APLICAÇÃO DE WORKFLOWS CIENTÍFICOS A PROJETOS DE SISTEMAS DE PRODUÇÃO DE PETRÓLEO OFFSHORE

Leandro Ouriques Mendes de Carvalho

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Civil, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Civil.

Orientadores: Breno Pinheiro Jacob
Marta Lima de Queirós Mattoso

Rio de Janeiro
Junho de 2009

APLICAÇÃO DE WORKFLOWS CIENTÍFICOS A PROJETOS
DE SISTEMAS DE PRODUÇÃO DE PETRÓLEO OFFSHORE

Leandro Ouriques Mendes de Carvalho

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA CIVIL.

Aprovada por:

_____
Prof. Breno Pinheiro Jacob, D.Sc.

_____
Prof [a]. Beatriz de Souza Leite Pires de Lima, D. Sc.

_____
Dr. Fabrício Nogueira Corrêa, D. Sc.

_____
Dr [a]. Stael Ferreira Senra, D. Sc.

RIO DE JANEIRO, RJ - BRASIL
JUNHO DE 2009

Carvalho, Leandro Ouriques Mendes de

Aplicação de Workflows Científicos a Projetos de Sistemas de Produção de Petróleo *Offshore* / Leandro Ouriques Mendes de Carvalho. – Rio de Janeiro: UFRJ/COPPE, 2009.

XV, 141 p.: il; 29,7 cm.

Orientadores: Breno Pinheiro Jacob

Marta Lima de Queirós Mattoso

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia Civil, 2009.

Referencias Bibliográficas: p. 122-132.

1. Workflows Científicos. 2. Proveniência. 3. Sistemas *Offshore*. I. Jacob, Breno Pinheiro, et al. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Civil. III. Titulo.

Aos meus pais, Manoel e Denise;
família e amigos ...

AGRADECIMENTOS

Aos meus pais, avós, irmão e família pelo respeito e pelo amor; e por sempre terem me incentivado a estudar, a aprender e a crescer como pessoa e como profissional.

Ao Departamento de Engenharia Eletrônica e de Computação da Universidade Federal do Rio de Janeiro por oferecerem uma excelente formação de graduação, cujos conhecimentos adquiridos contribuíram para solucionar os desafios enfrentados durante o mestrado.

Aos meus orientadores Breno Pinheiros Jacob e Marta Lima de Queirós Mattoso pela paciência, orientação e motivação; e aos demais professores do Programa de Engenharia da Civil da COPPE, pelos conhecimentos durante o mestrado.

Ao professor Luiz Landau, coordenador do Laboratório de Métodos Computacionais em Engenharia (LAMCE), pelos recursos disponibilizados e pelo apoio recebido durante a minha graduação e o mestrado enquanto eu trabalhava no laboratório.

Ao meu amigo Luís Fernando Nunes Mello e aos demais amigos do LAMCE pelos bons momentos que passei em quase cinco anos no laboratório.

Ao meu amigo Fabrício pela fundamental ajuda e motivação durante a dissertação me ensinando os conhecimentos em Engenharia *Offshore*.

Aos amigos do Laboratório de Métodos Computacionais em sistema *Offshore* (LAMCSO) pela recepção e ajuda no desenvolvimento dessa dissertação.

Aos amigos que estudaram comigo na graduação em Engenharia Eletrônica e Computação na UFRJ, pelo companheirismo, pela amizade e pelos bons momentos de diversão desde 2000.

Aos meus amigos do Centro de Análise de Sistemas Navais (CASNAV) da Marinha do Brasil que sempre contribuíram para meu crescimento profissional, particularmente, em banco de dados, análise de sistemas, e atualmente em georeferenciamento.

Finalmente, a todos aqueles que, direta ou indiretamente, contribuíram para a elaboração deste trabalho.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

APLICAÇÃO DE WORKFLOWS CIENTÍFICOS A PROJETOS DE SISTEMAS DE PRODUÇÃO DE PETRÓLEO OFFSHORE

Leandro Ouriques Mendes de Carvalho

Junho/2009

Orientadores: Breno Pinheiro Jacob
Marta Lima de Queirós Mattoso

Programa: Engenharia Civil

A exploração de petróleo em águas cada vez mais profundas aumenta os desafios da produção. As estruturas de suporte a plataformas *offshore* sofrem ação de forças naturais como ventos, ondas e correntes. Métodos numéricos são empregados para efetuar análises para o projeto e estudo do comportamento dessas estruturas, por meio do uso encadeado de diversos sistemas computacionais. Nesse contexto, esta dissertação propõe utilizar sistemas de gerência de *workflows* científicos (SGWfC) em metodologias de projeto de estruturas offshore, capazes de representar e organizar o uso das ferramentas computacionais utilizadas nas análises, compartilhar e reaproveitar análises bem-sucedidas, e prover interoperabilidade entre dados e as ferramentas.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

