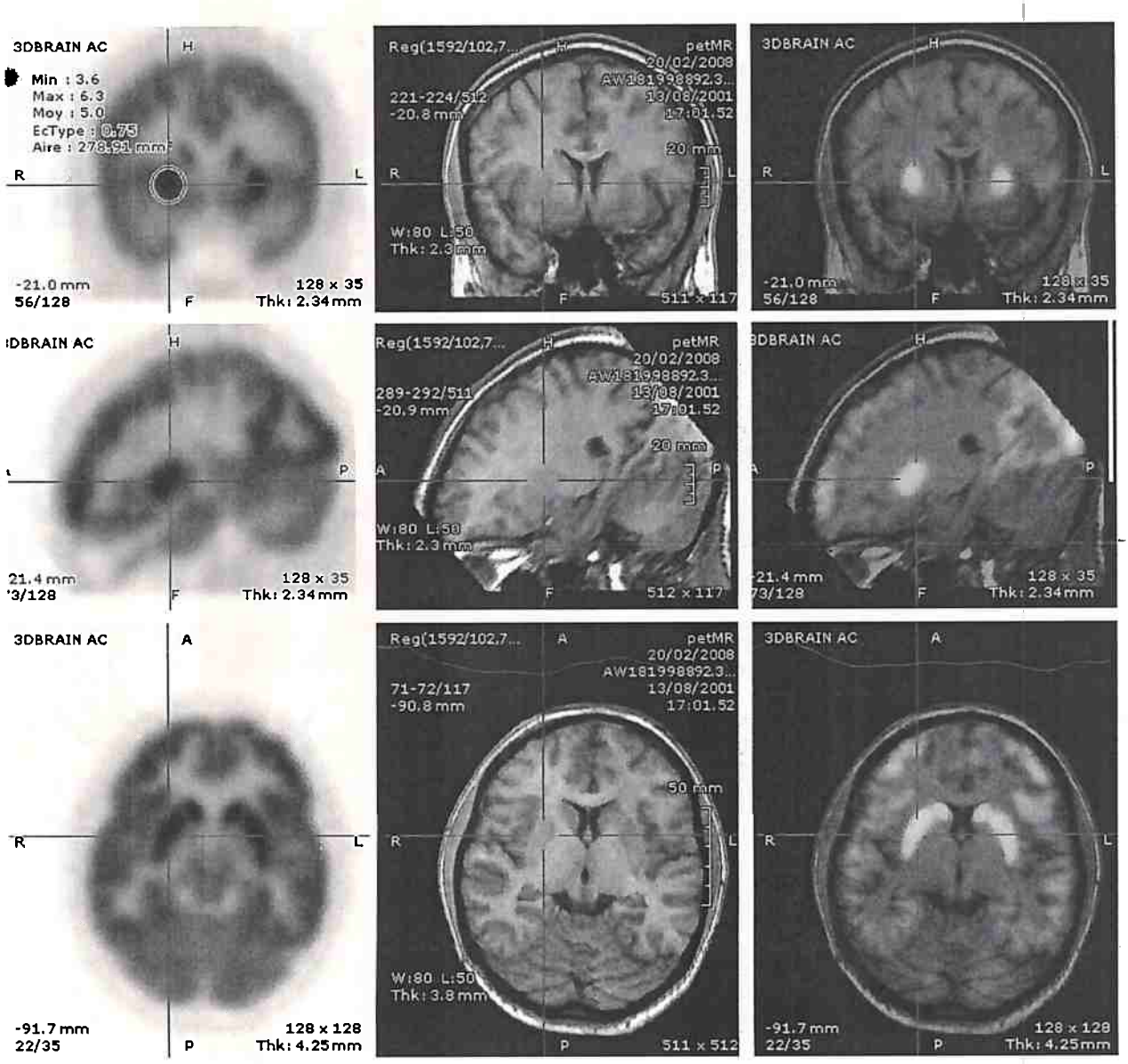

Figura 2.19: *Imagens axial, coronal e sagital obtidas por PET (primeira coluna), MRI (segunda coluna) e pela combinação das duas (terceira coluna) - imagens via Wikimedia Commons.*

2.4.2 Workflows de Análise de Imagens Médicas

Em geral, podemos analisar imagens médicas por **regiões de interesse** (ROI), **voxel-a-voxel** (*Voxel-Based Analysis*, VBA) ou vértica-a-vértice (para análise das superfícies de diferentes tecidos). Para as análises por ROI, a região estudada é delimitada semiautomática ou manualmente, sendo necessário o conhecimento prévio da região estudada (Bezerra *et al.*, 2012). Por outro lado, as análises por VBA são automáticas e globais, ou seja, são capazes de considerar o cérebro como um todo, sem o conhecimento prévio da região de interesse (Smith *et al.*, 2006). A morfometria baseada em voxel (*Voxel-Based Morphometry*, VBM) é a abordagem VBA mais utilizada para estudar as substância branca e cinzenta do cérebro, porém também existem outras abordagens específicas (e mais adequadas) para a análise de imagens de modalidades específicas. Por exemplo, o método de estatística espacial baseada em tratos (*Tract-Based Spatial Statistics*, TBSS), proposta por Smith *et al.* (2006), é mais adequado para imagens por DTI - os autores afirmam que a TBSS favorece a correta interpretação de dados através de um algoritmo de registro fundamentado no mapa de anisotropia fracional (FA) das imagens registradas. Outra implementação do método de VBM é o mapeamento estatístico paramétrico (*Statistical Parametric Mapping*, SPM), inicialmente desenvolvido para testar hipóteses estatísticas em imagens médicas funcionais, mas que também podem ser utilizado em imagens anatômicas.

Análises por ROI limitam a quantidade de regiões estudadas e, por consequência, há o risco de se negligenciar regiões importantes. Por outro lado, análises voxel-a-voxel e vértice-a-vértice permitem que toda a imagem seja estudada. Essa é uma das principais vantagens da VBA.

De modo geral, as abordagens de análise por ROI, VBA e vértice-a-vértice visam estimar a distribuição de probabilidade das intensidades dos voxels para gerar mapas estatísticos para: (i) determinar diferenças relevantes entre grupos de indivíduos com condições distintas; ou (ii) testar hipóteses sobre estes indivíduos e suas condições. (Fernandes *et al.*, 2011). A figura 2.20 mostra as etapas necessárias para a criação destes mapas estatísticos.

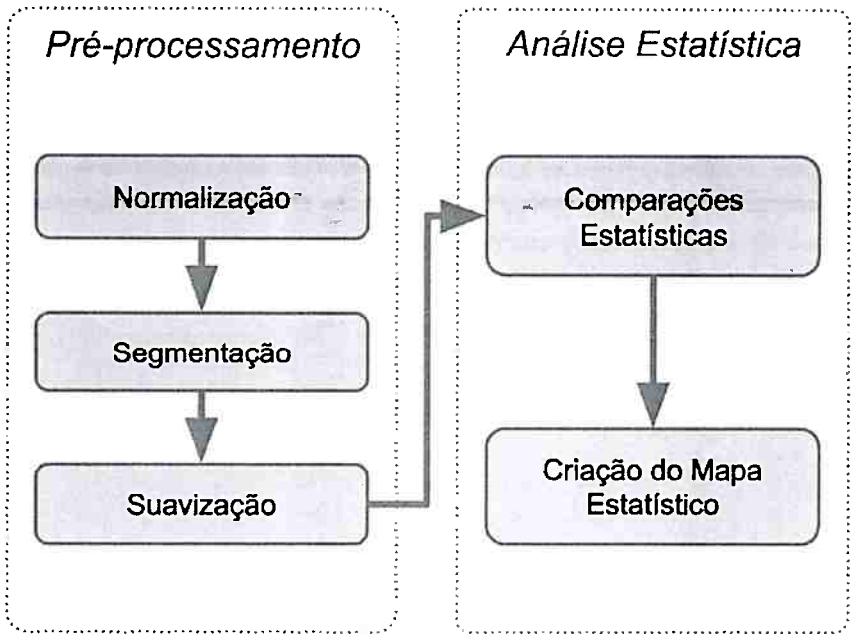

Figura 2.20: *Etapas necessárias para a criação de mapas que refletem diferenças anatômicas estatisticamente significantes entre os grupos de pacientes.*

Antes da criação dos mapas estatísticos, é necessário que cada imagem seja submetida a uma etapa conhecida como **pré-processamento**. Nesta etapa, as imagens são **normalizadas**, **segmentadas** e **suavizadas**, objetivando torná-las comparáveis. É importante notar que o pré-processamento varia de acordo com o método de análise que se deseja usar.

A **normalização** pode envolver uma ou mais das seguintes transformações: (i) orientação tem-

poral das imagens de cada série (*slice-timing*); (ii) correção dos artefatos gerados por movimentos do paciente, correntes de Foucault, etc; e (iii) alinhamento das imagens dos diferentes indivíduos para um mesmo espaço anatômico de referência (como o atlas do *Montreal Neurological Institute*, MNI);

Em seguida, as imagens são submetidas à **segmentação**, visando a extração das regiões de interesse para o estudo.(Duncan *et al.*, 2004). Essa fase também varia de acordo com a relevância que um determinado estudo atribui a uma dada região. Há diversos algoritmos e métodos de segmentação de imagens médicas, e para cada região que se deseja segmentar deve-se escolher o mais adequado. Por exemplo, a extração do cérebro do restante do crânio pode ser realizada através do algoritmo BET (*Brain Extraction Tool*), proposto por Smith (2002). Já a substância cinzenta do cérebro pode ser segmentada por meio do algoritmo FAST4 (Zhang *et al.*, 2001). Outras suites de ferramentas como o FreeSurfer, também possuem outras estratégias e algoritmos para realizar essas operações.

No último estágio do pré-processamento ocorre a **suavização** das regiões segmentadas, aplicando-se um filtro Gaussiano. São três as principais motivações para a suavização das imagens: (i) reduzir as variações de intensidade nos voxels da imagem devido às diferenças entre os tecidos biológicos (*e.g.* substâncias branca e cinzenta do cérebro); (ii) fazer com que a distribuição de intensidades dos voxels se torne mais próxima de uma distribuição normal; e (iii) atenuar os erros resultantes do processo de alinhamento das imagens. Em resumo, a suavização objetiva aumentar a eficiência da análise estatística que será aplicada na fase posterior (Mechelli *et al.*, 2005).

No caso do FreeSurfer, também há a possibilidade de se criar reconstruções em três dimensões dos diferentes tecidos do cérebro, como mostra a figura 2.21. As superfícies são formadas por vértices, como mostra a figura 2.22. Durante o processo de reconstrução, informações como espessura e volume cortical são mapeadas às posições dos vértices. As análises vértice-a-vértice se referem a comparações destas informações.

A **análise estatística** é alimentada com os resultados da fase de pré-processamento. Para gerar os mapas estatísticos, realiza-se comparações voxel-a-voxel ou por ROI entre as imagens dos grupos de pacientes estudados, utilizando o teste t de *Student* e a abordagem TBSS, por exemplo (Bezerra *et al.*, 2012). Os mapas criados refletem as diferenças anatômicas estatisticamente significantes entre os grupos de pacientes (*e.g.* grupos de controle e o de pacientes estudados) (Fernandes *et al.*, 2011).

No capítulo a seguir, combinaremos os conceitos de workflows científicos e médicos e gerenciamento de recursos computacionais na nuvem, objetivando criar uma solução para exploração de imagens médicas capaz de executar análises complexas como a que apresentamos nesta seção.

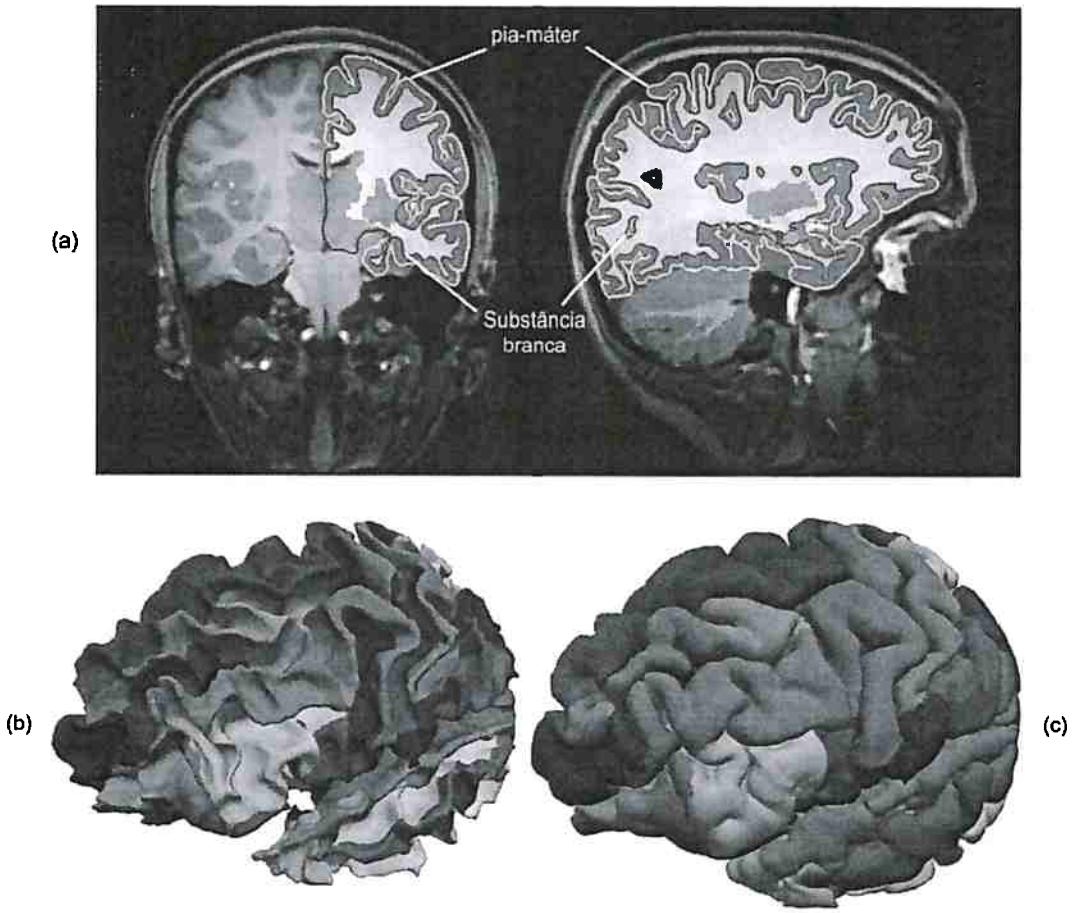

Figura 2.21: *Exemplo de reconstruções corticais realizadas pelo FreeSurfer: (a) volume cerebral, com o contorno da superfície da pia-máter em amarelo e da superfície entre a substância branca e cinzenta do cérebro; (b) reconstrução em três dimensões da superfície entre a substância branca e cinzenta; (c) reconstrução em três dimensões da superfície da pia-máter; (b) e (c) mostram o atlas de Destrieux com cores diferentes para cada região.*

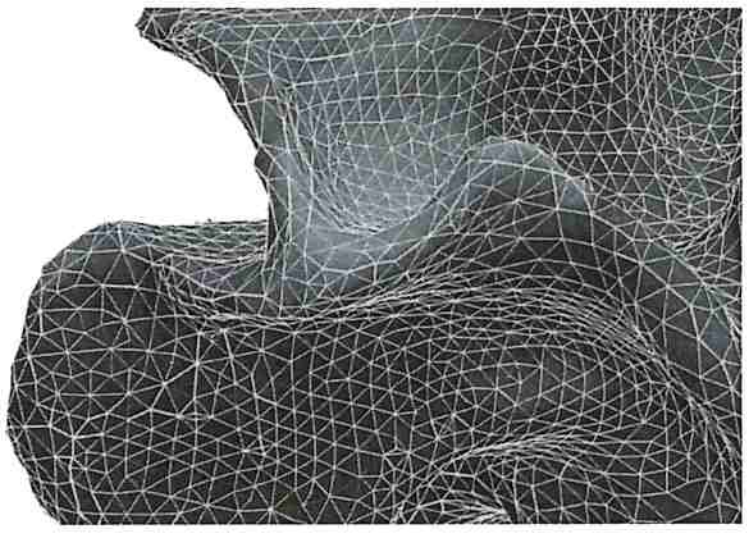

Figura 2.22: *Superfície do FreeSurfer e os vértices que a compõe.*

Capítulo 3

Metodologia

Conforme visto nos capítulos anteriores, este trabalho contempla os seguintes objetivos:

(1) identificar exemplos típicos de uso de workflows de análise de neuroimagem;

(2) identificar os requisitos funcionais e não funcionais da solução para exploração de imagens médicas;

(3) propor uma solução baseada em nuvem para exploração dessas imagens, capaz de executar os workflows identificados em (1);

Para tal, examinamos: i) o uso de workflows no CHU; e ii) a metodologia para processamento de imagens médicas utilizada em publicações relacionadas ao projeto ABIDE (Di Martino *et al.* , 2014).

Nossas observações no CHU foram importantes para a elaboração do fluxo de imagens desde sua aquisição até o início da análise, como veremos na seção 3.2.1. Porém, utilizamos workflows relacionados ao projeto ABIDE (Di Martino *et al.*, 2014) para nortear nossa solução para exploração de imagens médicas, bem como para verificação desde trabalho. Na próxima seção abordaremos o uso do ABIDE em nossa pesquisa.

3.1 Exemplos típicos de uso de workflows de análise de neuroimagem

O ABIDE (*Autism Brain Imaging Data Exchange*) promove o estudo do Transtorno do Espectro Autista (TEA) através da construção de um repositório de dados clínicos e imagens do cérebro de milhares de indivíduos com e sem TEA. Diversas publicações utilizam estes dados para tanto para a pesquisa o TEA, quanto para o desenvolvimento de novos biomarcadores técnicas de processamento e analise de imagens. Selecionamos três destes trabalhos para guiar nossa pesquisa: Schaer *et al.* (2015) investiga as diferenças no volume cortical em homens e mulheres diagnosticados com TEA, enquanto que Lefebvre *et al.* (2015) e Kucharsky Hiess *et al.* (2015) analisam o autismo através das diferenças no corpo caloso e volume cerebral de pacientes com e sem TEA.

Todos os três exemplos citados acima utilizam o software de análise *FreeSurfer* para pré-processar as imagens de ressonância magnética estrutural do cérebro de cada indivíduo e em seguida extrair informações como os volumes cerebral, intracraniano e do corpo caloso. Através desses dados, são realizadas comparações entre grupos, como pacientes masculinos *versus* pacientes femininos diagnosticados com TEA.

Por exemplo, em *Sex differences in cortical volume and gyrification in autism*, Schaer *et al.* (2015) selecionou 210 indivíduos dos 1112 participantes do projeto ABIDE cujas imagens de MRI ponderadas por T1 foram adquiridas. Nesse processo de seleção, algumas das aquisições foram desconsideradas de acordo com os critérios de controle de qualidade que o estudo utilizou. Para cada um dos participantes, realizou-se a reconstrução cortical através de algoritmos presentes no

FreeSurfer (Fischl *et al.*, 2002). De modo geral, a reconstrução envolveu os seguintes passos: i) extração dos tecidos cerebrais, removendo o crânio e o líquido; ii) segmentação das estruturas subcorticais; iii) extração das superfícies corticais (Dale *et al.*, 1999). O resultado do processo de reconstrução são os volumes cortical, subcortical, da substância branca do cérebro e supratentorial. As coordenadas dos volumes e superfícies foram registradas no espaço padrão *fsaverage*. Para estimar o efeito do sexo do paciente, os autores utilizaram um modelo linear geral (MLG).

Este tipo de workflow é comum em estudos que realizam análises estatísticas entre dois ou mais grupos. Em nosso trabalho, utilizamos um dos workflows usados por Schaer *et al.* (2015) para verificar se há diferenças estatísticas significativas no volume ou espessura cortical entre os grupos: homens com e sem TEA, mulheres com e sem TEA.

Notamos que os três trabalhos citados descrevem de modo geral o processo de análise, sem entrar em detalhes no modelo de regressão utilizado para se chegar às conclusões apresentadas (*e.g.* contrastes e matriz de planejamento). Também notamos que a lista de indivíduos considerados na análise geralmente não é publicada neste tipo de trabalho, o que dificulta a reprodução dos experimentos.

3.2 Requisitos da Solução para Exploração de Imagens Médicas

Sob uma perspectiva de alto nível, este trabalho abrange os requisitos gerais listados na tabela 3.1.

RG-1	**Recepção e organização** de imagens médicas: (i) adquiridas por equipamentos médicos, como MRI, CT, PET-CT, etc; (ii) presentes em um sistema de arquivamento de imagens (PACS); (iii) presentes no computador pessoal do usuário.
RG-2	**Modelagem de workflows** para exploração e análise de imagens médicas através de uma **GUI de fácil acesso**, que chamaremos aqui de **bancada virtual**. As ferramentas computacionais necessárias para realizar as análises estarão disponíveis por intermédio desta GUI.
RG-3	Processamento de **grandes quantidades de dados** utilizando a elasticidade da **computação em nuvem.**

Tabela 3.1: *Visão geral de alto nível dos requisitos da solução de software para exploração e análise de imagens médicas na nuvem.*

Na seção 3.2.1 discutiremos a recepção e organização de dados de imagens médicas. Em seguida, na seção 3.2.2, investigaremos os requisitos de um WfMS para análise de imagens médicas. Por fim, na seção 3.2.3 veremos como executar estas análises na nuvem.

3.2.1 Recepção e Organização de Imagens Médicas

Um dos principais desafios em hospitais, laboratórios e instituições de pesquisa é a organização e distribuição eficiente de dados médicos para a equipe de cientistas, médicos e biomédicos. No CHU, por exemplo, a preocupação com os dados se estende desde sua aquisição até sua utilização. A tabela 3.2 reúne os requisitos específicos relecionados ao requisito **RG-1**:

No próximo capítulo, apresentaremos a arquitetura de uma solução que contempla os requisitos identificados acima.