SCIENTIFIC WORKFLOWS APLICATION IN PROJECTS OF SYSTEMS FOR OFFSHORE OIL PRODUCTION

Leandro Ouriques Mendes de Carvalho

June/2009

Advisors: Breno Pinheiro Jacob
Marta Lima de Queirós Mattoso

Department: Civil Engineering

Oil exploration in deeper waters increases the challenges of production. Structures that support offshore platforms suffer the action of environmental loadings such as winds, waves and sea currents. Numerical methods are employed to perform analyses needed for the design of such structures, through a sequence of different computational systems. In this context, this work employs scientific workflow management systems (SWfMSs) in design methodologies of offshore structures. These systems are able to represent and organize the tools used in the analysis and the data involved and produced by the processes, allowing to share and reuse the successful analyses, and providing interoperability between data and scientific tools.

SUMÁRIO

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

GLOSSÁRIO

Bioinformática – É o estudo da aplicação de técnicas computacionais e matemáticas à geração e gerenciamento de informações para Biologia. Ela combina conhecimentos de química, física, biologia, ciência da computação, informática, matemática e estatística para processar dados biológicos ou biomédicos.

Canvas – Área de desenho onde um modelo ou uma arquitetura podem ser representados graficamente.

Grafo - Em matemática e ciência da computação, grafo é o objeto básico da teoria dos grafos que estuda as relações entre os objetos de um determinado conjunto. Tipicamente é representado como um conjunto de pontos (vértices) ligados por retas (arestas).

Metadados - São dados sobre outros dados. Um item de um metadado pode dizer do que se trata aquele dado, geralmente uma informação inteligível por um computador. Os metadados facilitam o entendimento dos relacionamentos e a utilidade das informações dos dados.

Ontologia – Em ciência da computação, uma ontologia é um modelo de dados que representa um conjunto de conceitos dentro de um domínio e os relacionamentos entre eles. É utilizada para realizar inferência sobre os objetos do domínio.

Token – Em um *workflow*, o token é um recurso que fornece a um elemento (ator) uma permissão exclusiva (ou não) de execução de uma atividade durante um determinado período de tempo, desta forma, permite controlar a transmissão dos dados.

Tupla – Corresponde a cada linha das tabelas do modelo relacional em um banco de dados. Representa um registro ou conjunto de informações relacionadas.

Capítulo 1 - Introdução

1.1 Contexto

1.1.1 Histórico da Exploração e Produção de Petróleo no Mar

O consumo mundial de petróleo é da ordem de mais de 80 milhões de barris de petróleo por dia. Para atender a essa demanda por combustíveis fósseis, as empresas petrolíferas vasculham constantemente o planeta em busca de novas reservas. Através do trabalho e das pesquisas realizadas nas últimas décadas o Brasil se tornou um dos principais países produtores e exportadores de petróleo no mundo.

A primeira descoberta de petróleo no país foi feita em 1939 no subúrbio de Lobato, em Salvador na Bahia. Já a produção começou a ser realizada na década de 40 no município de Candeias, localizado também na Bahia. Em 1953, no governo de Getúlio Vargas, foi sancionada uma lei que criou a Petrobras – Petróleo Brasil S/A, a empresa estatal brasileira de exploração e produção de petróleo [1].

A primeira descoberta de petróleo no mar foi feita na década de 60 no campo de Guaricema em Sergipe. Atualmente, as maiores reservas de petróleo e gás do país estão localizadas no mar, nas bacias sedimentares da plataforma continental. A principal província petrolífera do país é a Bacia de Campos situada na costa norte do Estado do Rio de Janeiro. Ela possui aproximadamente 100 mil quilômetros quadrados se estendendo do sul do Estado do Espírito Santo até seu limite com a Bacia de Santos. Hoje ela é responsável por cerca de 85% de todo o petróleo extraído do território brasileiro.

A produção na Bacia de Campos foi anunciada pela Petrobras na década de 70. O poço pioneiro foi perfurado a uma lâmina d'água de 100 metros onde hoje está situado o Campo de Garoupa. Entretanto, a exploração comercial só teve início em 1977 no Campo de Enchova utilizando uma plataforma flutuante.

Na década de 80, houve a descoberta dos Campos de Albacora e Marlim, os primeiros campos gigantes dessa bacia. Na década de 90, a Petrobras inicia a produção em águas profundas. Em 1994, a exploração atinge uma lâmina d'água superior a 1000 metros no Campo de Marlim. Em 1996, também na Bacia de Campos, é descoberto o campo gigante de Roncador. No ano seguinte o país supera a marca de produção de um

milhão de barris diários de petróleo. Nos anos seguintes a Petrobras foi se tornando a pioneira em exploração de petróleo em águas profundas ultrapassando lâminas d'água de 2000 metros. A auto-suficiência foi em 2006, isto quer dizer que as reservas nacionais representam um volume igual ou superior ao que pode ser refinado em nossas refinarias, de modo a atender a demanda do mercado interno.

Em 2007 foi descoberto o Campo Tupi na bacia de Santos, a 250 quilômetros da costa do Estado do Rio de Janeiro. Suas reservas, na camada pré-sal, são estimadas entre 5 e 8 bilhões de barris de petróleo do tipo alta qualidade, ou seja petróleo leve, além de gás natural. Estas anunciadas representam mais do dobro das reservas do Campo de Roncador, que contém aproximadamente 3 bilhões de barris, porém de petróleo pesado, que possui menor valor comercial. Esta era, até então, a maior descoberta de petróleo no Brasil

A Figura 1.1 [2] apresenta o mapa atual da Bacia de Campos, contendo a delimitação das áreas dos campos de exploração e produção.

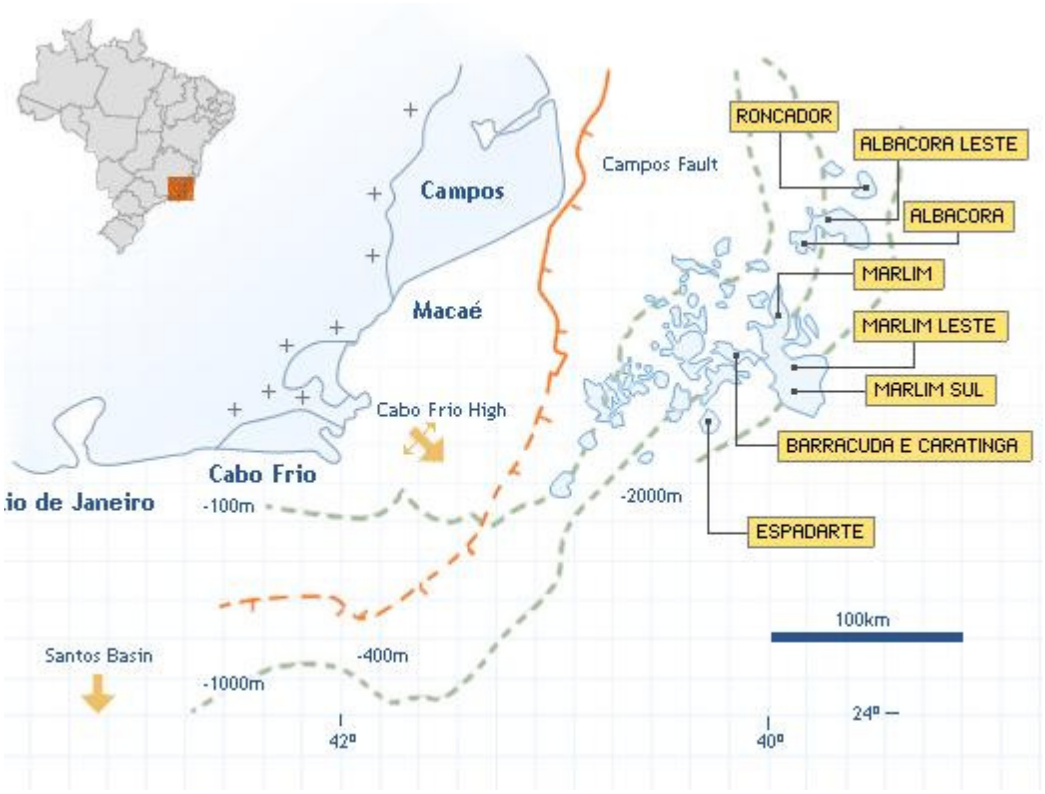


Figura 1.1: Bacia de Campos

A Figura 1.2 [3] apresenta o histórico da exploração de petróleo no mar pela Petrobras mostrando a evolução da profundidade alcançada.