3.2.2 Modelagem de Workflows para Análise de Imagens Médicas

Vimos que workflows de processamento de imagens são compostos por diversas atividades complexas que utilizam ferramentas computacionais para processar e transformar dados. Quando a análise é realizada através de *scripts shell*, o usuário precisa conhecer os detalhes dos programas

RE-1.1	**Recepção** de imagens médicas enviadas por usuários, equipamentos médicos e outros sistemas.
RE-1.2	**Armazenamento e gerenciamento** de *imagens* médicas e seus *metadados*.
RE-1.3	**Busca** por imagens médicas de forma amigável.
RE-1.4	**Organização** das imagens médicas em projetos.
RE-1.5	**Distribuição** das imagens médicas para uso em análises.

Tabela 3.2: *Requisitos para recepção e organização de imagens médicas.*

executados, bem como a infraestrutura computacional que efetivamente os executará (um servidor dedicado, um computador pessoal, etc). Portanto, podemos classificar nível de abstração deste modelo do workflow como **concreto**.

Para elevar o nível de abstração do modelo de workflow, precisamos adotar um WfMS que seja capaz de interpretar um modelo abstrado de workflow, e então transformá-lo em um modelo concreto imediatamente antes de sua execução. Desta forma, tornaremos o uso deste workflow mais acessível. A figura 2.20 ilustra um exemplo de modelo abstrato para workflows.

Observamos que os modelos de workflows usados em diversos trabalhos (*e.g.* Schaer *et al.* (2015), Lefebvre *et al.* (2015) e Kucharsky Hiess *et al.* (2015)) possuem as seguintes características:

(1) podem ser representados por DAGs;

(2) são fortemente orientados a dados;

(3) as estruturas dos tipos *pipeline* e agregação são predominantes;

Note que workflows de análise de imagens médicas implementados como *scripts shell* comumente apresentam laços que iteram sobre coleções de dados. Estes casos podem ser substituídos por estruturas do tipo agregação, redução ou redistribuição sem perda de semântica. De fato, este tipo de substituição contribui para o aumento do paralelismo no workflow, pois os processos contidos dentro do laço deixam de ser executados sequencialmente.

O sistema de workflows também deve ser capaz de reconhecer e manipular imagens médicas de diversos formatos e seus metadados. No formato DICOM, por exemplo, as imagens de uma sessão de MRI são armazenadas em múltiplos arquivos, cada um representando um corte tomográfico do volume estudado. Já o formato NIfTI pode agrupar todos os cortes de todas as sessões de um paciente em um só arquivo.

Outro importante requisito é a capacidade de executar um workflow (ou parte dele) para analisar um conjunto de imagens. Por exemplo, dado um workflow de conversão de formatos de imagem, o WfMS deve ser capaz de executá-lo para um conjunto de n imagens, paralelamente. Esse mecanismo deve ser transparente para o usuário, como mostra a figura 3.1.

Também é importante que se consiga reduzir essas linhas de execução paralelas, de modo a sincronizá-las novamente em uma única linha de execução, como é o caso representado na figura 3.2.

Por fim, é necessário que uma única atividade possa agrupar um conjunto de instruções de execução, a fim de facilitar seu uso. Por exemplo, o código abaixo realiza a correção de movimento de uma imagem por DTI:

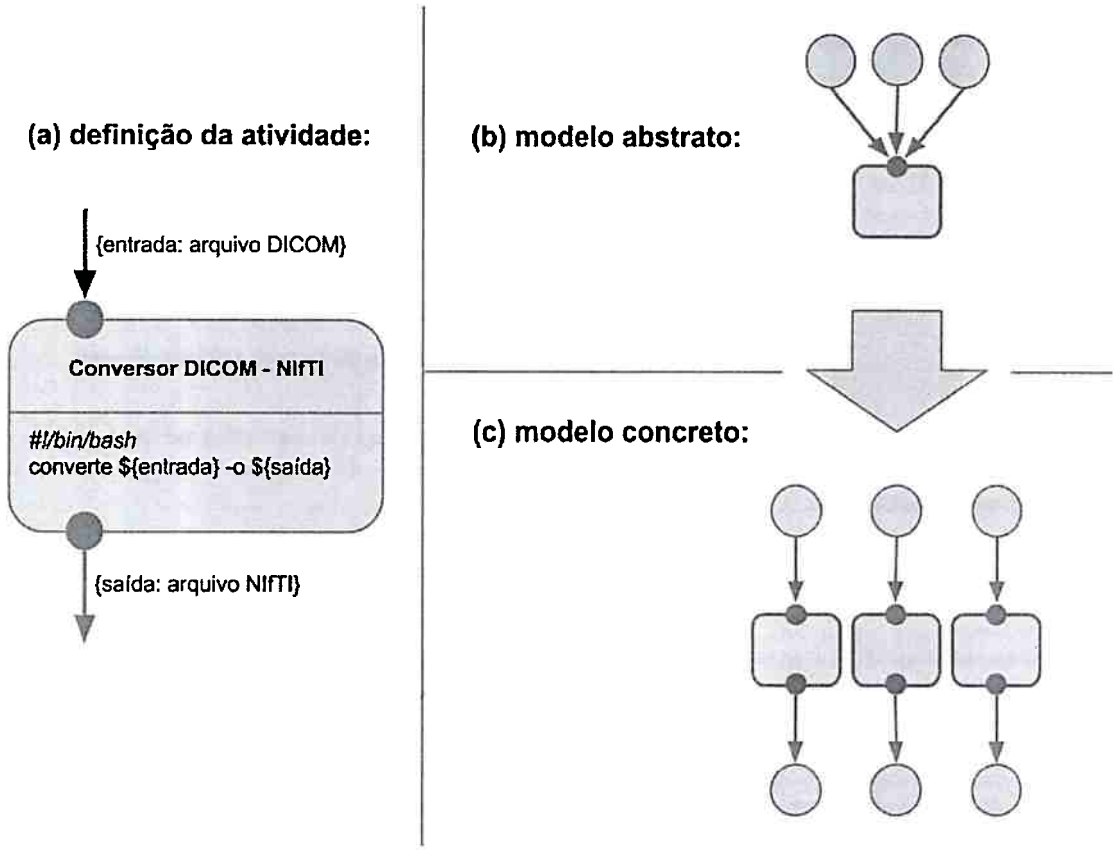

Figura 3.1: *Exemplo de paralelismo automático: dada uma atividade definida por (a), pode-se utilizá-la para processar uma coleção de dados (b). Em (c) temos a concretização deste cenário, que é a execução da atividade (a) para cada um dos dados na coleção.*

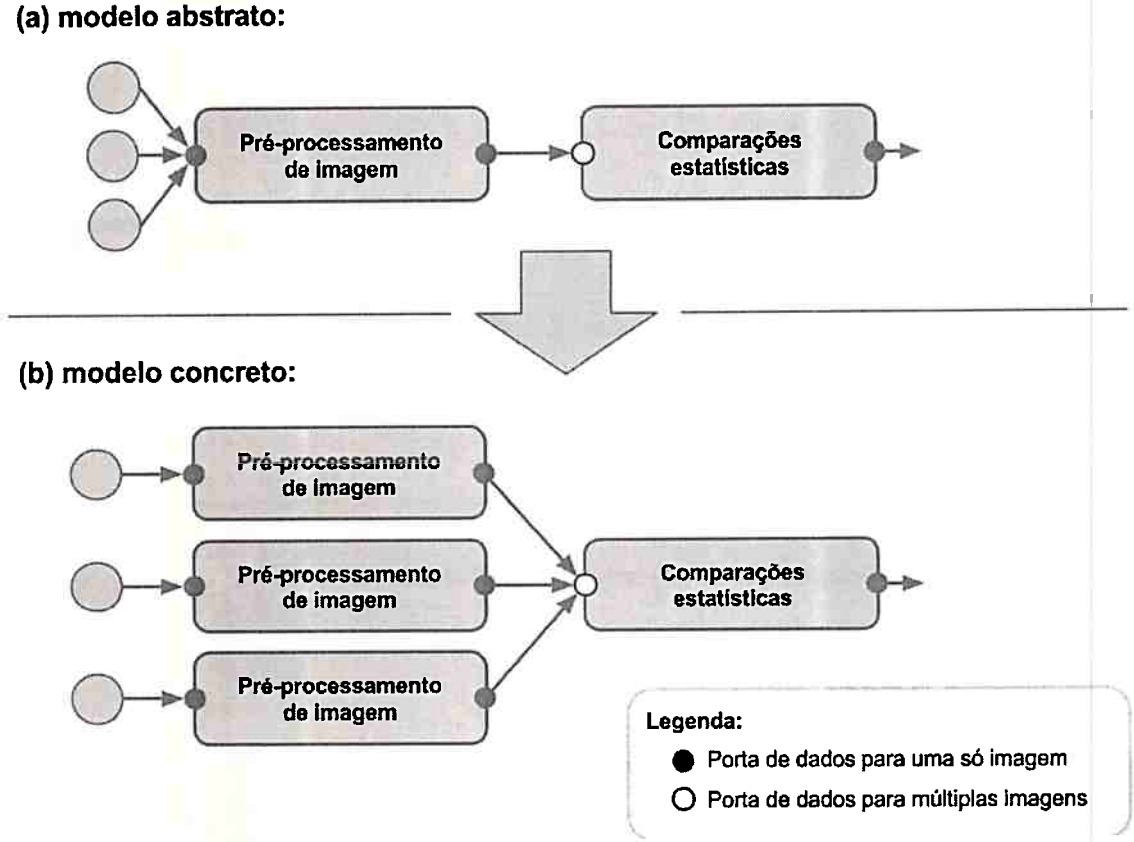

Figura 3.2: *Exemplo de sincronização automática: o modelo abstrato (a) se concretiza no modelo (b), onde várias linhas de execução se convergem para uma única atividade capaz de consumir múltiplas entradas de dados, e produzir uma única saída.*

```
1 # Estima  o de par metros de movimento
2 3dvolreg -verbose -Fourier -prefix IMAGEM_CORRIGIDA.nii.gz -base 0 \
3     -dfile PAR METROS_DE_CORRE O.txt IMAGEM_ORIGINAL.nii.gz
4
5 # Calcula a m dia de T2, reconstr i os dados e ajusta os autovetores e
      autovalores
6 3dTstat -mean -prefix IMAGEM_MDIA_DE_T2 IMAGEM_CORRIGIDA+orig[0..3]
7 3dTcat -prefix IMAGEM_T2_EM_32_DIRE ES IMAGEM_MDIA_DE_T2_+orig \
8     IMAGEM_CORRIGIDA+orig[4..35]
9 1d_tool.py -infile IMAGEM_ORIGINAL.bvec[3..35] -write IMAGEM_EM_32_DIRE ES.bvec
10 1d_tool.py -infile IMAGEM_ORIGINAL.bval[3..35] -write IMAGEM_EM_32_DIRE ES.bval
11 3dAPATItoNIFTI -prefix IMAGEM_T2_EM_32_DIRE ES.nii.gz \
12     IMAGEM_T2_EM_32_DIRE ES+orig
13
14 # O resultado final s o os arquivos IMAGEM_T2_EM_32_DIRE ES.nii.gz (arquivo
      NIfTI), IMAGEM_EM_32_DIRE ES.bvec (autovetores) e IMAGEM_EM_32_DIRE ES.
      bval (autovalores)
```

Para o cientista, não importa quais programas são efetivamente executados, pois seu objetivo é apenas efetuar a correção de movimento em uma dada imagem. Portanto, a atividade "correção de movimento" poderia ser definida como uma operação atômica, ou seja, no ponto de vista do usuário, a atividade possui três parâmetros de entrada (imagem original, autovetores e autovalores) e três saídas (imagem com correção de movimento e seus respectivos autovetores e autovalores ajustados), como mostra a figura 3.3.

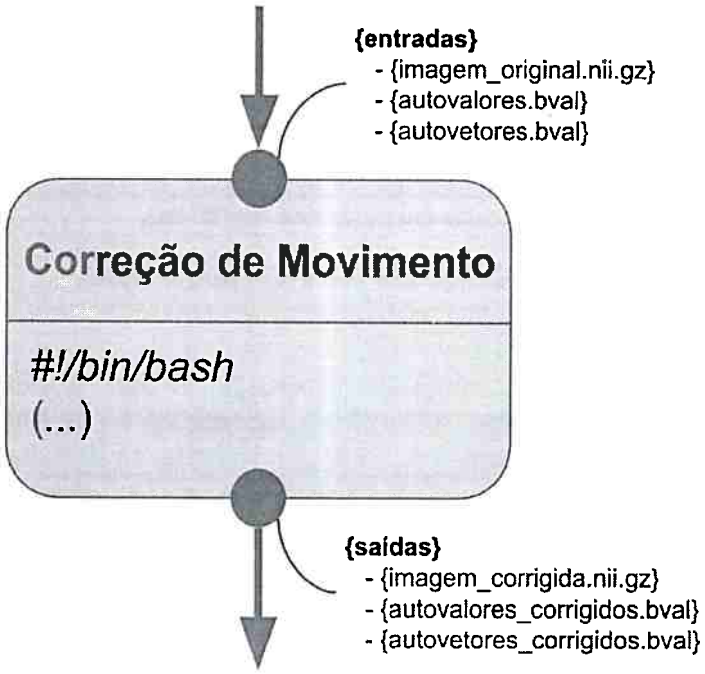

Figura 3.3: *Exemplo de definição de atividade atômica: os comandos necessários para realizar a correção de movimento estão encapsulados em uma única atividade, facilitando seu uso.*

Tabela de Requisitos Específicos

Com base no que apresentamos nesta seção, elaboramos a tabela 3.3 com os requisitos específicos para nosso sistema de gerenciamento de workflows. Note que nossa solução também agrega algumas características de um portal científico, permitindo que especialistas de domínio (*e.g.* médicos e biomédicos) também o utilizem.

RE-2.1	Interpretar modelos abstratos e conceituais de workflows e transformá-los em modelos concretos.
RE-2.2	Reconhecer e manipular os tipos de dados utilizados em processamento de imagens médicas.
RE-2.3	Executar um workflow (ou parte dele) sobre uma coleção de dados (paralelismo automático).
RE-2.4	Permitir que atividades possam consumir múltiplas entradas de dados, reduzindo-as a uma única saída (sincronização automática).
RE-2.5	Permitir que uma única atividade encapsule múltiplas execuções de comandos (atividade atômica).
RE-2.6	Possuir uma biblioteca de atividades (ou componentes) para processamento de imagens médicas, que podem ser utilizadas em diversos workflows. Essa biblioteca deve ser extensível, ou seja, novos componentes podem ser implementados e adicionados à biblioteca.

Tabela 3.3: *Requisitos específicos para o WfMS.*

O requisito **RE-2.6** é especialmente importante para o grupo de pesquisa *eScience* do Instituto de Matemática e Estatística da Universidade de São Paulo. Novos algoritmos e ferramentas, frutos das pesquisas realizadas no instituto, poderão ser publicados no portal.

Na tabela 3.4 também adicionamos os requisitos não funcionais que realizam os objetivos de

escalabilidade, acessibilidade e transparência propostos na seção 1.2.

RNF-1	A plataforma proposta deve ser distribuída como software livre.
RNF-2	Expor uma interface gráfica *web* para composição de workflows e exploração de dados.
RNF-3	Executar atividades em infraestrutura computacional distribuída.
RNF-4	Integrar-se ao sistema de gerenciamento de imagens médicas para importação dos dados que serão utilizados como entradas nos workflows.

Tabela 3.4: *Requisitos não funcionais para o WfMS.*

Nossa solução contempla a análise, escolha e adaptaptação de um WfMS com base nestes requisitos e na comparação entre os sistemas de workflows apresentada no capítulo 2 (veja as tabelas 2.1, 2.2 e 2.3). No próximo capítulo, evidenciaremos a relevância da plataforma através da implementação de um exemplo típico de workflow de análise estatística entre diferentes grupos de indivíduos.

3.2.3 Execução de Workflows de Análise de Imagens Médicas na Nuvem

Em geral, análises de imagens médicas são computacionalmente intensivas, principalmente quando o objetivo é processar grandes quantidades de dados de uma só vez. Este é o caso do workflow que apresentamos na seção 2.4.2. Podemos lançar mão da computação distribuída visando: i) diminuir o tempo total de processamento dessas análises; e ii) otimizar o uso dos recursos computacionais disponíveis uma rede de computadores.

A execução de workflows na nuvem pode ser vista como uma especialização da execução de workflows em um ambiente de computação distribuída, tal qual o que apresentamos na seção 2.3.4. Ou seja, em nossa proposta, vamos adotar a abordagem *VM-internal* (Tröger e Merzky, 2014), pois além de tornar possível o processamento de workflows em nuvens computacionais, não há necessidade de se alterar o protocolo e a API do arcabouço de gerenciamento de nuvem utilizado. Na prática, visamos desacoplar o DRMS do provedor de IaaS, possibilitando a portabilidade entre diferentes provedores (OpenNebula, OpenStack, Amazon AWS, Rackspace, etc).

O método para garantir a elasticidade da plataforma (*i.e.* provisionamento de recursos computacionais sob demanda) é dependente do provedor de nuvem. No OpenNebula, por exemplo, pode-se redimensionar o uso de recursos computacionais através de sua interface gráfica ou interface de linha de comando, introduzindo ou removendo máquinas virtuais conforme necessário, conforme mostram as figuras 3.4 e 3.5. Neste caso, escolhe-se um *template* de máquina virtual, que contém informações como: quantidade de processadores e memória RAM, tamanho do disco rígido e **imagem** utilizada. Uma **imagem** armazena o conteúdo do disco de uma máquina virtual, e é usada pelo OpenNebula para criar novas instâncias de máquinas virtuais. Máquinas virtuais, portanto, podem ser vistas como cópias de uma dada **imagem**, executada em um ambiente virtualizado com as especificações contidas em um *template*.

Além do método manual de redimensionamento da quantidade de recursos computacionais, o OpenNebula também oferece redimensionamento automático (*autoscaling*) através de um componente chamado *OneFlow*. Com ele é possível desenhar a arquitetura de um sistema, especificando quais serviços o compõe, quais *templates* de máquina virtual serão utilizados, e quantas instâncias de máquinas virtuais serão criadas para cada um destes *templates*. Pode-se configurar políticas de redimensionamento baseadas em métricas (*e.g.* taxa de uso dos processadores e memória, quantidade de *bytes* recebidos pelas interfaces de rede, taxa de uso de disco) ou em agendamento (*e.g.* entre 10 e 17h, de segunda à sexta-feira, primeira semana do mês).

Apresentaremos a seguir uma listagem que contém um trecho das configurações que utilizamos em nossos experimentos. Note que:

- *Roles* são diferentes serviços que compõem um sistema.

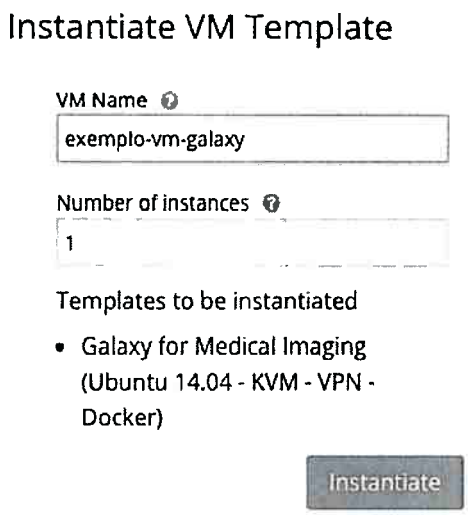

Figura 3.4: *Provisionamento de máquinas virtuais no OpenNebula: escolhe-se o nome das máquinas virtuais, o número de instâncias e o template utilizado.*

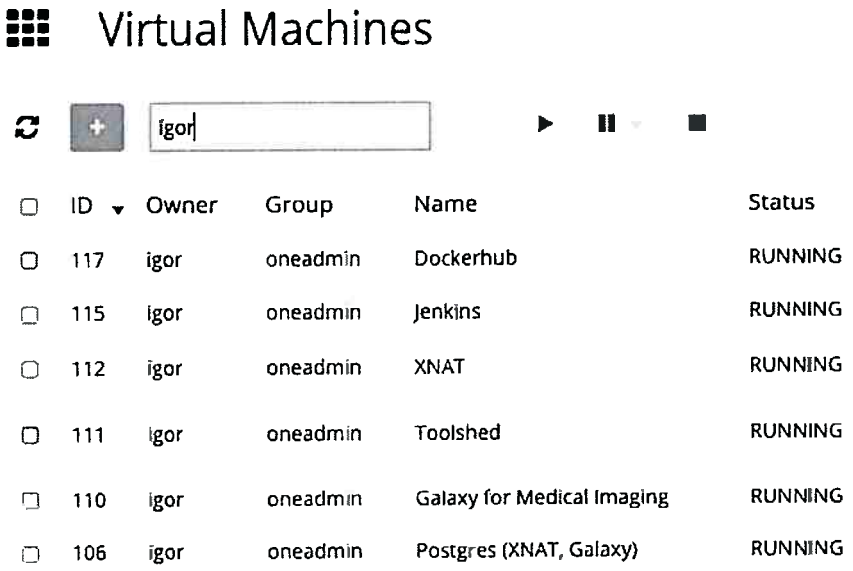

Figura 3.5: *Listagem de máquinas virtuais no OpenNebula: pode-se desligar, reiniciar ou iniciar instâncias.*

- A listagem a seguir contém três *roles*: i) `DrmsCentralManager`, que coordena o escalonamento de atividades e a comunicação entre os nós de submissão e os nós de execução do DRMS; ii) `DrmsGalaxyJobSubmitter`, que representa os nós de submissão do DRMS; e iii) `DrmsWorkerNode`, que representa os nós de execução do coordenados pelo gerenciador do DRMS. Em ambos `DrmsGalaxyJobSubmitter` e `DrmsWorkerNode`, cada uma das máquinas virtuais se registra automaticamente em `DrmsCentralManager`. Desta forma pode-se aumentar ou diminuir a quantidade de recursos distribuídos disponíveis no DRMS.