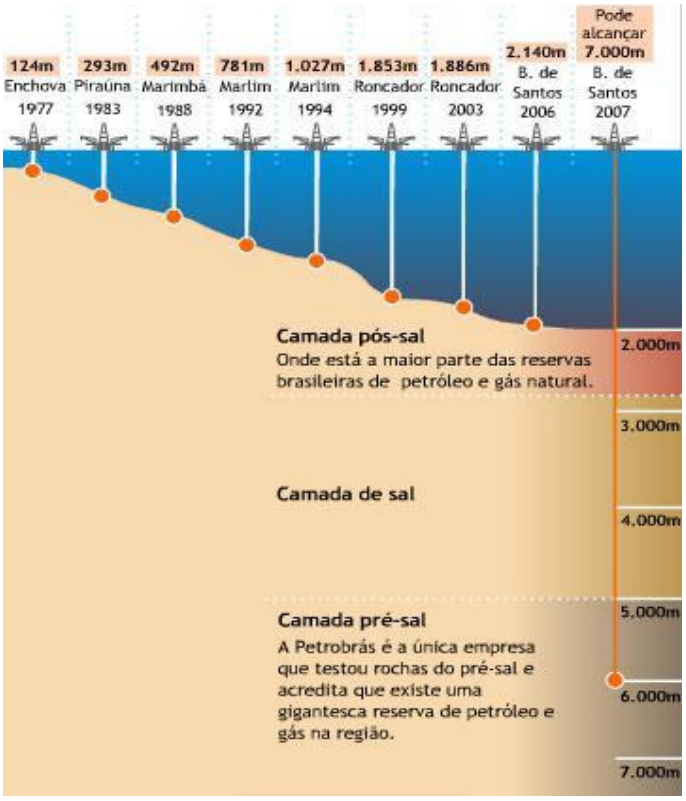


Figura 1.2: Evolução da exploração de petróleo no mar

Uma vez que as maiores reservas de petróleo do país estão situadas no fundo do mar, a Petrobras tem investido muitos recursos no desenvolvimento de novas tecnologias que a elevaram à excelência mundial na exploração e produção em águas profundas e ultra-profundas. Faz-se necessário montar uma infra-estrutura cada vez mais complexa e desafiadora para ter acesso a esses poços.

A exploração e produção de petróleo constituem um processo de pesquisa, localização, identificação, desenvolvimento, produção e incorporação de reservas de óleo e gás natural. A produção contém os processos de perfuração nas profundezas do mar e de transporte do petróleo à superfície.

Esses processos são realizados pelas plataformas e *risers*, que são os dutos de perfuração e produção que ligam as plataformas, na superfície, ao solo marinho. As empresas de petróleo investem muito capital em pesquisas para o desenvolvimento dessas estruturas. As plataformas modernas são estruturas gigantescas. Algumas são como cidades flutuantes, que empregam e abrigam centenas de pessoas. Uma plataforma pode conter em torno de 40 a 50 *risers*. A Figura 1.3 [4] mostra uma plataforma da Petrobras.

Figura 1.3: Plataforma de exploração e produção de petróleo

A instalação desse sistema integrado, plataforma juntamente com os *risers*, tem um custo muito elevado e demanda muito tempo. O sistema de produção exige um funcionamento perfeito. Há conscientização ecológica sustentável para que o processo que envolve toneladas de equipamentos em meio ao mar bravio, não represente um perigo ao meio ambiente marinho.

Se um *riser* apresentar problemas, a produção diminuiria, o que ocasionaria um prejuízo em função do tempo em que o poço está produzindo abaixo da sua capacidade. Um problema na integridade dessas estruturas pode causar acidentes muito graves, o que representaria um risco à vida dos trabalhadores alocados nas plataformas. Vazamentos de óleo representariam desperdício, consequentemente, outros prejuízos; e, certamente, são prejudiciais ao ecossistema marinho. A Figura 1.4 [5] ilustra a complexidade de um sistema de exploração e produção de petróleo em águas profundas.

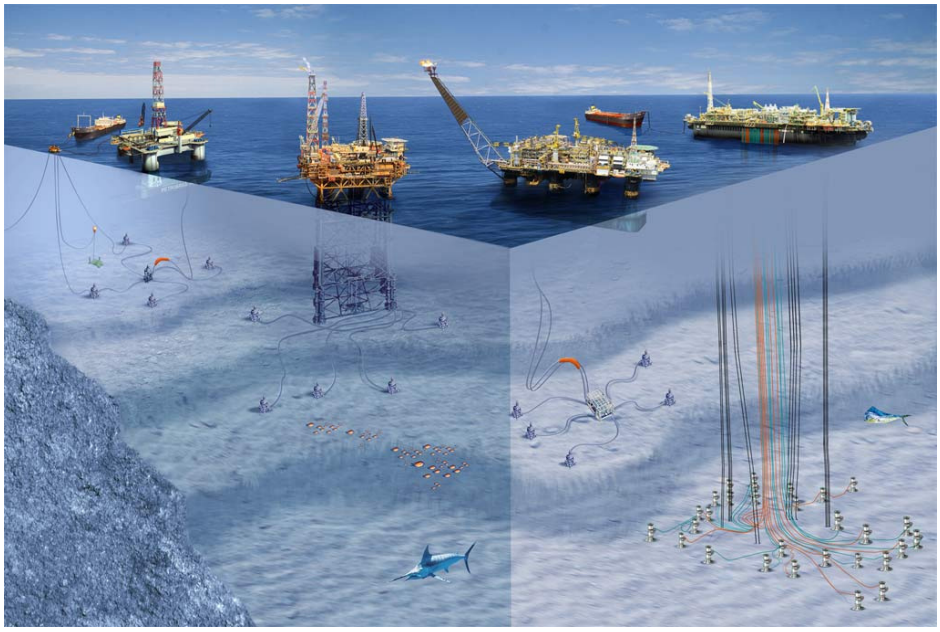


Figura 1.4: Sistemas de exploração e produção de petróleo

1.1.2 Atividades de Pesquisa e Desenvolvimento

A Petrobras mantém forte relacionamento com as comunidades científica e tecnológica e com empresas de projeto no país para desenvolver as tecnologias envolvidas nesse processo que asseguram seu funcionamento, confiabilidade e integridade. Uma de suas mais expressivas parcerias atuais com o meio acadêmico é a Rede Galileu que iniciou suas atividades com a participação de 76 instituições de pesquisa em 17 estados brasileiros. Dessa rede participam pesquisadores da Universidade Federal do Rio de Janeiro (UFRJ), Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ), Universidade de São Paulo (USP), dentre outras.

No total, são cerca de 700 pesquisadores subdivididos em 38 redes, cada uma com um tema e com a participação de no mínimo cinco instituições. Entre os principais objetivos dessas redes e do Cenpes (Centro de Pesquisa da Petrobras) estão aumentar cada vez mais a produção de petróleo e de gás natural, refinar o máximo possível de óleo, além de desenvolver novas fontes de energia alternativas, como biomassa, biodiesel, álcool, biogás e hidrogênio.

Para atingir esses objetivos, é necessário realizar estudos para desenvolver tecnologias para a prospecção de petróleo no fundo do mar, dutos para transporte de óleo e de combustíveis, além de estudos geológicos, produção de *softwares* específicos para o setor e pesquisa em energia alternativa e renovável. A rede será responsável por

elaborar projetos relacionados ao desenvolvimento de metodologias de cálculo estrutural, de experimentos e qualificação de dutos, conectores e *risers.* É nesse cenário de desenvolvimento de sistemas computacionais orientados à exploração e produção de petróleo e gás que esta dissertação está inserida, em parceria entre a Petrobras e a COPPE-UFRJ,

1.2 Motivação

Engenheiros que atuam em aplicações industriais de petróleo e gás utilizam um conjunto de ferramentas computacionais de simulação e visualização científica, muitas delas desenvolvidas por pesquisadores da COPPE, incluindo o LAMCSO (Laboratório de Métodos Computacionais em Sistemas *Offshore*). Tais ferramentas modelam e simulam o comportamento das plataformas de exploração e produção (E&P) de petróleo e já vêm sendo utilizados em problemas reais de simulação para as plataformas da Petrobras. Estão em constante atualização, cujas melhorias são sugeridas pelos pesquisadores para melhorar a qualidade dos projetos.

Dentre as aplicações dessas ferramentas computacionais, menciona-se o projeto dos *risers*. Esta dissertação irá tratar, especificamente, de um aspecto crítico para os projetos de plataformas: a estimativa da vida útil dos *risers*. A estimativa da vida útil é um problema de engenharia, que é solucionado através do uso de uma sequência de programas computacionais baseados em métodos numéricos. Nessa sequência é necessário modelar e analisar o conjunto completo composto pela plataforma, seu sistema de ancoragem e risers, bem como modelar cada *riser* de interesse, tudo isso sob a ação de diversas condições ambientais.

A própria experiência da equipe do LAMCSO em atividades de projeto de *risers* já havia identificado as dificuldades envolvidas na forma atual de controle do fluxo do processo entre os diferentes programas utilizados, efetuada de forma manual e muito trabalhosa; e na quantidade de informações processadas, envolvendo uma enorme gama de dados de entrada e de saída. Outro aspecto crítico é o controle dessas informações para garantir que os resultados das análises sejam gerenciados e rastreados de maneira mais efetiva, garantindo a proveniência das informações. No contexto desta dissertação, a proveniência pode ser definida como a documentação da origem destas informações envolvidas nos experimentos científicos.