- Para `DrmsWorkerNode` usamos políticas de redimensionamento baseada na métrica `USEDCPU`, que indica o percentual de uso de CPU nas máquinas virtuais. Acima de 80%, uma nova máquina virtual é criada. Abaixo de 50% uma máquina virtual é removida.

```
1 {
2    "name": "DRMS",
3    "roles": [
4       {
5          "name": "DrmsCentralManager",
6          "cardinality": 1, // Quantidade fixa de m quinas virtuais para este
               servi o.
7          "vm_template": 21 // O template cont m a imagem pr -configurada do
               gerenciador do DRMS.
8       },
9       {
10         "name": "DrmsGalaxyJobSubmitter",
11
12         "cardinality": 1,
13         "vm_template": 32, // O template cont m a imagem pr -configurada dos
               n s de submiss o.
14         "parents": [
15            "DrmsCentralManager" // O provisionamento deste servi o depende do
                  gerenciador de DRMS.
16         ]
17      },
18      {
19         "name": "DrmsWorkerNode",
20
21         "cardinality": 3, //    princ pio, s o provisionadas tr s m quina
               virtuais.
22         "vm_template": 23, // O template j  contem a imagem pr -configurada dos
               n s de submiss o.
23         "parents": [
24            "DrmsCentralManager"
25         ],
26         "min_vms": 3, // Quantidade m nima de n s de execu  o
27         "max_vms": 10, // Quantidade m xima de n s de execu  o
28
29         "elasticity_policies": [
30            {
31               "type": "CARDINALITY",
32               "adjust": 1,
33               "expression": "USEDCPU > 80",
34            },
35            {
36               "type": "CARDINALITY",
37               "adjust": -1,
38               "expression": "USEDCPU < 50",
39            }
40         ]
41      }
42   ],
43 }
```

Na Amazon AWS, os conceitos de elasticidade são similares aos encontrados no OpenNebula.

Pode-se provisionar novas máquinas virtuais através do console do *Elastic Cloud Computing* (EC2), como mostra a figura 3.6, e gerenciar as instâncias em execução, assim como na figura 3.7.

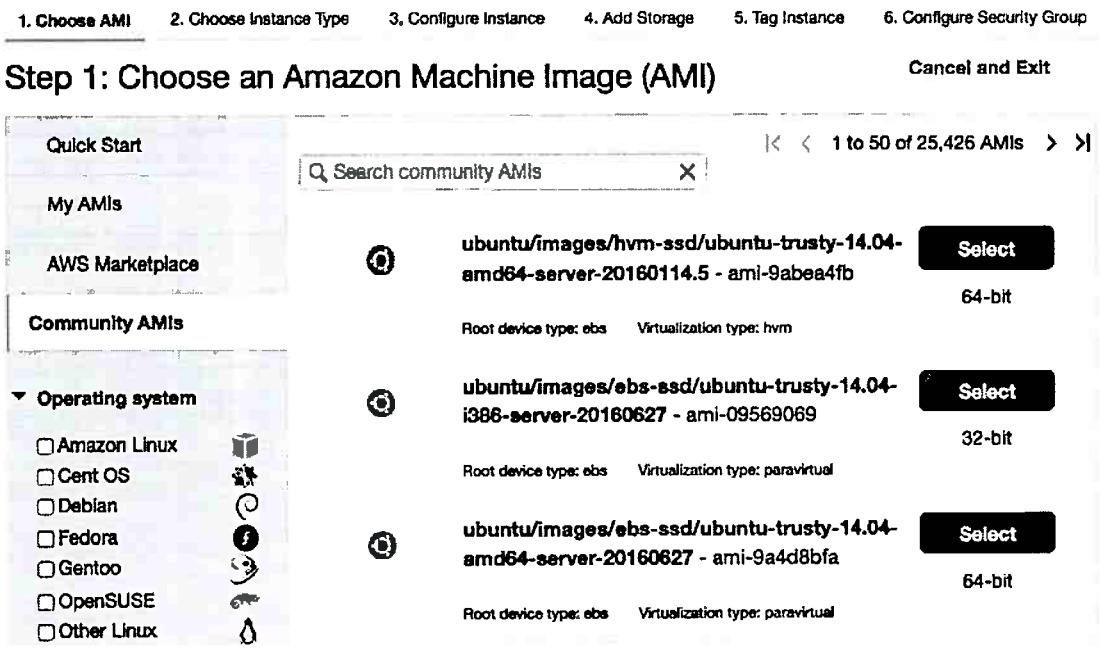

Figura 3.6: *Provisionamento de máquinas virtuais na AWS: escolhe-se a imagem, o tipo de instância (e.g. número de processadores, quantidade de memória RAM, velocidade da rede de dados); a quantidade de instâncias; o tipo e capacidade do dispositivo de armazenamento e finaliza-se com as configurações de rede e segurança das novas instâncias.*

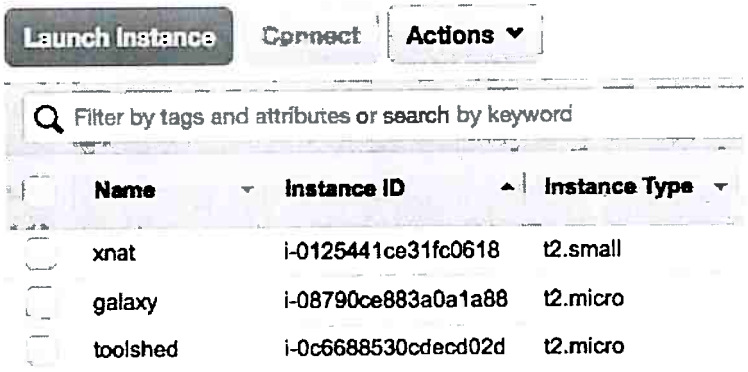

Figura 3.7: *Listagem de máquinas virtuais no console do Elastic Cloud Computing da AWS: pode-se desligar, reiniciar, configurar, copiar e iniciar instâncias.*

A AWS também dispõe de um mecanismo de redimensionamento automático similar ao do OpenNebula. Para usar este mecanismo é necessário criar uma *launch configuration*, que determina a imagem (pública ou personalizada), e o perfil de máquina virtual que se deseja (*e.g.* tipo de instância, tipo e capacidade do dispositivo de armazenamento, configurações de rede e de segurança). Em nossos experimentos utilizamos imagens personalizadas que foram pré-configuradas para executar os sistemas desejados, como o WfMS e o DRMS. A partir da *launch configuration*, cria-se o grupo de *autoscaling*, que contém as políticas de aumento e redução da quantidade de máquinas virtuais em execução. A figura 3.8 mostra a configuração que criamos para os nós de execução do nosso DRMS.

Em resumo, nosso trabalho objetiva a criação de uma plataforma única que possa ser compartilhada por diversos usuários, capaz de aumentar ou diminuir a quantidade de recursos computacionais de acordo com a demanda. Demonstraremos a validade de nossa proposta através da execução

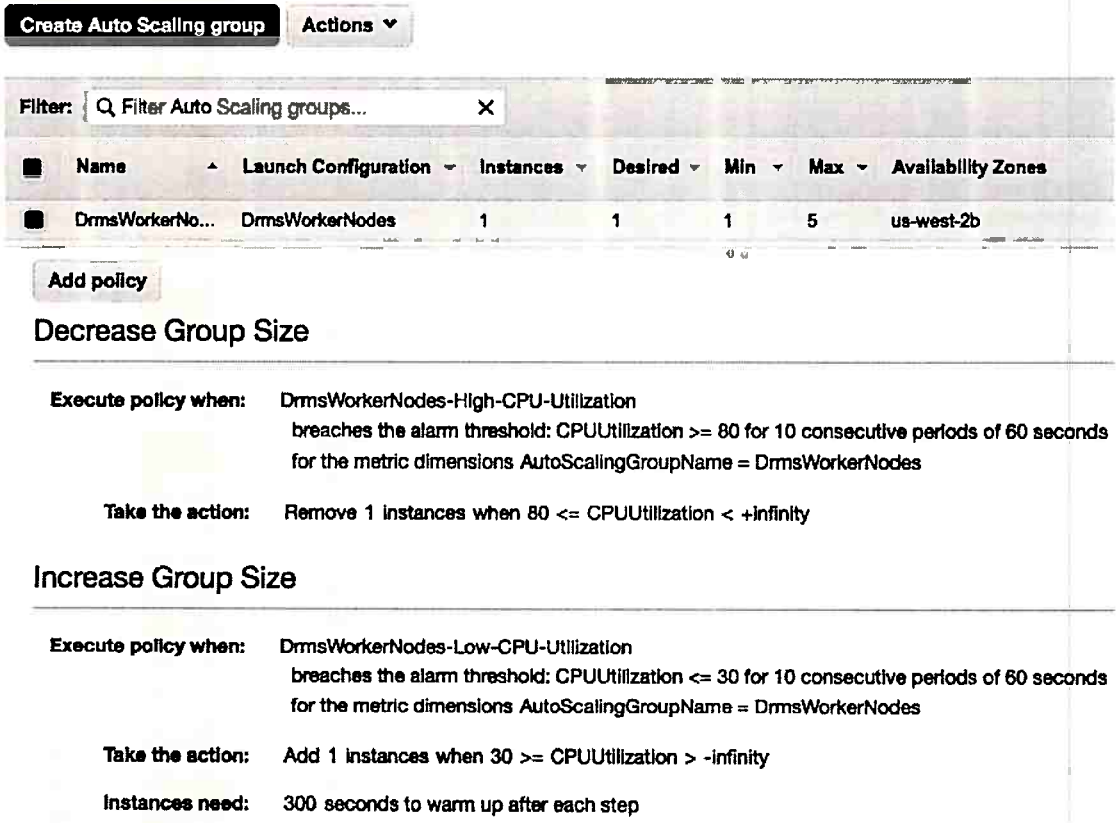

Figura 3.8: *Grupo de autoscaling para os nós de execução do DRMS. A parte inferior da figura mostra as políticas usadas para aumentar ou diminuir a quantidade de máquinas virtuais em execução.*

de exemplos de workflows reais, coletando os tempos de processamento do workflow na nuvem da Amazon AWS em contraste com o processamento do mesmo workflow em um único computador.

Capítulo 4

Resultados

Neste capítulo descreveremos os resultados que encontramos com a metodologia descrita no capítulo 3. Em resumo, nossos principais resultados foram:

- uma prova de conceito de serviço para recepção e organização de imagens médicas;
- uma prova de conceito de WfMS para análise de imagens médicas, utilizando como base os workflows encontrados em *Sex differences in cortical volume and gyrification in autism* (Schaer *et al.*, 2015);
- configuração e instalação do WfMS na nuvem da Amazon AWS;
- coleta e análise da execução dos workflows na nuvem.

4.1 Estratégias para Recepção e Organização de Imagens Médicas

Com base no que apresentamos na seção 3.2.1, consideramos duas estratégias:

(i) agregar ao WfMS as funcionalidades que atendam aos requisitos específicos de **RG-1** (arquitetura monolítica); e

(ii) implementar um serviço que contemple estes requisitos, mas que não interfira diretamente no WfMS (arquitetura por micro-serviços).

Neste trabalho, optamos pela opção (ii), pois os escopos de um WfMS e de um sistema de gerenciamento de imagens médicas são muito diferentes entre si: um WfMS refere-se a modelagem e execução de workflows, enquanto que um sistema de gerenciamento de imagens médicas refere-se à organização e distribuição de dados. Há diversas vantagens na segregação de um sistema em diversos serviços: melhor segregação do domínio de problema, bases de código menores, maior desacoplamento entre os ciclos de vida de cada serviço, dentre outras (Newman, 2015). A figura 4.1 mostra a diferença entre as duas estratégias que consideramos.

A partir desta definição, buscamos por projetos de código livre que pudéssemos utilizar em nossa solução. O XNAT, projeto da *Washington University School of Medicine* (Marcus *et al.*, 2007), é um sistema de gerenciamento de imagens e dados de experimentos que contempla diretamente os requisitos RE-1.1 à RE-1.5. Além disso, ele dispõe de recursos para armazenar imagens em diferentes formatos (*e.g.* DICOM, NIfTI) e de diferentes modalidades de imagens (*e.g.* MRI, CT, PET-CT).

4.1.1 XNAT: Experimentos na nuvem

A arquitetura do XNAT é complexa e sua curva de aprendizado para um desenvolvedor é grande, tornando a criação de extensões ou personalizações muito trabalhosa. Entretanto, mesmo sem personalizações, o XNAT já possui as funcionalidades básicas necessárias para a aquisição,

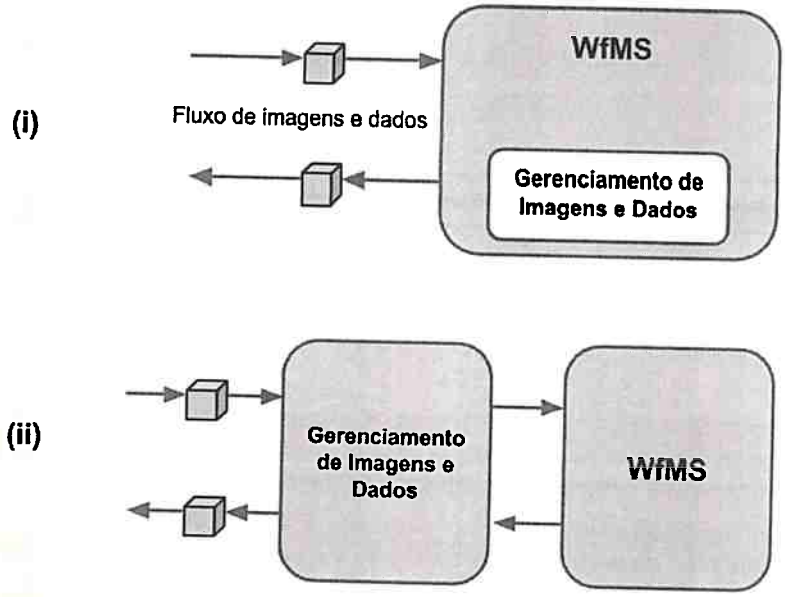

Figura 4.1: *Diagrama representando as duas estratégias para gerenciamento de imagens médicas e metadados: (i) as funcionalidades de recepção, armazenamento, gerenciamento e distribuição de imagens médicas são incorporadas ao WfMS; (ii) todas estas funcionalidades são segregadas em um outro sistema, mantendo o WfMS intacto.*

armazenamento e gerenciamento de imagens médicas de diversas modalidades. Outra vantagem é a rica API disponível para controlar e gerenciar programaticamente os dados e imagens nele contidos.

O XNAT pode ser executado em ambientes virtualizados, podendo escalonar vertical e horizontalmente, como descrito em Architecture . Em nossos experimentos na Amazon AWS, utilizamos uma única instância do EC2 para executar o XNAT. As imagens são armazenados em um volume do *Elastic Block Storage* (EBS), que é capaz de aumentar ou diminuir a velocidade de leitura ou escrita em disco sob demanda. O banco de dados do XNAT, responsável por armazenar informações sobre usuários, projetos, pacientes e aquisições de imagens, foi configurado em uma instância do *Relational Database Service* (RDS), que também permite escalonamento vertical ou horizontal.

Além de configurar o XNAT na Amazon AWS, também realizamos testes no *data center* de CHU, dentro de uma nuvem privada que criamos para este propósito. Efetuamos os seguintes testes:

- envio de imagens adquiridas por equipamentos de MRI;
- envio de imagens via interface *web*;
- visualização de sessões de MRI através da interface web; e
- acesso remoto às imagens armazenadas através do visualizador de imagens médicas *3DSlicer* (Fedorov *et al.*, 2012).

A figura 4.2 mostra uma sessão de aquisição de imagem de ressonância magnética dentro do XNAT. O acesso remoto às sessões de MRI armazenadas no XNAT é mostrado na figura 4.3.

Contudo, não desejamos acoplar o WfMS diretamente ao XNAT, pois isso dificultaria futuras integrações com outros sistemas de gerenciamentos de imagens médicas. Também não desejamos alterar o XNAT diretamente, pois: (i) criaríamos um forte acoplamento entre o provedor de serviço (XNAT) e o WfMS; e (ii) a customização do XNAT é extremamente trabalhosa, devido a sua complexa arquitetura. Portanto, criamos o **Hubble**, uma aplicação *adaptadora*, capaz de interagir com mais de um gerenciador de imagens médicas, simplificando e abstraindo a integração entre o WfMS e o sistema de gerenciamento de imagens e dados de experimentos, como mostra a figura 4.4. Essa estratégia também permitirá futuras integrações com bases públicas de imagens médicas, por

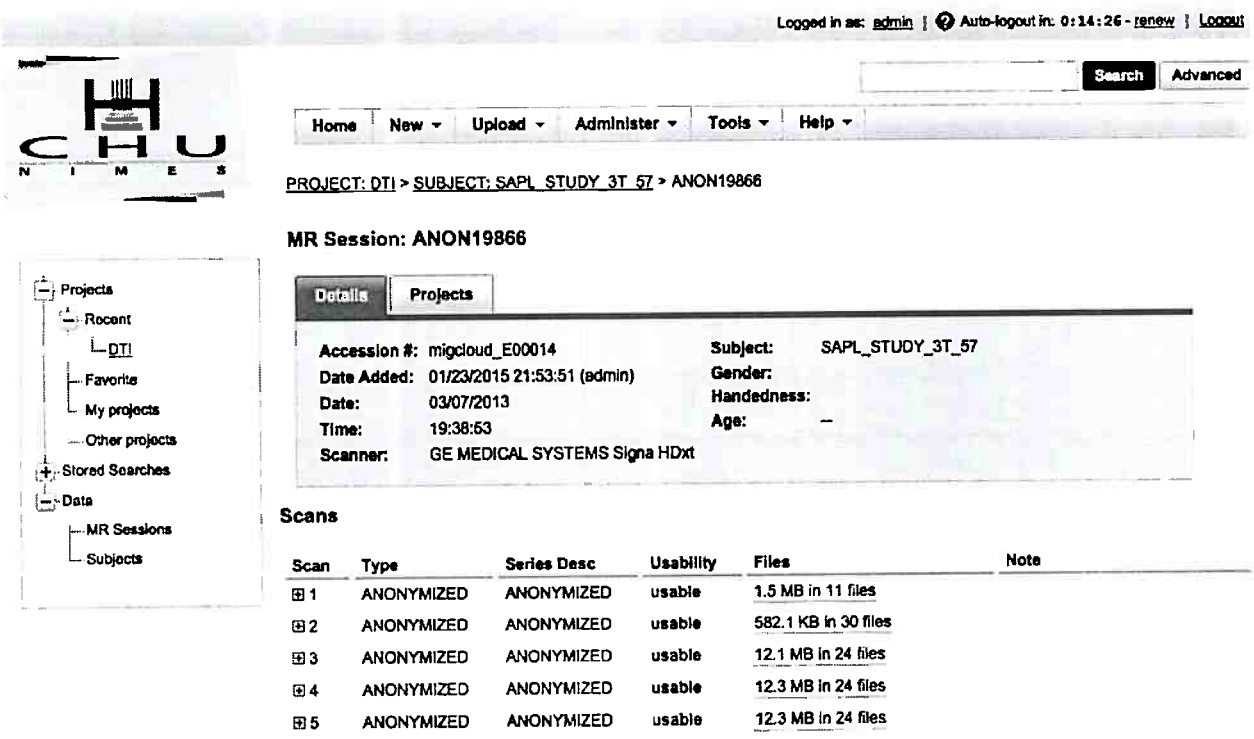

Figura 4.2: *Exemplo de sessão de MRI armazenada no XNAT.*

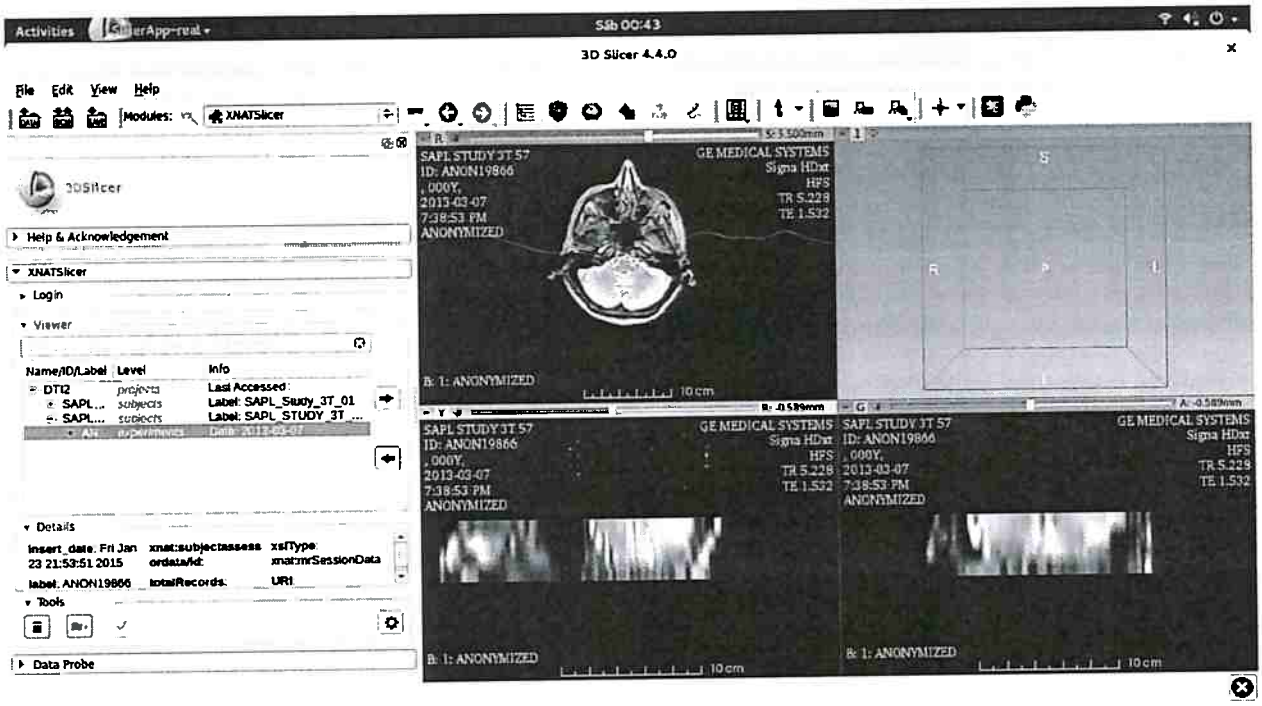

Figura 4.3: *Acesso às imagens armazenadas no XNAT via* 3DSlicer.

exemplo o OASIS (*Open Access Series of Imaging Studies*)[1], publicado pela *Washington University School of Medicine*, ou o *brain-development.org*, mantido pelo *Imperial College London*.