Estes fatos motivaram o interesse em utilizar os conceitos de *workflows* científicos para modelar e documentar as seqüências de análises envolvidas na avaliação da fadiga de *riser*s. Os *workflows* científicos têm sido aplicados em diversas áreas de pesquisas acadêmicas, principalmente na bioinformática [6, 7], astronomia, química e física. Os *workflows* são implementados em sistemas de gerência de workflow científico (SGWfCs), que são ferramentas computacionais responsáveis pela gerência do workflow, ou seja, dos processos computacionais e das ferramentas utilizadas nos experimentos, bem como dos dados envolvidos e produzidos pelos processos.

1.3 Objetivos

Esta dissertação tem como objetivo utilizar um SGWfC em metodologias de projeto de estruturas offshore, capazes de representar e organizar o uso das ferramentas computacionais utilizadas nas análises, compartilhar e reaproveitar análises bem-sucedidas, e prover interoperabilidade entre dados e as ferramentas utilizadas pela comunidade científica.

Considerando os aspectos inovadores dos SGWfC e as dificuldades em sua utilização, as seguintes metas podem ser destacadas para compor os objetivos da dissertação:

- Levantamento dos processos, dados e recursos a serem gerenciados pelo *workflow* científico;
- Definição do software de SGWfC mais adequado para gerenciar o *workflow* modelado para realizar a análise de fadiga dos *risers* dos sistemas *offshore*;
- Implementação dos *workflows* no SGWfC;
- Criação do modelo para a proveniência e de uma base de dados para armazenar as informações relevantes na execução dos *workflows*.
- Desenvolver um sistema para prover acesso aos engenheiros que realizaram as análises aos dados armazenados de proveniência.

1.4 Metodologia

Primeiramente foi necessário fazer um estudo sobre a Engenharia *Offshore*. Seria importante conhecer suas áreas de atuação, quais os problemas de engenharia que ela se propõe a solucionar, e em quais atividades de pesquisa os conceitos de *workflows* científicos poderiam ser aplicados.

Foi definido que os conceitos de *workflows* científicos poderiam ser aplicados num estudo de caso no processo para calcular a estimativa da vida útil de um *riser*. Então, foi feito o levantamento dos requisitos, dos dados e dos recursos a serem gerenciados pelo SGWfC. Era necessário que muitas perguntas fossem respondidas para compreender como é o trabalho dos engenheiros nesse processo.

- Quais etapas do processo são obrigatórias?
- Quais sistemas computacionais são utilizados?
- Qual é a função de cada programa?
- Há pré-condições para executá-los?
- Há dependência entre eles?
- Existe a possibilidade para que o processamento seja paralisado?

À medida que a Engenharia *Offshore* estava sendo conhecida, também eram feitos estudos dos *workflows* científicos, ressaltando quais eram as suas diferenças em relação aos *workflows* tradicionais, ou seja, os de negócio. Também foi feita uma pesquisa para conhecer os softwares de SGWfCs mais relevantes para definir o qual melhor atenderia às necessidades dos experimentos. Quando foi escolhida a ferramenta de SGWfC para montar o workflow, os esforços se concentraram na exploração dos seus recursos. A implementação dos *workflows* no SGWfC envolveu o conhecimento dos modos de execução da ferramenta, as estruturas de controle disponíveis e a grande quantidade de componentes para executar tarefas e outros programas.

O uso de *workflows* científicos ainda encontra-se num estágio inicial. Não há ainda metodologias que auxiliem a especificação de um workflow científico. A escolha de um SGWfC também não é trivial. Existem dezenas desses sistemas e todos estão em constante processo de evolução. O próprio uso de SGWfC, ou seja, como melhor explorar seus recursos e adequar às necessidades da aplicação, não é simples. Alguns recursos não se encontram disponíveis ou necessitam de ajustes não triviais. Um dos

recursos importantes é a gerência da proveniência dos *workflows* executados. Os conceitos envolvendo a gerência da proveniência serão apresentados no Capítulo 3 - Esse é um tópico recente e ainda incipiente em diversos SGWfCs, no entanto é fundamental para o registro e análise posterior do fluxo do workflow.

Como poderá ser observado adiante, durante a apresentação do workflow implementado no SGWfC, diante da diversidade de dados e controle no fluxo de projeto de fadiga, fica clara a relevância e aplicabilidade deste procedimento empregando um SGWfC.

O estudo da proveniência se constituiu na pesquisa de modelos e ferramentas para capturar, representar, armazenar e consultar os dados de proveniência referentes à definição e execução do workflow. Nessa análise, foi verificado que o apoio à proveniência fornecido pelos SGWfCs é bastante diverso e ainda incipiente em alguns deles. Além disso, foi constatado que algumas das perguntas que deveriam ser respondidas a partir dos dados de proveniência não seriam obtidas de forma simples. Sendo assim, foi realizada uma modelagem para representar os dados de proveniência e desenvolvido um sistema de armazenamento e consulta de proveniência associado à execução dos *workflows*.