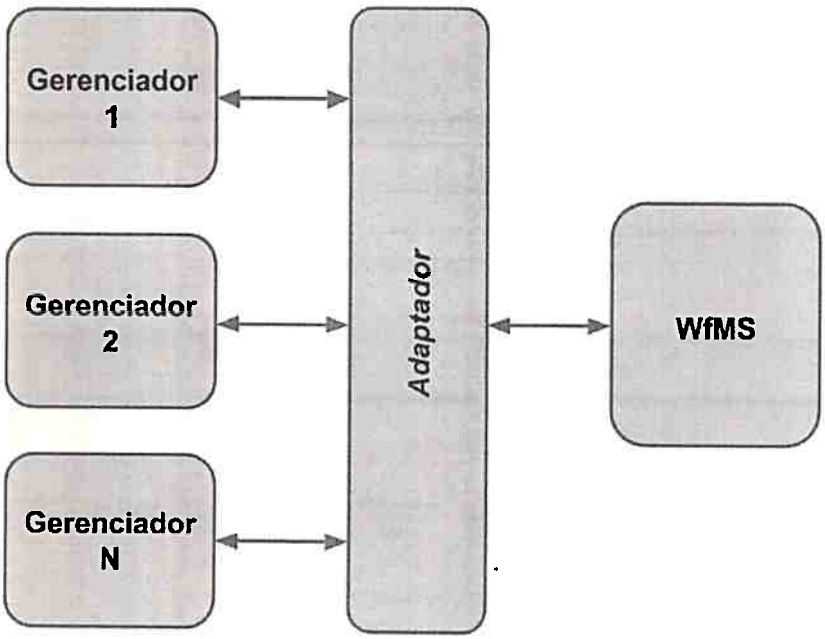

Figura 4.4: *Aplicação* adaptadora, *que abstrai e isola as especificidades de cada sistema de gerenciamento de imagens médicas.*

4.1.2 Hubble: Integrando um WfMS ao XNAT

O projeto Hubble é um dos produtos deste trabalho, desenvolvido em Java e distribuído como software livre. Em nossa pesquisa, usamos o Hubble para integrar o XNAT ao WfMS. O caso de uso mais comum que orientou a implementação do Hubble foi:

(1) Usuário do WfMS requisita dados para seu experimento.

(2) WfMS direciona usuário para o Hubble.

(3) Hubble apresenta os projetos, experimentos, indivíduos e sessões do XNAT aos quais o usuário tem acesso.

(4) Usuário seleciona os indivíduos para seu experimento.

(5) Hubble transmite ao WfMS as sessões dos indivíduos selecionados.

(6) Usuário é redirecionado ao WfMS.

Em nossos experimentos, utilizamos o Galaxy como WfMS. A comunicação entre os sistemas é realizada através das APIs RESTful (*Representational State Transfer*) presentes tanto no Galaxy quanto no XNAT. O Hubble armazena as chaves de acesso de seus usuários e as usa para realizar as operações de leitura e escrita . A figura 4.5 ilustra o diagrama de sequência para o principal caso de uso do Hubble.

Os usuários podem selecionar arquivos no formato DICOM (*Digital Imaging and Communications in Medicine*) ou NIfTI (*Neuroimaging Informatics Technology Initiative*). Uma vez selecionados, os arquivos podem ser enviados para qualquer um dos experimentos do Galaxy pertencentes

[1] http://www.oasis-brains.org

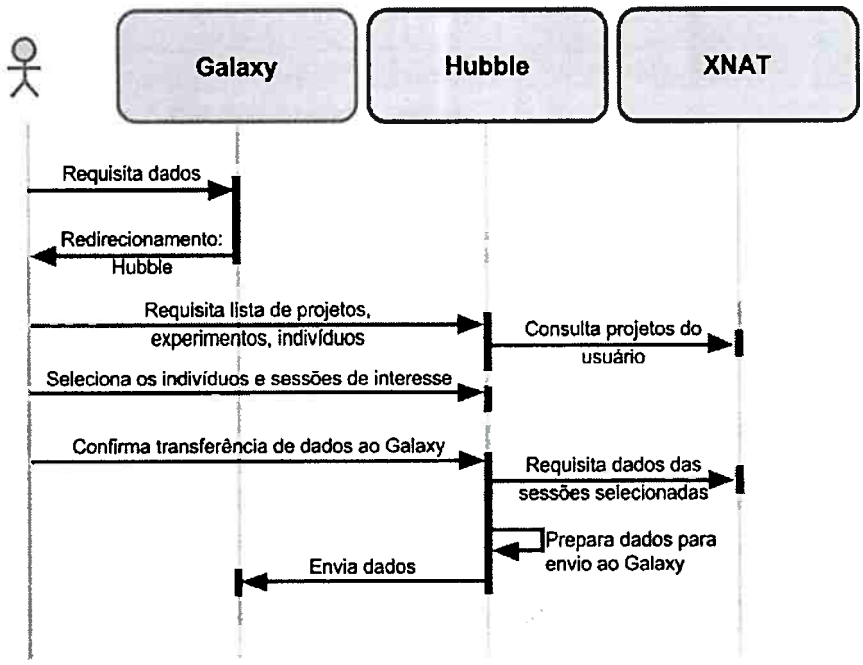

Figura 4.5: *Diagrama de sequência para o principal caso de uso do Hubble.*

ao usuário, como mostra a figura 4.6. Cada arquivo recebido pelo Galaxy é processado e transformado em um *dataset*, estrutura interna que representa um ou mais arquivos acompanhado de sua proveniência (*e.g.* origem, a qual usuário pertence, data de criação). Ao final da transferência, o Hubble cria uma coleção de *datasets* no Galaxy e inclui todos os arquivos transferidos dentro desta coleção. Essa estratégia possui duas vantagens com relação à usabilidade:

- os arquivos selecionados pelo usuário permanecem juntos dentro de uma mesma estrutura, deixando a interface gráfica do Galaxy menos confusa para o usuário;
- as atividades de um workflow podem ser realizadas diretamente sobre a coleção de *datasets*, evitando que o usuário tenha que selecionar cada *dataset* individualmente.

A figura 4.7 mostra como os arquivos ficam dispostos no experimento do usuário no Galaxy.

4.2 Um Sistema de Gerenciamento de Workflows para Imagens Médicas

Esta seção descreve os experimentos e argumentos que alicerçaram nossa proposta de WfMS para análise de imagens médicas. Parte das conclusões destes experimentos foram utilizadas para a elaboração dos requisitos descritos na seção 3.2.2.

Como estudo de caso escolhemos dois workflows de análise de grupos populacionais através de imagens de MRI estrutural, pois:

(i) esse tipo de análise é complexo e envolve diversas etapas, geralmente executadas manualmente pelo cientista;

(ii) é grande a quantidade de parâmetros utilizados em cada ferramenta, entretanto raras publicações citam quais foram usados para obter seus resultados, dificultando a reprodutibilidade; e

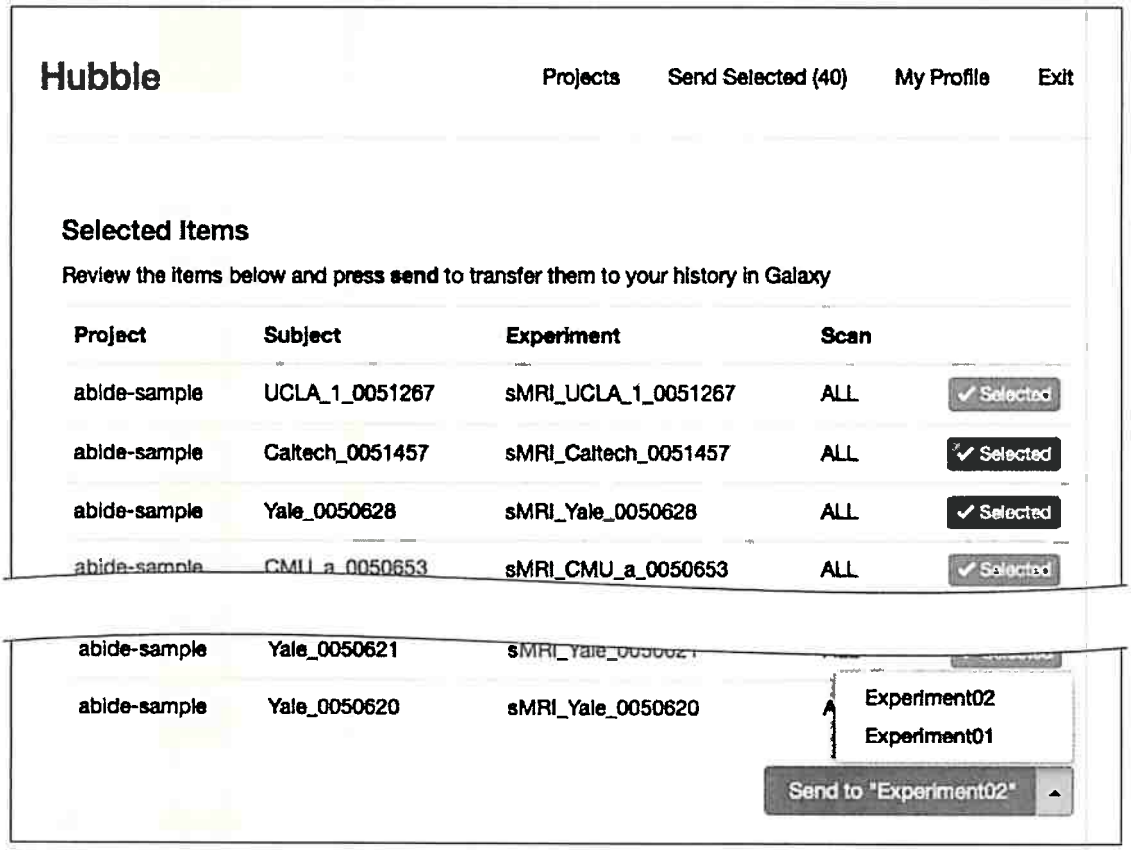

Figura 4.6: *Seleção e envio de dados ao Galaxy através do Hubble.*

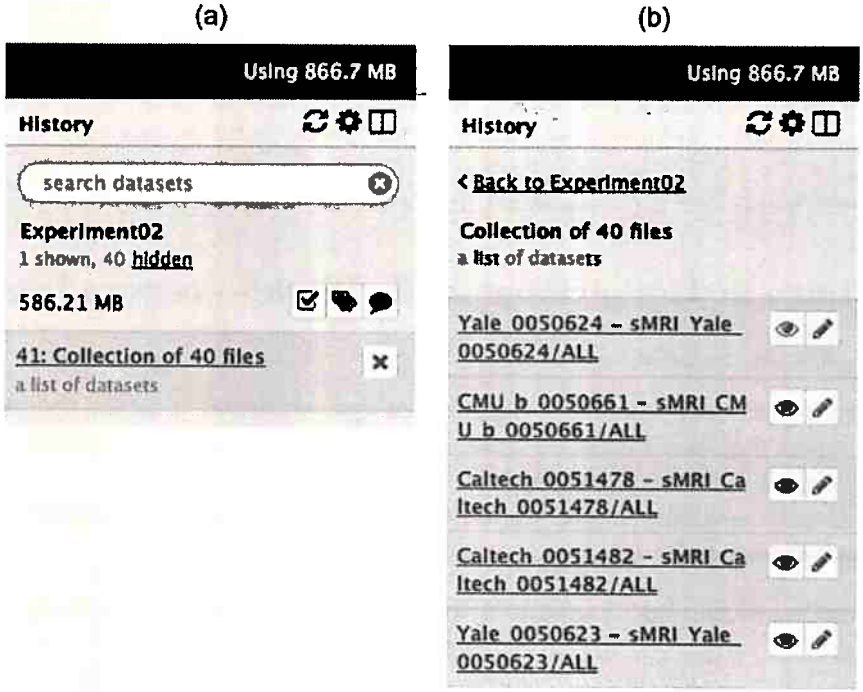

Figura 4.7: *Histórico de operações em um experimento no Galaxy: (a) coleção de arquivos (datasets) após a transferência de arquivos realizada pelo Hubble; (b) arquivos pertencentes à coleção.*

(iii) a paralelização da execução de workflows pode ajudar a reduzir o tempo de execução desses experimentos, que geralmente é grande (podendo facilmente chegar à ordem de semanas, dependendo do experimento e quantidade de indivíduos incluídos na análise).

A seguir, apresentaremos o workflow que escolhemos para investigar. Na seção 4.2.2 apresentaremos nossa escolha de WfMS e os argumentos utilizados nesta decisão. Depois mostraremos a abordagem que utilizaremos para abstrair a complexidade das ferramentas computacionais na seção 4.2.3. Por fim, na seção 4.2.4 apresentaremos a modelagem do workflow escolhido e os resultado que obtivemos.

4.2.1 Estudo de Caso: Aplicações Práticas da Análise de Imagens de MRI

Conforme abordamos em na seção 3.1 do capítulo anterior, em *Sex differences in cortical volume and gyrification in autism*, Schaer *et al.* (2015) investiga se existem diferenças estatísticas significativas no volume ou espessura cortical entre os grupos: homens com e sem Transtorno do Espectro Autista (TEA), mulheres com e sem TEA.

Para construir e validar nossa solução, utilizamos um workflow similar ao apresentado na figura 2.20, cujo objetivo é identificar variações significativas no volume cortical em uma população de indivíduos, mediante análise de imagens de MRI. Aplicações práticas deste workflow podem ser encontradas nos trabalhos de Schaer *et al.* (2015), Kucharsky Hiess *et al.* (2015) e (Lefebvre *et al.*, 2015). O fluxo de atividades está representado na figura 4.8.

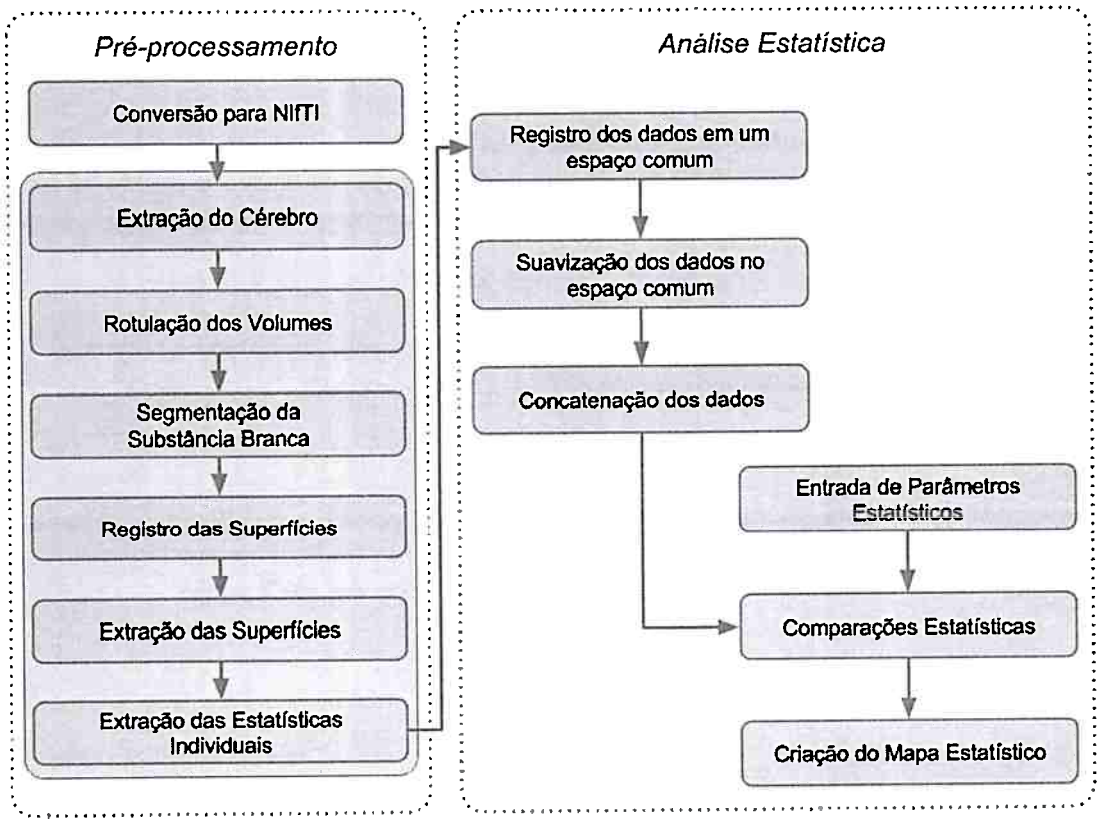

Figura 4.8: *Representação gráfica de um workflow para análise estatística de um grupo de indivíduos a partir de suas imagens do cérebro. O objetivo do pré-processamento é obter a reconstrução cortical dos volumes e superfícies, além de extrair dados úteis para diversos tipos de análises (e.g. rotulação, espessura, curvatura).*

Durante a execução deste workflow, as atividades que compõe o pré-processamento são executadas para cada paciente do experimento. Em seguida, ocorre a redução das linhas de execução paralelas, que alimentam a fase de análise estatística. A figura 4.9 ilustra as linhas de execução

convergindo para uma única atividade. Alguns WfMS são capazes de gerar este tipo de workflow a partir de uma representação mais abstrata, como na figura 4.8.

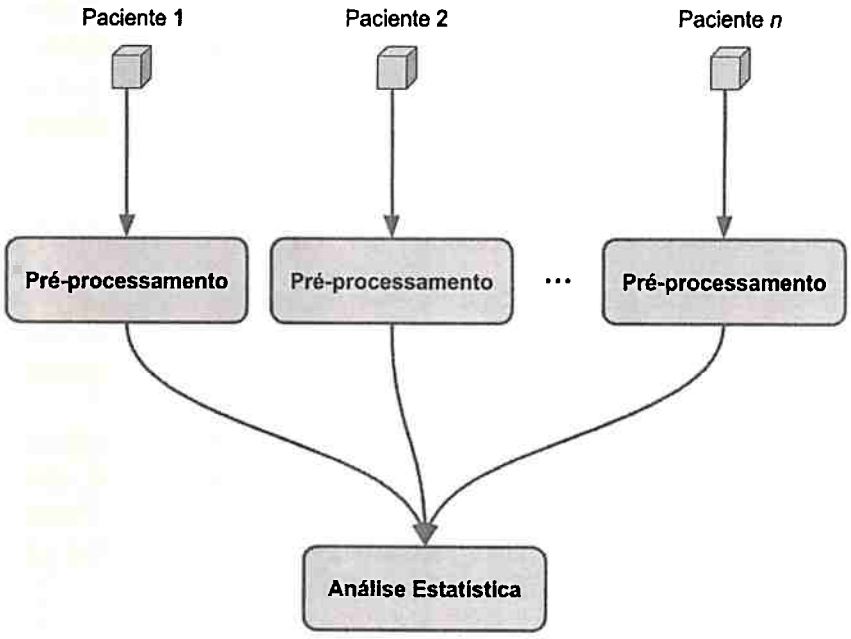

Figura 4.9: *Representação das instâncias de atividades no workflow para análise estatística de imagens de MRI.*

Este workflow pode ser executado manualmente pelo próprio pesquisador em seu computador, por intermédio de pequenos *scripts* ou linhas de comando. A primeira etapa converte as imagens adquiridas em formato DICOM para NIfTI, pois muitas ferramentas computacionais trabalham somente com este formato:

```
1 cd $PACIENTE
2 for SERIE in * ; do
3   cd $SERIE
4   IMAGEM=$( ls | head -1 )
5   dcm2nii -a n $IMAGEM
6   cd ..
7 done;
```

Para as etapas de pré-processamento, diversas ferramentas podem ser utilizadas, como o FSL, AFNI ou FreeSurfer. No nosso caso, utilizamos o FreeSurfer para realizar nossos experimentos. O trecho de código abaixo mostra como essas etapas são executadas em *shell script*.

```
# Define diret rio onde os arquivos originais NIfTI residem
RAW_IMAGES=/data/ABIDE/nii

# Define diret rio para onde os arquivos pr -processados ser o copiados
SUBJECTS_DIR=/Applications/freesurfer/subjects

# Pr -processa todos os pacientes no diret rio
for IMAGE in * ; do
    FILE_NAME=$(basename "$IMAGE")
    PATIENT_ID="${filename%.nii.gz}"

    # Extra  o do c rebro
    recon-all -i $IMAGE -s $PATIENT_ID -autorecon1 -notal-check -noappend -no-
        isrunning

    # NOTA: pode-se realizar o controle de qualidade manual do resultado

    # Rotula  o e segmenta  o da subst ncia branca
    recon-all -s $PATIENT_ID -autorecon2 -no-isrunning

    # Extra  o e registro das superf cies, al m das estat sticas individuais
    recon-all -s $PATIENT_ID -autorecon3 -no-isrunning
done
```

Ao final desta etapa, tem-se a reconstrução cortical dos volumes e superfícies, além da rotulação e métricas úteis para os próximos passos da análise, como o volume, espessura e curvatura dos tecidos. A partir desses resultados, pode-se iniciar a análise estatística do grupo de pacientes, conforme mostra o trecho de código a seguir.

```
cd $SUBJECTS_DIR

# Registro dos dados de cada indiv duo em um espa o comum (fsaverage)
# Suaviza  o dos dados em 0, 5, 10, 15, 20 e 25mm FWHM (tamanho do kernel do
    filtro Gaussiano empregado, em largura    meia altura em mil metros)
for PATIENT in * ; do
    recon-all -s $PATIENT -qcache
done

# Arquivo que descreve o grupo de indiv duos considerados neste workflow (
    identificador  nico , idade, sexo, diagn stico)
GROUP_DESCRIPTOR_FILE=/data/ABIDE/gender_age_diagnostics.fsgd
ANALYSIS_NAME="GenderAgeDiagnosis_Volume_LeftHemisphere"

# Arquivo contendo a matriz de contraste a ser usada no modelo linear geral
CONTRAST_MATRIX=/data/ABIDE/effects_of_gender_and_diagnosis_covariate_age.mtx

# Concatena os dados de todos os indiv duos presentes no arquivo
    GROUP_DESCRIPTOR_FILE
mris_preproc --fsgd $GROUP_DESCRIPTOR_FILE --cache-in volume.fwhm10.fsaverage --
    target fsaverage  --hemi lh --out $ANALYSIS_NAME.mgh

# Cria modelo linear geral para o grupo de indiv duos
mri_glmfit --y $ANALYSIS_NAME.mgh --fsgd $GROUP_DESCRIPTOR_FILE dods --C
    $CONTRAST_MATRIX --surf fsaverage lh --cortex --glmdir $ANALYSIS_NAME
```

A partir daqui, temos o modelo linear geral para os parâmetros estatísticos dados. O modelo é um arquivo que armazena um mapa de significância $-log(p)$ para cada região do cérebro. Podemos sobrepor este mapa em uma superfície qualquer do cérebro que tenha sido reconstruida pelo FreeSurfer durante o pré-processamento (*e.g.* substância branca, pia-máter). Para saber quais regiões do cérebro apresentam diferenças estatísticas significativas, podemos filtrar os dados onde $p < 0.001$, por exemplo. A figura 4.11 mostra este resultado para $p < 0.05$, extraído a partir do programa *freeview*, parte integrante do FreeSurfer.