Para concluir o trabalho, foram feitas as análises dos resultados obtidos e foi pensado num meio para que os engenheiros fossem capazes de acompanhar o andamento da execução do workflow e tivessem acesso ao histórico das análises concluídas. Então, foi desenvolvido um sistema web com uma interface rápida, amigável e intuitiva para eles realizarem consultas a esses dados, bem como registrar comentários sobre as análises.

O diagrama da Figura 1.5 resume as atividades realizadas nessa dissertação. Os retângulos claros representam as atividades de estudo, e os retângulos escuros representam as atividades de implementação.

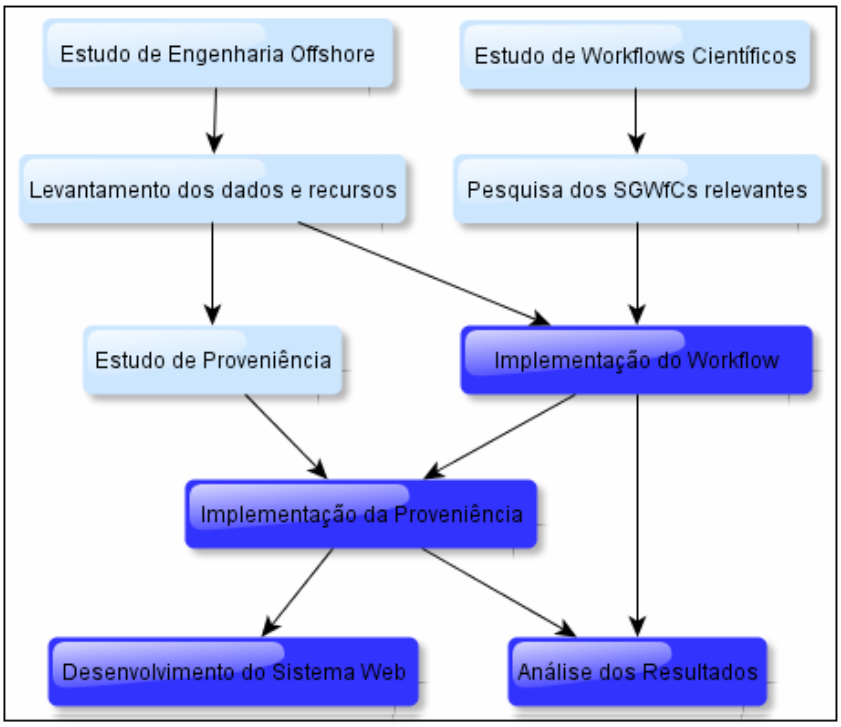


Figura 1.5: Metodologia do trabalho

1.5 Revisão Bibliográfica

Essa dissertação foi desenvolvida a partir da colaboração entre duas linhas de pesquisas de áreas acadêmicas distintas: sistemas *offshore,* da Engenharia Civil; e *workflows* científicos, da Engenharia de Sistema e Computação. A seguir será apresentada uma breve revisão bibliográfica dos trabalhos que têm sido realizados nessas linhas de pesquisas.

1.5.1 Projetos de Sistemas *Offshore*

As pesquisas acadêmicas em projetos de sistema flutuantes *offshore* têm como objetivo avaliar as metodologias atuais, assim como desenvolver novas metodologias; desenvolver sistemas computacionais numéricos para implementar essas metodologias; e pesquisar novos algoritmos para otimizar esses sistemas com o objetivo de reduzir o tempo de processamento. A COPPE e a Petrobras possuem uma estreita parceria de sucesso nessas pesquisas. Dentre as diversas dissertações e teses defendidas como fruto desta parceria, destacam-se no contexto de metodologias e ferramentas de simulação acoplada (as quais serão descritas mais adiante no Capítulo 2), as apresentada a seguir.

No PEC/COPPE, os primeiros trabalhos relacionados a este tema foram a Tese de Doutorado de Senra [8] e a Dissertação de Mestrado de Correa [9]. A Dissertação de

Mestrado de Correa [9] apresentou resultados de estudos paramétricos comparando o procedimento tradicional de projeto baseado em análises desacopladas com procedimentos mais recentes englobando análises acopladas. Já a Tese de Doutorado de Senra [8], intitulada "Metodologias de Análise e Projeto Integrado de Sistemas Flutuantes para Explotação de Petróleo *Offshore*", teve como objetivo sugerir e estudar metodologias de análise acoplada, desacoplada e híbrida para o projeto de sistemas de produção de petróleo em águas profundas, focando também na análise de fadiga de *risers*. Procurou identificar as mais adequadas para cada fase do projeto, levando em conta o nível de precisão requerido em cada fase e, além disso, o comprometimento entre custo e benefício para o projeto dos *risers* e sistema de ancoragem integrado.