As ferramentas apresentadas acima são complexas e possuem vários pré-requisitos para sua

Figura 4.10: *Regiões com $p < 0.05$ para a comparação entre os quatro grupos estudados com relação ao volume cortical.*

utilização:

- conhecimento prévio de seus parâmetros, funcionamento e estrutura de diretórios utilizada;
- conhecimento de programação em *shell script*; e
- conhecimento de como instalá-las.

Note que estes pré-requisitos diminuem consideravelmente a acessibilidade e reprodutibilidade de experimentos alicerçados nestas ferramentas. Um dos principais objetivos deste trabalho é eliminar ou atenuar estes pré-requisitos.

4.2.2 Escolha do WfMS para Exploração de Imagens Médicas

Com base nos requisitos apresentados na seção 3.2.2 e na comparação entre os sistemas de workflows apresentada no capítulo 2 (veja as tabelas 2.1, 2.2 e 2.3), elegemos o **Galaxy** como o arcabouço para a plataforma de exploração de imagens médicas aqui proposta.

Implementação de Novos Componentes do Galaxy

De modo geral, o sistema de workflows Galaxy contempla grande parte dos requisitos que definimos na seção 3.2.2 do capítulo anterior. A princípio, realizamos algumas modificações no código para que pudéssemos usá-lo para workflows de imagens médicas, porém em 2016 foram lançadas diversas melhorias com relação à extensibilidade do Galaxy através de componentes personalizados. Essas modificações são referentes aos requisitos listados a seguir:

(i) **RE-2.2**: implementar um novo tipo de dados no Galaxy que represente uma série de aquisição de imagem médica. No Galaxy, isso é possível através da definição de um novo `datatype`. Para nossa plataforma, implementamos os seguintes novos `datatypes`:

   (a) `Dicom`, um tipo de dado binário capaz de armazenar arquivos no formato DICOM, bem como seus metadados (e.g. paciente, modalidade, equipamento médico);

   (b) `Nifti`, representando um arquivo no formato NIfTI, capaz de também armazenar metadados;

(c) `FreeSurferSubject`, representando a estrutura de diretórios e arquivos resultantes da reconstrução cortical de um indivíduo;

(ii) **RE-2.6**: implementar os componentes necessários para que o workflow de análise de imagens possa ser modelado. Neste caso, utilizamos o arcabouço de componentes do Galaxy, conhecido como **Toolshed**. Aprofundaremos este requisito na próxima seção.

(iii) **RNF-3**: implementar e configurar a infraestrutura computacional sob a qual as instâncias de workflow serão executadas. Este tópico será abordado na seção 4.3.

(iv) **RNF-4**: implementar um componente para importação de `datasets` presentes no sistema de gerenciamento de imagens médicas (*e.g.* XNAT).

Todas as customizações, componentes e serviços resultantes deste trabalho estão disponíveis em https://github.com/igortopcin/galaxy e https://bitbucket.org/migusp.

4.2.3 Abstração das Ferramentas Computacionais

Todo WfMS possui um arcabouço para desenvolvimento de componentes, porém alguns são mais flexíveis que outros. Através desses arcabouços é possível definir uma especificação de componente, que poderá ser utilizado pelos usuários em seus workflows. Essa especificação tipicamente descreve: i) como os programas ou comandos devem ser executados; ii) quais parâmetros devem ser passados para cada programa; e iii) os produtos finais da execução do programa.

No Galaxy, novos componentes são descritos através de um XML de configuração, onde são especificados: i) a linha de comando para execução; ii) as portas de entrada e saída de dados com seus respectivos `datatypes`; e iii) opções relacionadas ao programa executado e texto de ajuda para os usuários. As principais estratégias para definir a linha de comando para a execução são: i) encapsular os comandos em um único executável (*e.g.* em Python, Java, shell); ou ii) escrever todo o código em shell, usando os recursos do processador de modelos *Cheetah*, conhecido na comunidade de usuários de Python por sua simplicidade. A listagem abaixo mostra o XML de configuração do componente de reconstrução cortical chamado *recon-all*. O resultado deste XML é a componente da figura.

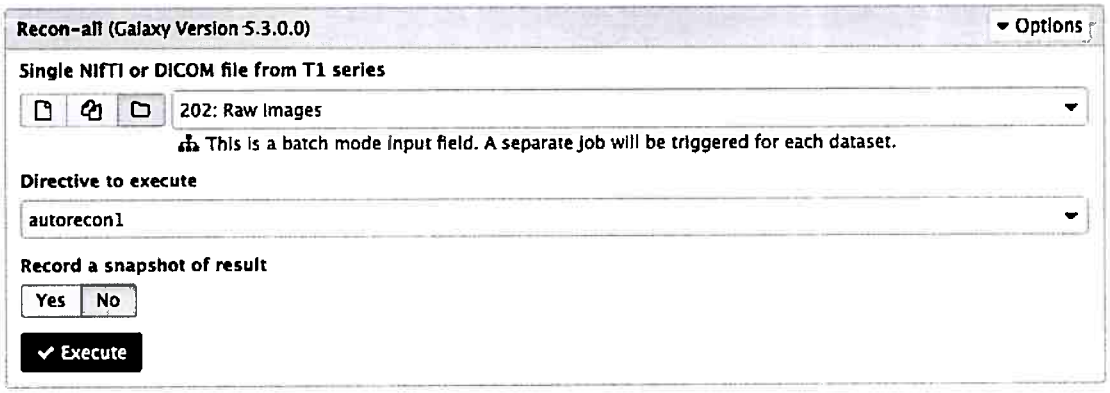

Figura 4.11: *Componente recon-all dentro do Galaxy. Os campos e opções exibidos são declaradas no XML de configuração do componente.*

```
1 <tool id="recon-all" name="Recon-all" version="5.3">
2     <description>perform cortical reconstruction in one or more inputs</description
          >
3     <command>
4 <![CDATA[
5         #from os.path import join, exists, dirname
6
7         #set $subjects_dir = join(dirname(str($outfile)), 'freesurfer')
```

```
8           mkdir -p $subjects_dir &&
9
10          #set $subject_id = str($input_image.metadata.PatientID)
11          #if $subject_id == 'None':
12              #set $subject_id = $input_image.name.split()[0]
13          #end if
14          #set $subject_id = $subject_id.replace(' ', '_')
15
16          #set $input_image_symlink = str($input_image) + '.' + $input_image.ext
17          #if not exists($input_image_symlink):
18              ln -s $input_image $input_image_symlink &&
19          #end if
20
21          #set $subject_files_dir = join($subjects_dir, $subject_id)
22
23          recon-all
24              $directive
25              #if not exists($subject_files_dir)
26                  -i $input_image_symlink
27              #end if
28              -sd $subjects_dir
29              -subjid $subject_id
30              -no-isrunning
31
32          #set $subject_files_dir = join($subjects_dir, $subject_id)
33          #if str($snapshot) == 'true':
34              && mkdir -p $outfile.extra_files_path
35              && cp -R $subject_files_dir $outfile.extra_files_path/$subject_id
36          #end if
37  ]]>
38      </command>
39      <inputs>
40          <param name="input_image" format="nii.gz,dcm.zip" type="data" label="$single
                NIfTI or DICOM file from T1 series" />
41          <param name="directive" type="select" label="Directive to execute">
42              <option value="-all">all</option>
43              <option value="-autorecon1">autorecon1</option>
44              <option value="-autorecon2">autorecon2</option>
45              <option value="-autorecon3">autorecon3</option>
46              (...)
47          </param>
48          <param name="snapshot" type="boolean" label="Record a snapshot of result"
                truevalue="true" falsevalue="false" />
49      </inputs>
50      <outputs>
51          <data name="outfile" format="fssub" metadata_source="input_image" label="${
                tool.name} on ${input_image.name}" />
52      </outputs>
53      <help>(...)</help>
54  </tool>
```

Note que a *tag* <command> define uma série de comandos na linguagem *Cheetah*. As linhas prefixadas com o caractere # são pré-processadas e o resultado é o *script shell* que será executado. Durante o pré-processamento, as opções selecionadas pelo usuário são levadas em conta. Por exemplo, observe o trecho de código abaixo:

```
1 recon-all
2     $directive
3     #if not exists($subject_files_dir)
4         -i $input_image_symlink
5     #end if
```

A opção selecionada pelo usuário no campo *"Directive to execute"* substituirá o valor de `$directive`. Já o `#if` será usado para determinar se a linha posterior será incluído no resultado final ou não. Ao final do pré-processamento, o resultado final será:

```
1 recon-all --all -i /data/images/pacient.nii.gz
```

Componentes geralmente dependem de executáveis, programas ou bibliotecas para poderem ser executados. O arcabouço de componentes do Galaxy permite que essas relações de dependência sejam mapeadas de duas formas: i) dependências por outros componentes do Galaxy; e ii) dependências por programas ou bibliotecas externas ao Galaxy. A listagem abaixo mostra um exemplo de declaração de dependência a uma biblioteca Python chamada *PyDICOM*:

```
1 <?xml version="1.0"?>
2 <tool_dependency>
3   <package name="pydicom" version="0.9.9">
4     <install version="1.0">
5       <actions>
6         <action type="download_by_url">(...)</action>
7         <action type="make_directory">$INSTALL_DIR/lib/python</action>
8         <action type="shell_command">
9             export PYTHONPATH=$PYTHONPATH:$INSTALL_DIR/lib/python
10            python setup.py install
11                --install-lib $INSTALL_DIR/lib/python
12                --install-scripts $INSTALL_DIR/bin
13        </action>
14        <action type="set_environment">
15            <environment_variable action="prepend_to" name="PYTHONPATH">
16              $INSTALL_DIR/lib/python
17            </environment_variable>
18        </action>
19      </actions>
20    </install>
21  </package>
22 </tool_dependency>
```

Novos componentes podem ser publicados no **Toolshed**, um sistema de gerenciamento de componentes análogo ao *apt-get* ou *yum*. Toolsheds podem ser públicos ou privados, e seus componentes podem ser instalados em instâncias do Galaxy por seus administradores. Desta forma, componentes podem ser distribuídos juntamente com publicações científicas que os originaram, facilitando a reprodutibilidade e transparência dos experimentos publicados.

Até o presente momento desenvolvemos os seguintes componentes para o workflow de análise de imagens:

- `Datatypes` para imagens médicas: DICOM, NIfTI, FreeSurferSubject
- Transferência de imagens via Hubble
- Conversão de arquivos em formato DICOM para formato NIfTI
- Reconstrução cortical, com as seguintes funcionalidades:
  (a) Extração do cérebro
  (b) Rotulação e segmentação da substância branca
  (c) Extração e registro de superfícies
  (d) Extração dos dados estatísticos individuais, por região de interesse (ou ROI, *Region of Interest*), volume e superfície
  (e) Suavização dos dados estatísticos de cada indivíduo
- Criação de tabela consolidada de estatísticas de regiões de interesse do cérebro (ROI, ou *Region of Interest*) para um grupo de indivíduos
- Concatenação dos dados estatísticos de um grupo de indivíduos
- Criação do modelo linear geral, com as seguintes funcionalidades:

(a) Criação do modelo linear geral a partir de estatísticas de ROI de um grupo de indivíduos

(b) Criação do modelo linear geral a partir dos mapas estatísticos de um grupo de indivíduos

Estes componentes foram publicados no Toolshed público deste projeto, que pode ser consultado em http://toolshed.migcloud.org, e seu código fonte está disponível em https://bitbucket.org/migusp/migala tools. Dividimos os componentes nos quatro repositórios abaixo, como mostra a figura 4.12:

1. `datatypes`;
2. conversão entre formatos de imagens médicas;
3. ferramentas do FreeSurfer; e
4. integração com Hubble.

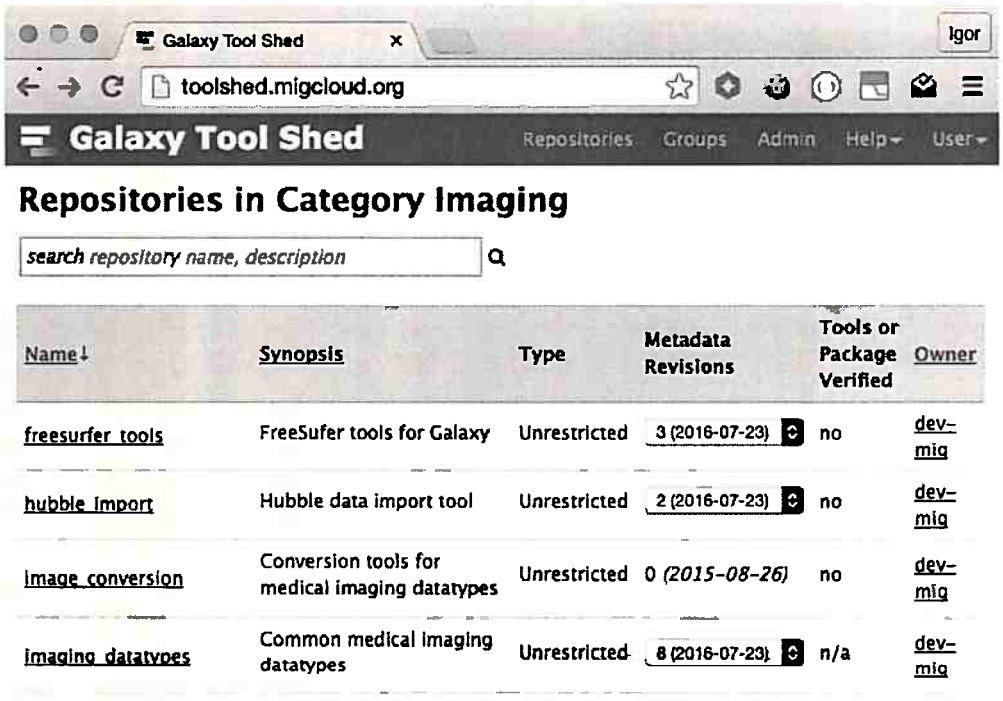

Figura 4.12: *Repositórios de componentes criados no Toolshed.*

Cada repositório armazena um ou mais componentes, e pode definir relações de dependência com outros repositórios. Por exemplo, o repositório "*ferramentas para análise de DTI*" possui quatro componentes disponíveis, e depende do repositório "`datatypes` *para imagens médicas*", como mostra a figura 4.13.

Alguns dos componentes desenvolvidos dependem de binários ou bibliotecas externas ao Galaxy. Nesses casos, existem duas alternativas: i) instala-se as dependências diretamente no servidor Galaxy; ou ii) declara-se as dependências via Toolshed. Usamos a primeira estratégia para instalar as ferramentas computacionais necessárias para processar imagens médicas, como o FSL, AFNI e FreeSurfer. No entanto, utilizamos a segunda estratégia para instalar bibliotecas Python (*e.g.* NumPy e PyDICOM).

Após o desenvolvimento e a publicação desses componentes em um Toolshed, podemos instalá-las em uma instância do Galaxy. O processo de instalação de um componente no Galaxy é semelhante ao processo de instalação de programas no Linux via *apt-get* ou *yum*. Uma vez instalados no Galaxy, os componentes estarão disponíveis para os usuários utilizarem em seus workflows. Abordaremos este assunto na próxima seção.

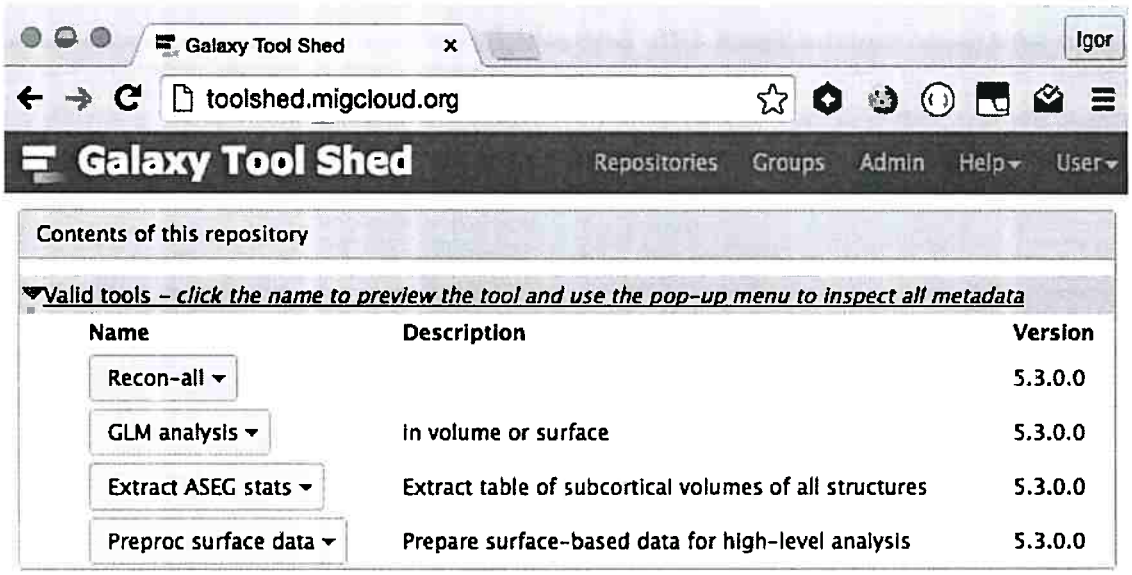

Figura 4.13: *Componentes disponíveis no repositório* "ferramentas do FreeSurfer" *do Toolshed.*

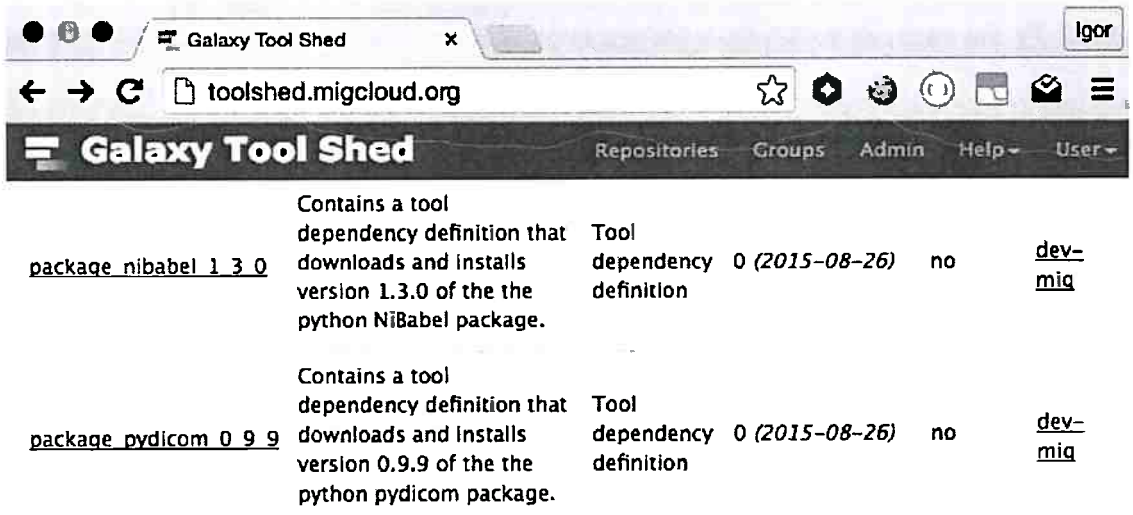

Figura 4.14: *Exemplos de dependências externas (e.g. bibliotecas ou binários) gerenciadas pelo Toolshed.*

4.2.4 Modelagem do Workflow de Análise de DTI

Nesta seção final sobre o WfMS, vamos abordar o método para exploração e análise de imagens médicas através do Galaxy, utilizando os componentes desenvolvidos e publicados no Toolshed.

Todo componente instalado em uma instância de Galaxy ficará disponível no painel esquerdo da aplicação **web**, como ilustra a figura 4.15. Este painel permite a utilização dos componentes de forma *ad-hoc*, ou seja, podemos utilizá-los para processar uma única aquisição DTI, sem que seja necessário modelar e executar um workflow. Por exemplo, um usuário pode querer converter um arquivo em formato DICOM para o formato NIfTI, sem necessariamente construir um workflow para isso.

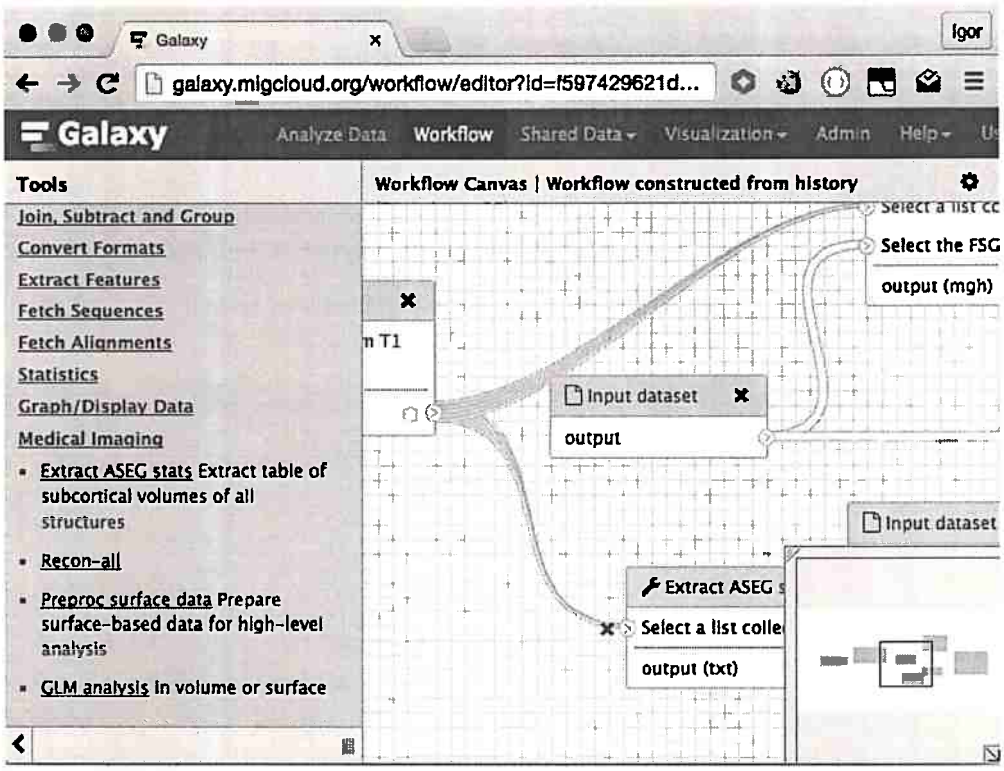

Figura 4.15: *Página edição de workflows do Galaxy, exibindo os componentes de processamento de imagens médicas disponíveis para uso.*

Assim como todo sistema de workflows científicos, o Galaxy é orientado a dados, ou seja, dados são parte central da sua arquitetura. Por esse motivo, o painel direito da aplicação exibe todos os *datasets* originais e processados, mantendo o histórico de transformações e armazenando os metadados de proveniência. A figura 4.16 mostra o painel de histórico, exibindo os *datasets* do usuário autenticado.

Essa abordagem permite que o usuário explore livremente seus dados através das ferramentas disponíveis na aplicação. A figura 4.17 mostra um exemplo de interface que construímos para o componente de *reconstrução cortical*. Note que o usuário pode escolher os dados que deseja processar de diferentes modos: apenas uma image, múltiplas imagens ou uma coleção de imagens. Ao final do processamento, cada *dataset* resultante do processamento é adicionado ao histórico do usuário. Caso seja escolhida uma coleção de imagens, o resultado será uma coleção de *datasets*.

Cada *dataset* possui um ponteiro para o componente que o gerou, e outro para o componente que o consumiu (informações de proveniência). O Galaxy utiliza essas informações para gerar automaticamente um modelo de workflow para o usuário.

A figura 4.18 mostra o workflow gerado pelo Galaxy. Cada caixa representa a execução de um componente (uma atividade). As setas representam a ligação entre a portas de dados de saída e entrada de cada atividade. Note que uma única atividade pode possuir múltiplas portas de entrada e saída.

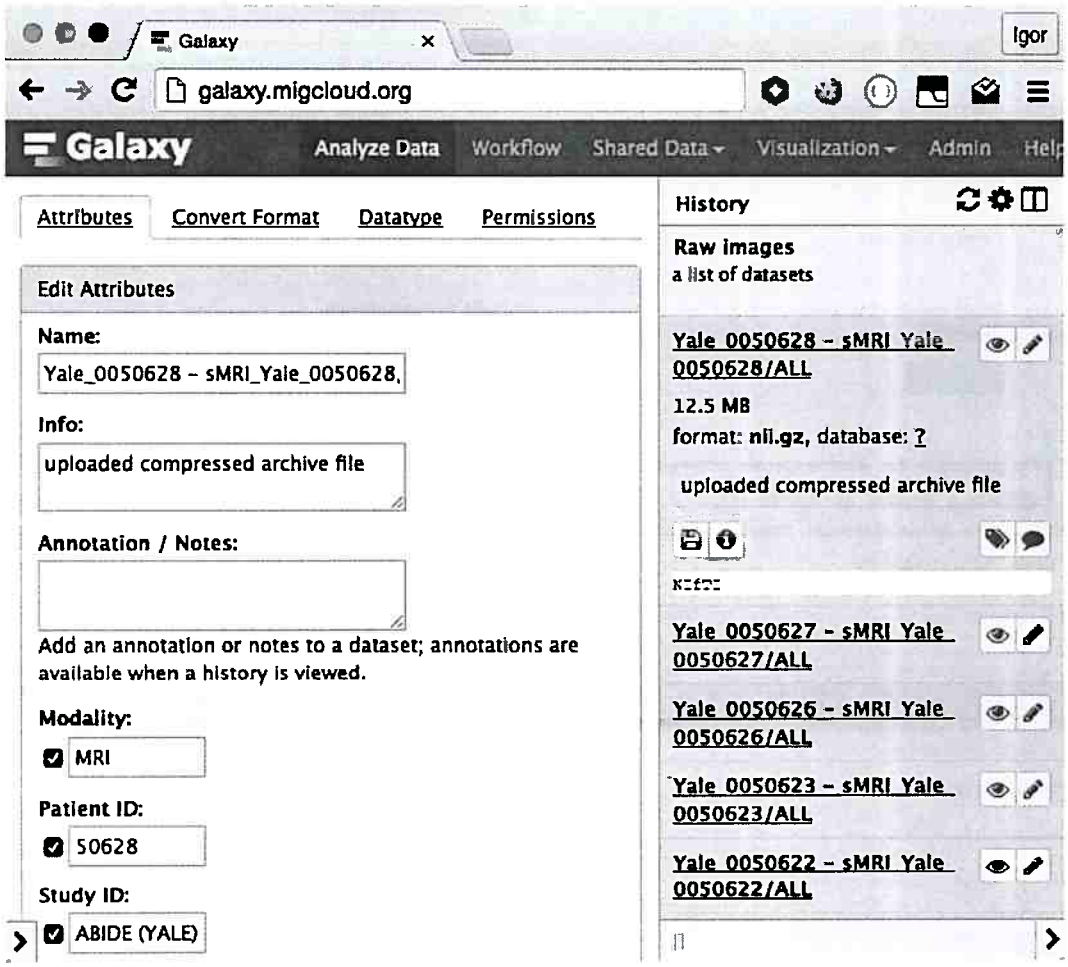

Figura 4.16: *Painel de histórico de* datasets, *exibindo os dados sendo analisados pelo usuário.*

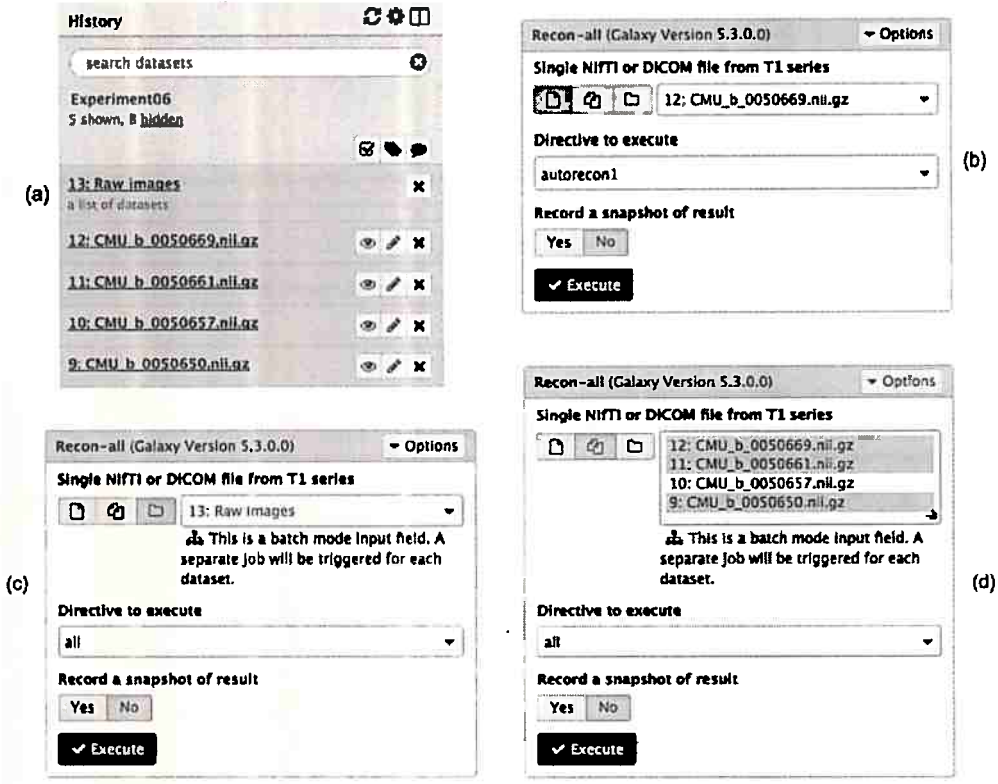

Figura 4.17: *Seleção de dados para processamento: (a) estado atual do histórico do usuário, com uma coleção de datasets e quatro imagens NIfTI; (b) componente Recon-all, com apenas um dataset selecionado; (c) e (d) mostram a seleção de múltiplos datasets de uma só vez. Neste caso, Recon-all será executado múltiplas vezes, em paralelo.*

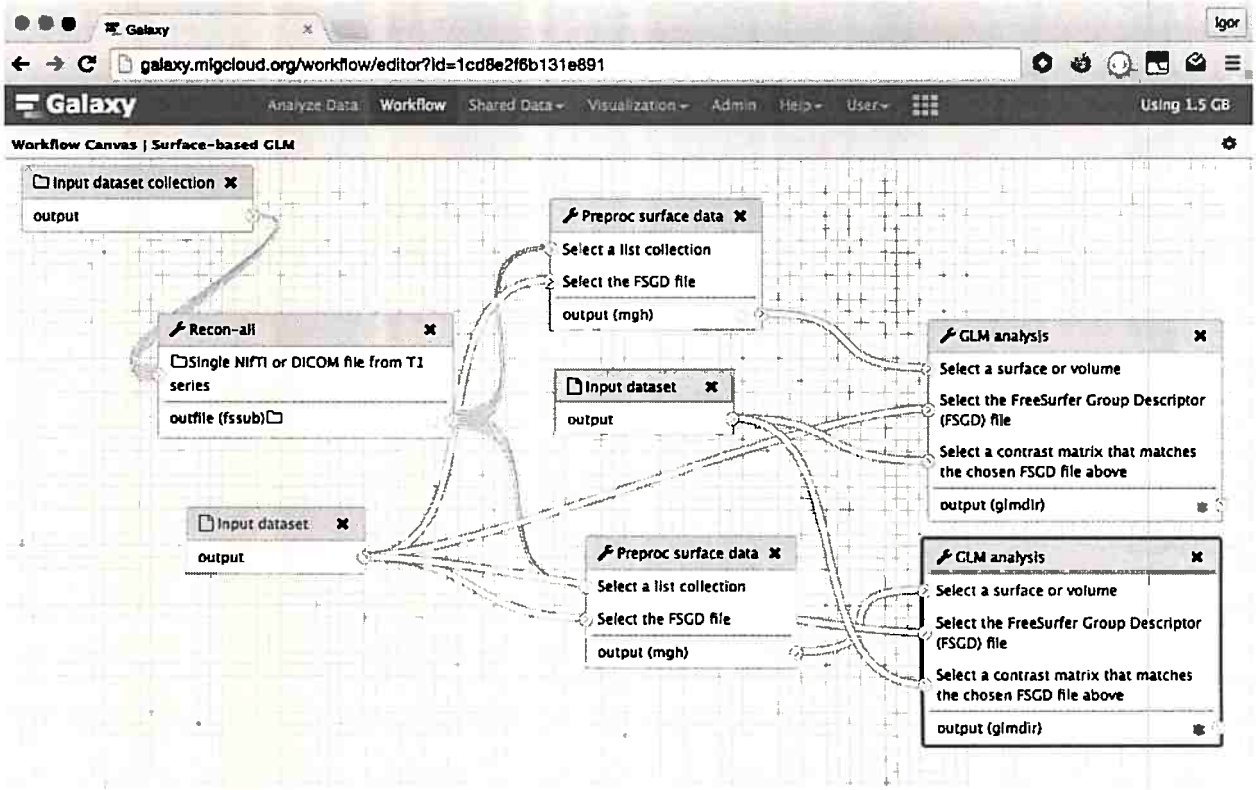

Figura 4.18: *Modelo de workflow gerado pelo Galaxy através dos dados de proveniência no histórico de* datasets *do usuário. Dois modelos estatísticos são gerados: um para cada hemisfério do cérebro.*

Por fim, nosso workflow foi capaz de reproduzir os passos para se obter o modelo linear geral descrito por (Schaer *et al.*, 2015). Em nossa análise encontramos regiões do cérebro que apresentaram resultados significativos para os efeitos do sexo e diagnóstico no volume cortical (com $p < 0.05$), como mostra a figura 4.19. No entanto, ao realizar correção por múltiplas comparações, procedimento adotado pelos autores, não detectamos agrupamentos com $p < 0.001$. O mesmo resultado se repetiu para a análise da espessura ao invés do volume cortical.

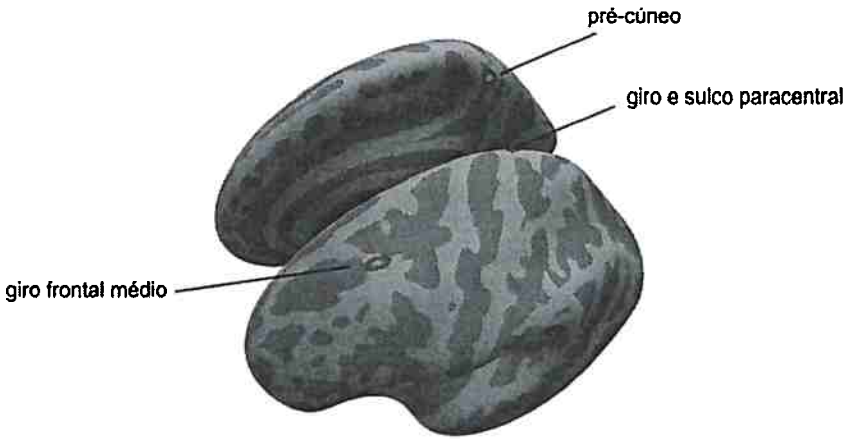

Figura 4.19: *Resultados encontrados para o estudo de 210 indivíduos com e sem TEA. As áreas destacadas mostram as regiões onde há efeitos do sexo e diagnóstico no volume cortical, com $p < 0.05$, sem correção por múltiplas comparações.*

4.3 Galaxy na Nuvem

Afgan *et al.* (2010) propuseram uma nova plataforma chamada **CloudMan**, cujo objetivo é eliminar a complexidade de instalação e configuração do Galaxy em nuvens computacionais. O CloudMan é capaz de criar instâncias pré-configuradas do Galaxy sob demanda através da automação dos procedimentos necessários para criar uma máquina virtual na nuvem computacional, instalar os pré-requisitos de software necessários e finalmente executar o Galaxy. Todo este processo é transparente para o usuário, ou seja, não são necessários conhecimentos prévios de infraestrutura computacional, sistemas operacionais ou programação. O usuário pode realizar seus experimentos em sua instância pessoal do Galaxy pelo tempo que desejar. Ao final dos experimentos, o CloudMan se encarrega de destruir as máquinas virtuais criadas.

Entretanto, esta plataforma não foi criada para o uso colaborativo de dados, workflows e infraestrutura computacional. Uma nova instância de Galaxy é criada sob-demanda para cada usuário, e destruída quando ela não é mais necessária. Nosso trabalho, por outro lado, objetiva a criação de uma plataforma única que possa ser compartilhada com diversos usuários, capaz de aumentar ou diminuir a quantidade de recursos computacionais de acordo com a demanda.

Executamos nossos experimentos em duas nuvens diferentes:

- uma nuvem privada, OpenNebula, configurada em uma única máquina física.
- uma nuvem pública, a Amazon AWS.

Execução de Componentes em Ambiente de Computação Distribuída

O Galaxy é arquiteturalmente dividido em diversos módulos: registro de `datatypes`, modelo do banco de dados, componentes, gerenciador de atividades, dentre outros. No entanto, em tempo de execução os principais processos do Galaxy são:

- a aplicação **web**, responsável pela GUI e por processar as diversas requisições de usuários provenientes das páginas *web* e API RESTful (*Representational State Transfer*).
- o gerenciador de atividades (*job manager*), responsável por coordenar a execução de atividades de um workflow (**e.g.** conversão de arquivo em formato DICOM para NIfTI).

A figura 4.20 mostra um exemplo de configuração do Galaxy, em que há múltiplos processo da aplicação *web*, e múltiplos processos de *job manager*.

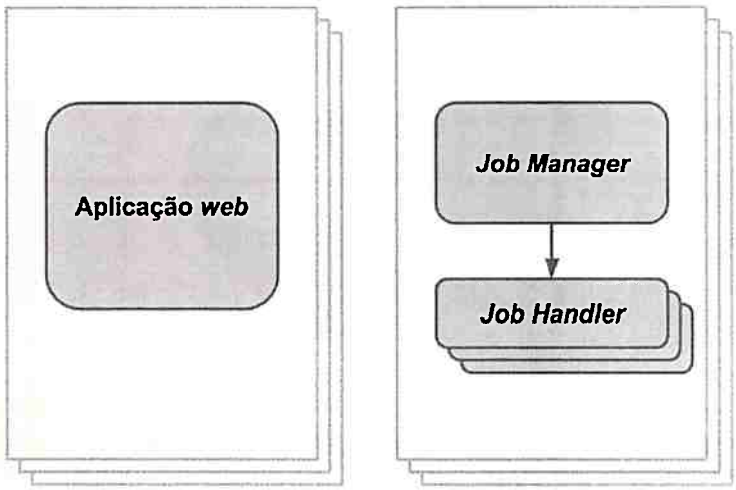

Figura 4.20: *Principais processos do Galaxy: o servidor* web *e o* job manager.

O processo *job manager*, por sua vez, delega o controle das atividades a *threads* especializadas, chamadas *job handlers* (manipulador de atividades). Apenas um processo de *job manager* é executado por vez, porém há várias *threads* de *job handlers* responsáveis pela execução das atividades de um workflow. A figura 4.21 ilustra o fluxo de execução das atividades no Galaxy.

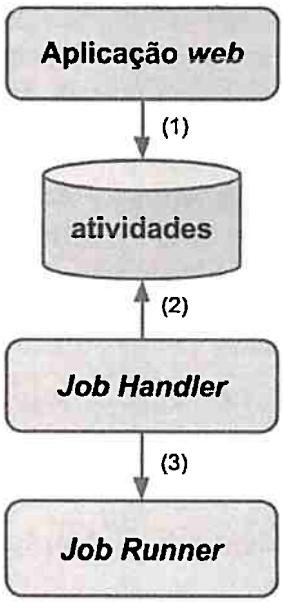

Figura 4.21: *Fluxo de atividades no Galaxy: (1) a interface* web *recebe o comando do usuário para a execução de uma atividade (ou workflow), que é gravada no banco de dados; (2) o processo que executa o* job handler *lê a atividade; (3) o* job runner *efetivamente executa a atividade.*

Em linhas gerais, *job handlers* podem executar atividades de duas formas:

- localmente, ou seja, a própria *thread* do *job handler* executará a atividade; e
- em ambiente distribuído, através de um sistema de gerenciamento de recursos distribuídos.

As diferentes estratégias de execução de atividades são implementadas através dos *job runners*. Por exemplo, a execução local de atividades é implementada pela classe `LocalJobRunner`. Para execução distribuída através do HTCondor, utiliza-se a classe `CondorJobRunner`. Outros DRMS também possuem implementações específicas. A figura 4.22 ilustra como o Galaxy interage com um DRMS.

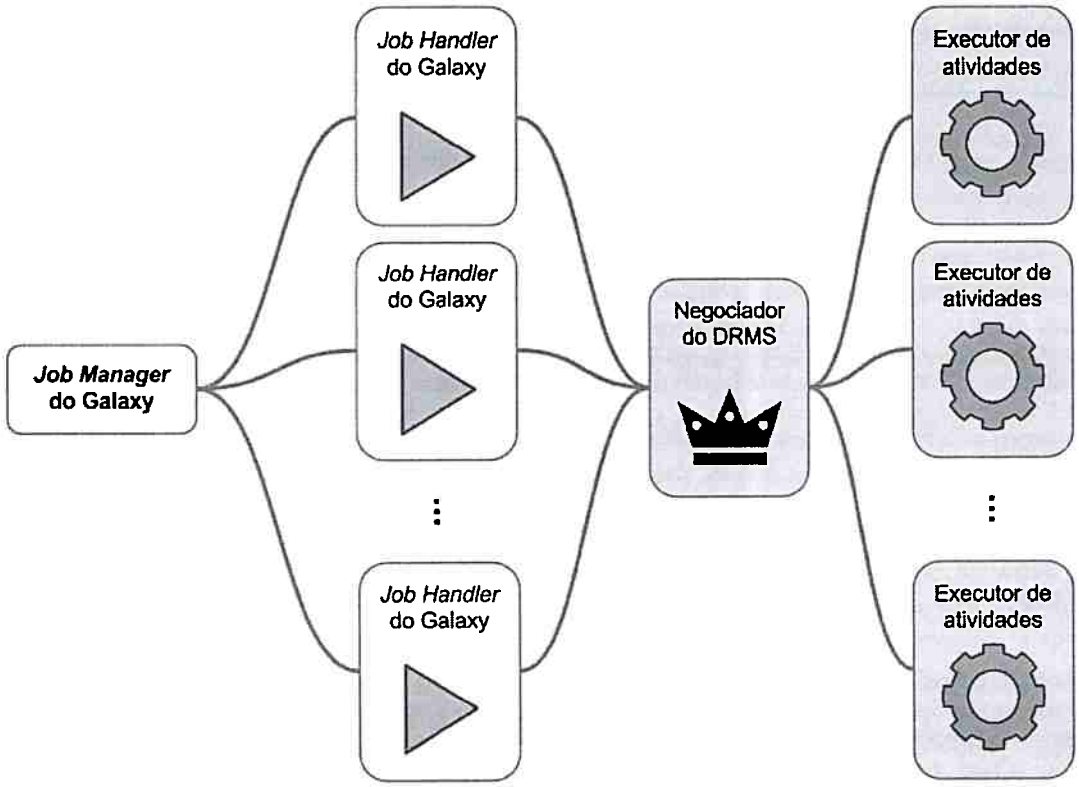

Figura 4.22: *Diagrama ilustrando a interação entre o Galaxy e um DRMS.*

4.3.1 Usando um DRMS para Distribuir Atividades

Nosso interesse é executar os componentes do Galaxy em um ambiente distribuído, portanto é necessário configurá-lo para isso. Dentre as opções de DRMS suportadas pelo Galaxy, optamos pelo **HTCondor**, por ser um projeto de software livre estável, bem documentado e com uma comunidade ativa de usuários e colaboradores. Configuramos o *job runner* da seguinte forma:

```
1 <?xml version="1.0"?>
2 <job_conf>
3     <plugins workers="4">
4         <plugin id="condor" type="runner" load="galaxy.jobs.runners.
              condor:CondorJobRunner" />
5     </plugins>
6     <handlers default="handlers">
7         <handler id="handler0" tags="handlers" />
8     </handlers>
9     <destinations default="condor">
10        <destination id="condor" runner="condor" />
```

```
11      </destinations>
12 </job_conf>
```

Por fim, configuramos o DRMS em uma série de máquinas virtuais capazes de executar as atividades de um workflow, conforme ilustrado pelo figura 4.22. Esta parte da solução arquitetural é a principal responsável pela elasticidade da plataforma que propusemos. O redimensionamento automático (*autoscaling*) da plataforma de nuvem é configurado para aumentar ou diminuir sob demanda a quantidade de máquinas virtuais controladas pelo DRMS.

Conforme abordamos em 3.2.3, no capítulo anterior, cada uma das novas instâncias do DRMS executa uma imagem pré-configurada. Sempre que uma dessas instâncias é inicializada, ela se auto-registra no negociador central do HTCondor. A partir desse momento, o negociador enviará atividades para a nova instância executar.

A transferência de arquivos é um dos principais desafios que encontramos com essa estratégia distribuída. São dois os principais problemas:

- O Galaxy não é capaz de fazer a transferência dos arquivos de entrada e saída ao delegar uma atividade ao HTCondor. A única estratégia que existe no momento é o uso de um sistema de arquivos distribuído.
- É desejável que o escalonamento de atividades considere a localidade de dados, ou seja, evite a transferência desnecessária de arquivos sempre que possível. Atualmente isso não é possível usando apenas o Galaxy e HTCondor como escalonadores.

Este trabalho não contempla a solução dos problemas acima, mas os contorna da seguinte forma:

- Utilizamos o NFS (Network File System), resolvemos de forma simples o problema de distribuição de arquivos para os nós de processamento de atividades.
- Para o problema de localidade dos dados, limitamos nosso uso da Amazon AWS para apenas uma zona de disponibilidade (*availability zone*). Dessa forma, eliminamos o problema de transferências entre diferentes zonas.

4.3.2 Resultados da Execução Distribuída de um Workflow na nuvem

Na seção 4.2.4, mostramos como nossa solução foi capaz de modelar um workflow real, exibindo os resultados finais de uma análise estatística da comparação de 210 indivíduos. Entretanto, para validar a execução distribuída de um workflow em um ambiente de nuvem, realizamos dois tipos de validação: a primeira é a verificação de que o redimensionamento da nuvem é possível quando a demanda por poder computacional aumenta. A segunda é a medição do tempo total de execução do workflow.

Para a primeira parte da análise, submetemos o workflow descrito na seção 4.2.4 para uma máquina virtual do Galaxy configurada para executar suas atividades em um aglomerado de máquinas virtuais do HTCondor. Este aglomerado é controlado por um grupo de *autoscaling* da Amazon AWS, conforme abordamos na seção 3.2.3 do capítulo anterior. As regras de redimensionamento que configuramos foram:

- se o uso de CPU for superior a 50%, adiciona uma nova instância;
- se o uso de CPU for inferior a 10%, remova uma instância do aglomerado;
- o número máximo e mínimo de instâncias no aglomerado é 1 e 3, respectivamente; e
- o tipo de instância é *t2.small* (2GB de RAM, 1 CPU).

Configuramos o Galaxy para enviar ao HTCondor apenas as atividades de pré-processamento, pois estas são as que consomem mais tempo e demandam mais CPU. O gráfico na figura 4.23 mostra o aumento da quantidade de nós no aglomerado de executores. O aumento não ocorre de 1 à 3 instâncias instantaneamente, pois:

- esperamos 5 observações de CPU > 50% antes de ativar a regra de adição de instância;
- o provisionamento da máquina virtual pode levar alguns minutos para ser concluído;

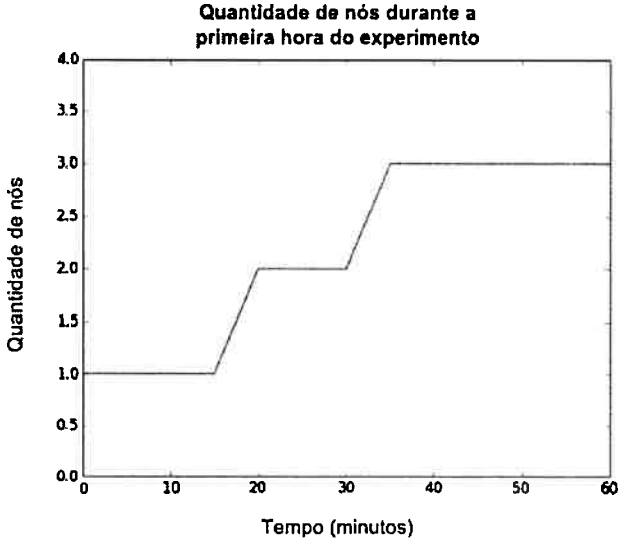

Figura 4.23: *Quantidade de nós no aglomerado de executores a partir de t_0, quando a submissão do workflow foi realizada.*

Também medimos os seguintes resultados, como parte da validação de nosso trabalho:

- Tempo de execução de cada imagem durante o pré-processamento. Para isso, consideramos o processamento realizado em máquinas virtuais com acesso a 1 e 2 processadores Intel Xeon 3.3GHz e pelo menos 2GB de RAM. As máquinas virtuais foram executadas em dois tipos de nuvem: Amazon AWS e OpenNebula. Os tempos de execução foram extraídos a partir dos *logs* de execução do FreeSurfer, que foram armazenados no decorrer desta pesquisa.
- Tempo total de execução de um workflow na AWS.para um total de 10 indivíduos.

A figura 4.24 mostra a distribuição do tempo de execução do pré-processamento das imagens de 200 indivíduos do projeto ABIDE. A média de tempo de processamento foi de $15,29h$, com desvio padrão de $3,06h$. Neste cálculo, a atividade de conversão entre os formatos DICOM e NIfTI não foi contabilizada, pois todas as imagens originais distribuídas pelo ABIDE possuem formato NIfTI. Outras atividades do workflow são executadas em alguns minutos apenas, pois utilizamos o comando `recon-all -qcache` durante o pré-processamento para pré-calcular os mapas de estatísticas suavizados em vários tamanhos de *kernel* e registrados no espaço comum do *fsaverage*.

O tempo total de execução para nosso experimento envolvendo 10 pacientes foi de 65,4 horas, contabilizados a partir da criação do primeiro *dataset* do workflow, até a finalização da escrita do último *dataaset*.

No próximo capítulo, sumarizamos as principais conclusões e contribuições deste trabalho, discutiremos suas limitações e finalizaremos com algumas sugestões para trabalhos futuros.

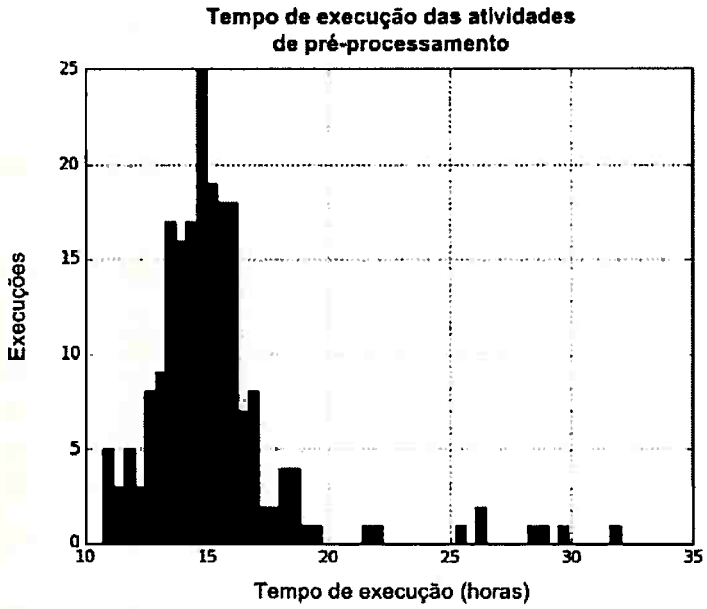

Figura 4.24: *Distribuição do tempo de execução das atividades de reconstrução cortical em nossos experimentos na Amazon AWS.*

Capítulo 5

Conclusão

Ferramentas e recursos computacionais tem se tornado fundamentais para a análise de imagens médicas em diversos campos da medicina, a exemplo do que ocorre em outros domínios da ciência. A complexidade dessas ferramentas de análise e a crescente demanda por poder computacional para executar workflows, no entanto, tornam-se barreiras importantes para qualquer pesquisador sem conhecimentos em programação e computação distribuída. Neste trabalho, identificamos as principais propriedades de uma solução para exploração de imagens médicas que seja capaz de reduzir ou eliminar estas barreiras:

- mediar a recepção e organização de imagens médicas em centros de pesquisa de forma eficiente;
- permitir a exploração dos dados brutos através de ferramentas computacionais de análise de imagens;
- permitir a extensão da solução através da instalação de novas ferramentas computacionais;
- possibilitar a modelagem de workflows que podem ser reproduzidos por outros pesquisadores;
- ser capaz de redimensionar a quantidade de recursos computacionais sob demanda, possibilitando o aumento da capacidade de processamento para análises em grande escala.

Com relação ao problema de recepção e organização de imagens médicas, escolhemos o XNAT para realizar experimentos no CHU. Em termos práticos, o XNAT foi capaz de: i) receber imagens provenientes de equipamentos médicos no CHU; ii) armazenar os resultados em uma nuvem privada OpenNebula; iii) melhorar o processo de distribuição de imagens para os pesquisadores.

Guiados pelos critérios de análise de WfMS que estabelecemos, escolhemos o Galaxy como nosso sistema de gerenciamento de workflows. Embora tenha sido inicialmente projetado para pesquisa genômica, o Galaxy também se adequou aos requisitos para análise de imagens médicas. Nossos principais resultados no âmbito da solução de gerenciamento de workflows foram:

- reprodução de workflows comuns em publicações que envolvem análises estatísticas de grupos populacionais;
- extensão da lista de ferramentas disponíveis para os pesquisadores;
- execução de workflows em um ambiente de computação em nuvem;

Validamos os resultados obtidos através da modelagem e execução de um workflow que mensura os efeitos do diagnóstico de Transtorno do Espectro Autista e do sexo no volume cerebral. Dois tipos de workflow foram avaliados: análise por região de interesse e voxel-a-voxel. Nossos resultados foram similares aos encontrados nos estudos em que nos baseamos.

Quanto ao aspecto da execução de workflows em ambientes distribuídos, realizamos testes na nuvem pública da Amazon AWS. Configuramos um grupo de *autoscaling* para aumentar ou diminuir a quantidade de instâncias de acordo com métricas de uso de CPU. Com os resultados obtidos em

nossos testes, concluímos que: i) é possível redimensionar a quantidade de recursos computacionais sob demanda para processos de análise de imagens que demandam CPU; e ii) conseguimos medir o aumento do grau de paralelismo, resultado do escalonando das atividades em diversos nós de execução do nosso DRMS.

Com os resultados deste trabalho, confiamos que é possível democratizar o uso das ferramentas complexas de processamento de imagens médicas, bem como tornar as publicações na área de neurociência mais transparentes e fáceis de reproduzir.

Limitações

Estudos envolvendo imagens médicas de grupos populacionais geralmente possuem grandes quantidades de dados. Por exemplo, o resultado da reconstrução cortical de um único paciente possui em média 353,3MB. Portanto um estudo com 200 pacientes teria no total 70,7GB de dados de reconstrução. Em um workflow que não leva a localidade dos dados em consideração durante o processo de escalonamento de atividades, pode ocorrer que os 70,7GB tenham que ser transferidos para o nó de execução. Em nossos experimentos, limitamos a região e a zona de disponibilidade da AWS à *us-west-2* (*datacenter* de Portland, OR, nos EUA) para evitar que os dados trafeguem inter-regiões ou zonas. Entretanto, existem formas de minimizar o problema da transferência que não foram abordados neste trabalho.

Uma outra limitação da modelagem de workflows no Galaxy em comparação com a modelagem via *scripts shell* é a flexibilidade do uso de recursos e parâmetros disponíveis apenas para o *shell*. Pode-se, por exemplo, manipular livremente as entradas e saídas, sem se prender às limitações da interface de usuário do WfMS. O nível de abstração que muitas vezes é desejado em um WfMS, pode se tornar um fator limitante para alguns casos de uso.

No que se refere a visualização dos resultados no WfMS, também existem limitações. Os workflows desenvolvidos neste trabalho tem como objetivo coletar e processar dados e disponibilizar os resultados para o usuário. Entretanto, a visualização destes resultados ainda é realizada no computador do pesquisador. Idealmente, a plataforma possibilitaria a exploração dos resultados, volumes e superfícies sem a necessidade de se transferir os arquivos e abri-los localmente.

Trabalhos Futuros

Além da extensão da quantidade de ferramentas disponíveis no Galaxy e das melhorias na arquitetura da solução aqui proposta, existem outras possibilidades de trabalhos futuros na área de workflows para imagens médicas.

Por exemplo, os testes descritos neste trabalho foram executados em ambientes controlados, sem o uso intensivo de seus componentes por cientistas realizando experimentos, submetendo atividades e executando workflows com grandes quantidades de dados. Os resultados deste trabalho, como o código fonte, estratégias utilizadas e análises dos sistemas empregados na solução poderiam dar origem a uma nuvem acadêmica, que poderia ser utilizada em grande escala por pesquisadores interessados em contribuir com o desenvolvimento do projeto.

Outro exemplo é a aplicação dos conceitos deste trabalho em outros áreas da medicina, como a na rotina clínica, possibilitando o uso de workflows de processamento de imagens não só na pesquisa, mas também como ferramenta de auxílio no diagnóstico.

Bibliografia

OAS (2007) Web Services Business Process Execution Language Version 2.0. Relatório técnico. Citado na pág. 17

OMG (2011) Business Process Model and Notation (BPMN). Relatório técnico. Citado na pág. 17

Aalst e Hee (2002) Wil Aalst e Kees M. Hee. Workflow management models, methods, and systems. Hardcover, Janeiro 2002. URL http://www.worldcat.org/isbn/0262011891. Citado na pág. 9

Afgan *et al.* (2010) Enis Afgan, Dannon Baker, Nate Coraor, Brad Chapman, Anton Nekrutenko e James Taylor. Galaxy CloudMan: delivering cloud compute clusters. *BMC Bioinformatics*, 11 (Suppl 12):S4+. ISSN 1471-2105. doi: 10.1186/1471-2105-11-s12-s4. URL http://dx.doi.org/10.1186/1471-2105-11-s12-s4. Citado na pág. 73

Afgan *et al.* (2011) Enis Afgan, Jeremy Goecks, Dannon Baker, Nate Coraor, Anton Nekrutenko e James Taylor. Galaxy: A Gateway to Tools in e-Science. Em Xiaoyu Yang, Lizhe Wang e Wei Jie, editors, *Guide to e-Science*, Computer Communications and Networks, páginas 145–177. Springer London. doi: 10.1007/978-0-85729-439-5_6. URL http://dx.doi.org/10.1007/978-0-85729-439-5_6. Citado na pág. 6

Altintas *et al.* (2004) Ilkay Altintas, Chad Berkley, Efrat Jaeger, Matthew Jones, Bertram Ludascher e Steve Mock. Kepler: an extensible system for design and execution of scientific workflows. Em *Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on*, páginas 423–424. IEEE. ISBN 0-7695-2146-0. doi: 10.1109/ssdm.2004.1311241. URL http://dx.doi.org/10.1109/ssdm.2004.1311241. Citado na pág. 7

Architecture () Example XNAT Architecture. Example xnat architecture at washington university. https://wiki.xnat.org/display/XNAT16/Example+XNAT+Architecture. Último acesso em 30/06/2016. Citado na pág. 56

AWS () Amazon AWS. http://aws.amazon.com. Último acesso em 30/06/2016. Citado na pág. 6

Barga e Gannon (2007) Roger Barga e Dennis Gannon. Scientific versus Business Workflows. páginas 9–16. doi: 10.1007/978-1-84628-757-2_2. URL http://dx.doi.org/10.1007/978-1-84628-757-2_2. Citado na pág. 12

Bezerra *et al.* (2012) Diana M. Bezerra, Fabrício R. S. Pereira, Fernando Cendes, Marcel P. Jackowski, Eduardo Y. Nakano, Marco A. A. Moscoso, Salma R. I. Ribeiz, Renata Ávila, Cláudio C. Castro e Cássio M. C. Bottino. DTI voxelwise analysis did not differentiate older depressed patients from older subjects without depression. *Journal of Psychiatric Research*, 46(12):1643–1649. ISSN 00223956. doi: 10.1016/j.jpsychires.2012.09.001. URL http://dx.doi.org/10.1016/j.jpsychires.2012.09.001. Citado na pág. 41, 42

Bharathi *et al.* (2008) S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, Mei-Hui Su e K. Vahi. Characterization of scientific workflows. Em *Workflows in Support of Large-Scale Science, 2008. WORKS 2008. Third Workshop on*, páginas 1–10. IEEE. ISBN 978-1-4244-2827-4. doi: 10.1109/works.2008.4723958. URL http://dx.doi.org/10.1109/works.2008.4723958. Citado na pág. 19

Braghetto e Cordeiro (2014) Kelly R. Braghetto e Daniel Cordeiro. Introdução à Modelagem e Execução de Workflows Científicos. Em *Atualizações em Informática*, páginas 1–40. Sociedade Brasileira de Computação. Citado na pág. 2, 31

Cerezo (2013) Nadia Cerezo. *Conceptual Workflows*. Tese de Doutorado, Université Nice Sophia Antipolis. URL https://tel.archives-ouvertes.fr/tel-00942559. Citado na pág. xi, 7, 10, 13, 18, 21, 22, 24

CGFB () CGFB. Centre de génomique fonctionnelle de bordeaux. http://services.cbib.u-bordeaux2.fr/galaxy/. Último acesso em 30/06/2016. Citado na pág. 6

Chen *et al.* (2014) Shiping Chen, Tomasz Bednarz, Piotr Szul, Dadong Wang, Yulia Arzhaeva, Neil Burdett, Alex Khassapov, John Zic, Surya Nepal, Tim Gurevey e John Taylor. Galaxy + Hadoop: Toward a Collaborative and Scalable Image Processing Toolbox in Cloud. Em AlessioR Lomuscio, Surya Nepal, Fabio Patrizi, Boualem Benatallah e Ivona Brandić, editors, *Service-Oriented Computing - ICSOC 2013 Workshops*, volume 8377 of *Lecture Notes in Computer Science*, páginas 339–351. Springer International Publishing. doi: 10.1007/978-3-319-06859-6_30. URL http://dx.doi.org/10.1007/978-3-319-06859-6_30. Citado na pág. 6

Courses (a) FreeSurfer Courses. Freesurfer courses at martinos center for biomedical imaging. http://freesurfer.net/fswiki/CourseDescription, a. Último acesso em 30/06/2016. Citado na pág. 2

Courses (b) FSL Courses. Fsl courses at centre for functional mri of the brain. http://fsl.fmrib.ox.ac.uk/fslcourse/, b. Último acesso em 30/06/2016. Citado na pág. 2

Cox (2012) Robert W. Cox. AFNI: What a long strange trip it's been. *NeuroImage*, 62(2):743–747. ISSN 10538119. doi: 10.1016/j.neuroimage.2011.08.056. URL http://dx.doi.org/10.1016/j.neuroimage.2011.08.056. Citado na pág. 2

CSIRO () CSIRO. Commonwealth scientific and industrial research organisation. http://www.csiro.au. Último acesso em 30/06/2016. Citado na pág. 6

Dale *et al.* (1999) A. M. Dale, B. Fischl e M. I. Sereno. Cortical surface-based analysis. I. Segmentation and surface reconstruction. *NeuroImage*, 9(2):179–194. ISSN 1053-8119. doi: 10.1006/nimg.1998.0395. URL http://dx.doi.org/10.1006/nimg.1998.0395. Citado na pág. 45

De Roure *et al.* (2009) David De Roure, Carole Goble e Robert Stevens. The design and realisation of the Virtual Research Environment for social sharing of workflows. *Future Generation Computer Systems*, 25(5):561–567. ISSN 0167739X. doi: 10.1016/j.future.2008.06.010. URL http://dx.doi.org/10.1016/j.future.2008.06.010. Citado na pág. 2

Deelman *et al.* (2014) Ewa Deelman, Karan Vahi, Gideon Juve, Mats Rynge, Scott Callaghan, Philip J. Maechling, Rajiv Mayani, Weiwei Chen, Rafael Ferreira da Silva, Miron Livny e Kent Wenger. Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*. ISSN 0167739X. doi: 10.1016/j.future.2014.10.008. URL http://dx.doi.org/10.1016/j.future.2014.10.008. Citado na pág. 2

Di Martino *et al.* (2014) A. Di Martino, C-G G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto, B. Deen, S. Delmonte, I. Dinstein, B. Ertl-Wagner, D. A. Fair, L. Gallagher, D. P. Kennedy, C. L. Keown, C. Keysers, J. E. Lainhart, C. Lord, B. Luna, V. Menon, N. J. Minshew, C. S. Monk, S. Mueller, R-A A. Müller, M. B. Nebel, J. T. Nigg, K. O'Hearn, K. A. Pelphrey, S. J. Peltier, J. D. Rudie, S. Sunaert, M. Thioux, J. M. Tyszka, L. Q. Uddin, J. S. Verhoeven, N. Wenderoth, J. L. Wiggins, S. H. Mostofsky e M. P. Milham. The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. *Molecular psychiatry*, 19(6):659–667. ISSN 1476-5578. doi: 10.1038/mp.2013.78. URL http://dx.doi.org/10.1038/mp.2013.78. Citado na pág. 44

Dinov (2009) Ivo Dinov. Efficient, Distributed and Interactive Neuroimaging Data Analysis using the LONI Pipeline. *Frontiers in Neuroinformatics*, 3. ISSN 16625196. doi: 10.3389/neuro.11.022.2009. URL http://dx.doi.org/10.3389/neuro.11.022.2009. Citado na pág. 7

Duncan *et al.* (2004) James S. Duncan, Xenophon Papademetris, Jing Yang, Marcel Jackowski, Xiaolan Zeng e Lawrence H. Staib. Geometric strategies for neuroanatomic analysis from {MRI}. *NeuroImage*, 23, Supplement 1:S34 – S45. ISSN 1053-8119. doi: http://dx.doi.org/10.1016/j.neuroimage.2004.07.027. URL http://www.sciencedirect.com/science/article/pii/S1053811904003921. Mathematics in Brain Imaging. Citado na pág. 42

Fedorov *et al.* (2012) Andriy Fedorov, Reinhard Beichel, Jayashree Kalpathy-Cramer, Julien Finet, Jean-Christophe Fillion-Robin, Sonia Pujol, Christian Bauer, Dominique Jennings, Fiona Fennessy, Milan Sonka, John Buatti, Stephen Aylward, James V. Miller, Steve Pieper e Ron Kikinis. 3D Slicer as an image computing platform for the Quantitative Imaging Network. *Magnetic Resonance Imaging*, 30(9):1323–1341. ISSN 0730725X. doi: 10.1016/j.mri.2012.05.001. URL http://dx.doi.org/10.1016/j.mri.2012.05.001. Citado na pág. 56

Fernandes *et al.* (2011) M. Fernandes, J. Sato, G. Busatto e C. Thomaz. Mapeamento Estatístico Paramétrico das Alterações Estruturais Cerebrais devido à Doença de Alzheimer e ao Transtorno Cognitivo Leve. páginas 22–27. ISSN 2175-6120. Citado na pág. 41, 42

Fischl *et al.* (2002) Bruce Fischl, David H. Salat, Evelina Busa, Marilyn Albert, Megan Dieterich, Christian Haselgrove, Andre van der Kouwe, Ron Killiany, David Kennedy, Shuna Klaveness, Albert Montillo, Nikos Makris, Bruce Rosen e Anders M. Dale. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33(3):341–355. ISSN 0896-6273. URL http://view.ncbi.nlm.nih.gov/pubmed/11832223. Citado na pág. 45

Foster e Kesselman (2004) Ian Foster e Carl Kesselman. The Grid 2: Blueprint for a New Computing Infrastructure. Hardcover, Dezembro 2004. URL http://www.worldcat.org/isbn/1558609334. Citado na pág. 28

Foster *et al.* (2008) Ian Foster, Yong Zhao, Ioan Raicu e Shiyong Lu. Cloud Computing and Grid Computing 360-Degree Compared. Em *Grid Computing Environments Workshop, 2008. GCE '08*, páginas 1–10. IEEE. ISBN 978-1-4244-2860-1. doi: 10.1109/gce.2008.4738445. URL http://dx.doi.org/10.1109/gce.2008.4738445. Citado na pág. 31

Garijo *et al.* (2014) Daniel Garijo, Oscar Corcho, Yolanda Gil, Meredith N. Braskie, Derrek Hibar, Xue Hua, Neda Jahanshad, Paul Thompson e Arthur W. Toga. Workflow Reuse in Practice: A Study of Neuroimaging Pipeline Users. Em *e-Science (e-Science), 2014 IEEE 10th International Conference on*, volume 1, páginas 239–246. IEEE. ISBN 978-1-4799-4288-6. doi: 10.1109/escience.2014.33. URL http://dx.doi.org/10.1109/escience.2014.33. Citado na pág. 7

Goecks *et al.* (2010) Jeremy Goecks, Anton Nekrutenko, James Taylor e $Author. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, 11(8):R86+. ISSN 1465-6906. doi: 10.1186/gb-2010-11-8-r86. URL http://dx.doi.org/10.1186/gb-2010-11-8-r86. Citado na pág. 2, 6, 10, 12

Görlach *et al.* (2011) Katharina Görlach, Mirko Sonntag, Dimka Karastoyanova, Frank Leymann e Michael Reiter. Conventional Workflow Technology for Scientific Simulation. Em Xiaoyu Yang, Lizhe Wang e Wei Jie, editors, *Guide to e-Science*, Computer Communications and Networks, páginas 323–352. Springer London. doi: 10.1007/978-0-85729-439-5_12. URL http://dx.doi.org/10.1007/978-0-85729-439-5_12. Citado na pág. 11, 12

Haux (2003) R. Haux. *Strategic Information Management in Hospitals : an Introduction to Hospital Information Systems*. Springer. ISBN 0387403566. URL http://www.worldcat.org/isbn/0387403566. Citado na pág. 10

Hey e Trefethen (2003) T. Hey e A. Trefethen. The Data Deluge: An e-Science Perspective. páginas 809–824. doi: 10.1002/0470867167.ch36. URL http://dx.doi.org/10.1002/0470867167.ch36. Citado na pág. 2, 3

Hindman *et al.* (2011) Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy Katz, Scott Shenker e Ion Stoica. Mesos: A Platform for Fine-grained Resource Sharing in the Data Center. Em *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, páginas 295–308, Berkeley, CA, USA. USENIX Association. URL http://portal.acm.org/citation.cfm?id=1972488. Citado na pág. 26

Hollingsworth (1993) David Hollingsworth. The Workflow Reference Model, 1993. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.198.5206&rep=rep1&type=pdf. Citado na pág. 9

Jackowski *et al.* (2005) Marcel Jackowski, Chiu Yen Kao, Maolin Qiu, R. Todd Constable e Lawrence H. Staib. White matter tractography by anisotropic wavefront evolution and diffusion tensor imaging. *Medical Image Analysis*, 9(5):427 – 440. ISSN 1361-8415. doi: http://dx.doi.org/10.1016/j.media.2005.05.008. URL http://www.sciencedirect.com/science/article/pii/S1361841505000617. Medical Image Computing and Computer-Assisted Intervention - {MICCAI} 2004Medical Image Computing and Computer-Assisted Intervention. Citado na pág. 36

James e Dasarathy (2014) Alex P. James e Belur V. Dasarathy. Medical image fusion: A survey of the state of the art. *Information Fusion*, 19:4–19. ISSN 15662535. doi: 10.1016/j.inffus.2013.12.002. URL http://dx.doi.org/10.1016/j.inffus.2013.12.002. Citado na pág. 40

Korkhov *et al.* (2013) Vladimir Korkhov, Dagmar Krefting, Tamas Kukla, GaborZ Terstyanszky, MatthanW Caan e SilviaD Olabarriaga. Exploring Workflow Interoperability for Neuroimage Analysis on the SHIWA Platform. 11(3):505–522. doi: 10.1007/s10723-013-9262-7. URL http://dx.doi.org/10.1007/s10723-013-9262-7. Citado na pág. 7, 17

Kucharsky Hiess *et al.* (2015) R. Kucharsky Hiess, R. Alter, S. Sojoudi, B. A. Ardekani, R. Kuzniecky e H. R. Pardoe. Corpus Callosum Area and Brain Volume in Autism Spectrum Disorder: Quantitative Analysis of Structural MRI from the ABIDE Database. 45(10):3107–3114. doi: 10.1007/s10803-015-2468-8. URL http://dx.doi.org/10.1007/s10803-015-2468-8. Citado na pág. 44, 46, 61

Lefebvre *et al.* (2015) Aline Lefebvre, Anita Beggiato, Thomas Bourgeron e Roberto Toro. Neuroanatomical Diversity of Corpus Callosum and Brain Volume in Autism: Meta-analysis, Analysis of the Autism Brain Imaging Data Exchange Project, and Simulation. *Biological Psychiatry*, 78(2):126–134. ISSN 00063223. doi: 10.1016/j.biopsych.2015.02.010. URL http://dx.doi.org/10.1016/j.biopsych.2015.02.010. Citado na pág. 44, 46, 61

Liu *et al.* (2012) Xiao Liu, Dong Yuan, Gaofeng Zhang, Wenhao Li, Dahai Cao, Qiang He, Jinjun Chen e Yun Yang. *The Design of Cloud Workflow Systems*. Springer New York, New York, NY. ISBN 978-1-4614-1932-7. doi: 10.1007/978-1-4614-1933-4. URL http://dx.doi.org/10.1007/978-1-4614-1933-4. Citado na pág. 9, 29

Maintz e Viergever (1998) J. B. Maintz e M. A. Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1–36. ISSN 1361-8415. URL http://view.ncbi.nlm.nih.gov/pubmed/10638851. Citado na pág. 34

Marcus *et al.* (2007) DanielS Marcus, TimothyR Olsen, Mohana Ramaratnam e RandyL Buckner. The extensible neuroimaging archive toolkit. 5(1):11–33. doi: 10.1385/ni\%253a5\%253a1\%253a11. URL http://dx.doi.org/10.1385/ni%253a5%253a1%253a11. Citado na pág. 55

Mechelli *et al.* **(2005)** Andrea Mechelli, Cathy Price, Karl Friston e John Ashburner. Voxel-Based Morphometry of the Human Brain: Methods and Applications. *Current Medical Imaging Reviews*, 1(2):105–113. ISSN 15734056. doi: 10.2174/1573405054038726. URL http://dx.doi.org/10.2174/1573405054038726. Citado na pág. 42

Moreno-Vozmediano *et al.* **(2012)** Rafael Moreno-Vozmediano, Rubén S. Montero e Ignacio M. Llorente. IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures. *Computer*, 45(12):65–72. ISSN 0018-9162. doi: 10.1109/mc.2012.76. URL http://dx.doi.org/10.1109/mc.2012.76. Citado na pág. 6

Mori e Tournier (2013) S. Mori e J-Donald Tournier. Introduction to Diffusion Tensor Imaging and Higher Order Models, 2013. URL http://www.worldcat.org/isbn/0123984076. Citado na pág. viii, 36, 37

Mori (2007a) Susumu Mori. Chapter 4 - principle of diffusion tensor imaging. Em Susumu Mori, editor, *Introduction to Diffusion Tensor Imaging*, páginas 33 – 40. Elsevier Science B.V., Amsterdam. ISBN 978-0-444-52828-5. doi: http://dx.doi.org/10.1016/B978-044452828-5/50018-1. URL http://www.sciencedirect.com/science/article/pii/B9780444528285500181. Citado na pág. 36

Mori (2007b) Susumu Mori. Chapter 5 - mathematics of diffusion tensor imaging. Em Susumu Mori, editor, *Introduction to Diffusion Tensor Imaging*, páginas 41 – 47. Elsevier Science B.V., Amsterdam. ISBN 978-0-444-52828-5. doi: http://dx.doi.org/10.1016/B978-044452828-5/50019-3. URL http://www.sciencedirect.com/science/article/pii/B9780444528285500193. Citado na pág. 36

Newman (2015) Sam Newman. Building microservices : designing fine-grained systems, 2015. URL http://www.worldcat.org/isbn/9781491950357. Citado na pág. 55

Oakley *et al.* **(2014)** Todd Oakley, Markos Alexandrou, Roger Ngo, M. Pankey, Celia K. Churchill, William Chen e Karl Lopker. Osiris: accessible and reproducible phylogenetic and phylogenomic analyses within the Galaxy workflow management system. *BMC Bioinformatics*, 15 (1):230+. ISSN 1471-2105. doi: 10.1186/1471-2105-15-230. URL http://dx.doi.org/10.1186/1471-2105-15-230. Citado na pág. 6

Olabarriaga *et al.* **(2014)** S. Olabarriaga, G. Pierantoni, G. Taffoni, E. Sciacca, M. Jaghoori, V. Korkhov, G. Castelli, C. Vuerli, U. Becciani, E. Carley e B. Bentley. Scientific Workflow Management – For Whom? Em *e-Science (e-Science), 2014 IEEE 10th International Conference on*, volume 1, páginas 298–305. IEEE. ISBN 978-1-4799-4288-6. doi: 10.1109/escience.2014.8. URL http://dx.doi.org/10.1109/escience.2014.8. Citado na pág. 5, 7, 12, 13, 15

OpenStack () OpenStack. http://www.openstack.org. Último acesso em 30/06/2016. Citado na pág. 6

Owens (2010) Dustin Owens. Securing Elasticity in the Cloud. *Commun. ACM*, 53(6):46–51. ISSN 0001-0782. doi: 10.1145/1743546.1743565. URL http://dx.doi.org/10.1145/1743546.1743565. Citado na pág. 30

Plankensteiner *et al.* **(2011)** Kassian Plankensteiner, Johan Montagnat e Radu Prodan. IWIR: A Language Enabling Portability Across Grid Workflow Systems. Em *Proceedings of the 6th Workshop on Workflows in Support of Large-scale Science*, WORKS '11, páginas 97–106, New York, NY, USA. ACM. ISBN 978-1-4503-1100-7. doi: 10.1145/2110497.2110509. URL http://dx.doi.org/10.1145/2110497.2110509. Citado na pág. 17

Plankensteiner *et al.* **(2013)** Kassian Plankensteiner, Radu Prodan, Matthias Janetschek, Thomas Fahringer, Johan Montagnat, David Rogers, Ian Harvey, Ian Taylor, Ákos Balaskó e Péter Kacsuk. Fine-Grain Interoperability of Scientific Workflows in Distributed Computing Infrastructures. 11(3):429–455. doi: 10.1007/s10723-013-9261-8. URL http://dx.doi.org/10.1007/s10723-013-9261-8. Citado na pág. 17

Rackspace () Rackspace. http://www.rackspace.com/. Último acesso em 30/06/2016. Citado na pág. 6

Rangayyan (2005) Rangaraj M. Rangayyan. Biomedical image analysis, 2005. URL http://www.worldcat.org/isbn/9780203492543. Citado na pág. vii, 34, 35, 38, 40

Rice *et al.* (2014) Peter Rice, Ian Longden e Alan Bleasby. EMBOSS: The European Molecular Biology Open Software Suite. *Trends in Genetics*, 16(6):276–277. ISSN 01689525. doi: 10.1016/s0168-9525(00)02024-2. URL http://dx.doi.org/10.1016/s0168-9525(00)02024-2. Citado na pág. 6

Schaer *et al.* (2015) Marie Schaer, John Kochalka, Aarthi Padmanabhan, Kaustubh Supekar e Vinod Menon. Sex differences in cortical volume and gyrification in autism. *Molecular Autism*, 6(1). ISSN 2040-2392. doi: 10.1186/s13229-015-0035-y. URL http://dx.doi.org/10.1186/s13229-015-0035-y. Citado na pág. 44, 45, 46, 55, 61, 73

Semmlow e Griffel (2014) John L. Semmlow e Benjamin Griffel. Biosignal and Medical Image Processing, Third Edition, 2014. URL http://www.worldcat.org/isbn/9781466567375. Citado na pág. 34

Smith e Webb (2010) Nadine B. Smith e Andrew Webb. *Introduction to Medical Imaging*. Cambridge University Press, Cambridge. ISBN 9780511760976. doi: 10.1017/cbo9780511760976. URL http://dx.doi.org/10.1017/cbo9780511760976. Citado na pág. 34

Smith (2002) Stephen M. Smith. Fast robust automated brain extraction. *Human brain mapping*, 17(3):143–155. ISSN 1065-9471. doi: 10.1002/hbm.10062. URL http://dx.doi.org/10.1002/hbm.10062. Citado na pág. 42

Smith *et al.* (2006) Stephen M. Smith, Mark Jenkinson, Heidi Johansen-Berg, Daniel Rueckert, Thomas E. Nichols, Clare E. Mackay, Kate E. Watkins, Olga Ciccarelli, M. Zaheer Cader, Paul M. Matthews e Timothy E. J. Behrens. Tract-based spatial statistics: Voxelwise analysis of multi-subject diffusion data. *NeuroImage*, 31(4):1487–1505. ISSN 10538119. doi: 10.1016/j.neuroimage.2006.02.024. URL http://dx.doi.org/10.1016/j.neuroimage.2006.02.024. Citado na pág. 41

Staples (2006) Garrick Staples. TORQUE Resource Manager. Em *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, SC '06, New York, NY, USA. ACM. ISBN 0-7695-2700-0. doi: 10.1145/1188455.1188464. URL http://dx.doi.org/10.1145/1188455.1188464. Citado na pág. 26

Teixeira *et al.* (2014) Ana M. Teixeira, Ana Kleinman, Marcus Zanetti, Marcel Jackowski, Fábio Duran, Fabrício Pereira, Beny Lafer, Geraldo F. Busatto e Sheila C. Caetano. Preserved white matter in unmedicated pediatric bipolar disorder. *Neuroscience Letters*, 579:41–45. ISSN 03043940. doi: 10.1016/j.neulet.2014.06.061. URL http://dx.doi.org/10.1016/j.neulet.2014.06.061. Citado na pág. 2

Thain *et al.* (2005) Douglas Thain, Todd Tannenbaum e Miron Livny. Distributed Computing in Practice: The Condor Experience. *Concurrency and Computation: Practice and Experience*, 17: 2–4. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.6.3035. Citado na pág. 26

Tröger e Merzky (2014) Peter Tröger e Andre Merzky. Towards Standardized Job Submission and Control in Infrastructure Clouds. 12(1):111–125. doi: 10.1007/s10723-013-9275-2. URL http://dx.doi.org/10.1007/s10723-013-9275-2. Citado na pág. 27, 28, 32, 50

Vavilapalli *et al.* (2013) Vinod K. Vavilapalli, Arun C. Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, Bikas Saha, Carlo Curino, Owen O'Malley, Sanjay Radia, Benjamin Reed e Eric Baldeschwieler. Apache Hadoop YARN: Yet Another Resource Negotiator. Em *Proceedings of the 4th Annual Symposium on Cloud Computing*, SOCC '13, New York, NY, USA. ACM. ISBN 978-1-4503-2428-1. doi: 10.1145/2523616.2523633. URL http://dx.doi.org/10.1145/2523616.2523633. Citado na pág. 26

Vöckler *et al.* (2011) Jens S. Vöckler, Gideon Juve, Ewa Deelman, Mats Rynge e Bruce Berriman. Experiences Using Cloud Computing for a Scientific Workflow Application. Em *Proceedings of the 2Nd International Workshop on Scientific Cloud Computing*, ScienceCloud '11, páginas 15–24, New York, NY, USA. ACM. ISBN 978-1-4503-0699-7. doi: 10.1145/1996109.1996114. URL http://dx.doi.org/10.1145/1996109.1996114. Citado na pág. 32

Wang *et al.* (2013) D. Wang, T. Bednarz, Y. Arzhaeva, J. Taylor, P. Szul, Chen, N. Burdett, A. Khassapov e T. Gureyev. Cloud Computing for High Performance Image Analysis on a National Infrastructure. Em *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, páginas 172–173. IEEE. ISBN 978-1-4673-6465-2. doi: 10.1109/ccgrid.2013.32. URL http://dx.doi.org/10.1109/ccgrid.2013.32. Citado na pág. 6

Weske (2012) Mathias Weske. Business process management concepts, languages, architectures, 2012. URL http://dx.doi.org/10.1007/978-3-642-28616-2. Citado na pág. 11, 14

White (2009) Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, original ed. ISBN 0596521979. Citado na pág. 28

Workflow () SHIWA Workflow. Sharing interoperable workflows for large-scale scientific simulations on available dcis. http://www.shiwa-workflow.eu. Último acesso em 30/06/2016. Citado na pág. 7

Yu e Buyya (2005) Jia Yu e Rajkumar Buyya. A taxonomy of scientific workflow systems for grid computing. *SIGMOD Rec.*, 34(3):44–49. ISSN 0163-5808. doi: 10.1145/1084805.1084814. URL http://dx.doi.org/10.1145/1084805.1084814. Citado na pág. 10, 13, 25

Zhang *et al.* (2009) Jinyan Zhang, Xudong Lu, Hongchao Nie, Zhengxing Huang e W. M. P. van der Aalst. Radiology Information System: a Workflow-Based Approach. *International Journal of Computer Assisted Radiology and Surgery*, 4(5):509–516. doi: 10.1007/s11548-009-0362-6. URL http://dx.doi.org/10.1007/s11548-009-0362-6. Citado na pág. 11

Zhang *et al.* (2001) Y. Zhang, M. Brady e S. Smith. Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. *IEEE transactions on medical imaging*, 20(1):45–57. ISSN 0278-0062. doi: 10.1109/42.906424. URL http://people.cs.uu.nl/robby/teaching/2010_pr/ch09.pdf. Citado na pág. 42

Zisman (1977) Michael D. Zisman. *Representation, Specification and Automation of Office Procedures*. Tese de Doutorado, University of Pennsylvania. Citado na pág. 9