Em 2006, foi apresentada a dissertação de mestrado "Avaliação de Metodologias de Análise de Unidades Estacionárias de Produção de Petróleo *Offshore*" [10]. Este trabalho apresenta a execução e a validação de análises dinâmicas acopladas de sistemas *offshore*, considerando a interação entre um corpo rígido, representando a unidade flutuante, e as linhas de ancoragem e *risers*, modeladas por elementos finitos.

Em 2007, foi apresentada a dissertação de mestrado "Implementação e Avaliação de uma Metodologia Fortemente Acoplada para Análise de Sistemas Flutuantes *Offshore*" [11]. O objetivo deste trabalho é comparar duas formulações para o modelo acoplado: fracamente acoplada e fortemente acoplada. Em ambas as implementações, o comportamento hidrodinâmico/estrutural das linhas é representado por modelos de Elementos Finitos acoplados com os 6 graus de liberdade do modelo de corpo rígido que representa o casco da plataforma. A diferença entre as metodologias está na maneira como as equações de movimento destes graus de liberdade são solucionadas, acoplando-as ou não na matriz efetiva do sistema de equações de movimento das linhas.

Em 2008, foi apresentada a tese de doutorado "Ferramentas Computacionais para Análise Acoplada de Sistemas *Offshore*" [12]. O objetivo deste trabalho é apresentar uma nova estratégia numérica para reduzir o custo computacional e, ao mesmo tempo, melhorar a precisão de análises acopladas. Também apresenta uma inovadora ferramenta numérica para análise de vibrações livres de sistemas flutuantes acoplados.

Em 2009, foi apresentada a dissertação de mestrado "Aplicação de Metodologias de Projeto Integrado de Sistemas de Ancoragem e *Risers* na Explotação de Petróleo *Offshore*" [13]. Este trabalho estabelece um procedimento de projeto integrado baseado no conceito de desenvolvimento da "Zona Segura de Operação" do conjunto dos *risers* (*SAFOP*), onde os critérios de utilização dos *risers* são utilizados como um critério adicional no projeto do sistema de ancoragem.

1.5.2 *Workflows* Científicos

As pesquisas acadêmicas em projetos de *workflows* científicos têm como objetivo utilizar procedimentos computacionais com o intuito de lidar com o aumento constante dos volumes de dados, e manipulações necessárias para garantir produtividade e qualidade na condução dos experimentos científicos. Os primeiros SGWfC surgiram a partir de 2000 e apenas a partir de 2005 tornaram-se ferramentas de uso mais sistemático. As dissertações e teses desenvolvidas nessa área propõem metodologias para modelagem de *workflows*, modelos para gerenciar a proveniência dos dados e implementam experimentos nos sistemas de gerência de *workflows* científicos, etc. A maioria dos trabalhos apresentados recentemente no Brasil desenvolve *workflows* em bioinformática. Dentre diversos trabalhos apresentados, pode-se destacar o seguinte histórico que contribuiu como referência de pesquisa para esta dissertação.

Em 2003, foi apresentada a tese de doutorado "Gerência de Recursos Científicos: Apoiando a Realização de Experimentos In Silico" [14]. Este trabalho ressalta importância da gerência de recursos científicos, isto é, modelos, algoritmos, programas, dados e *workflows*, envolvidos nos experimentos científicos. Apresenta uma arquitetura que utiliza serviços Web junto a SGWfC para compartilhar e reutilizar recursos para estes experimentos.

Em 2004, foi apresentada a dissertação de mestrado "O Ambiente 10+C para Definição e Execução de *Workflows* In Silico através de Serviços Web" [15]. Este trabalho propõe um sistema cuja arquitetura visa facilitar o cadastro de programas e dados a serem utilizados na definição de um workflow científico. O trabalho foi uma das primeiras iniciativas na representação e consulta a dados que representam a proveniência de um *workflow* científico. Os testes foram realizados com *workflows* de bioinformática.

Em 2007, foi apresentada a tese de doutorado “Gerenciamento de *Workflows* Científicos em Bioinformática” [16]. Este trabalho propõe uma infra-estrutura que permite projetar, re-usar, anotar, validar, compartilhar e documentar experimentos de bioinformática. Ela se beneficia do uso de ontologias para permitir a especificação e anotação de workflows de bioinformática e para servir como base aos mecanismos de rastreabilidade.

Também em 2007, foi apresentada uma dissertação de mestrado utilizando o SGWfC Kepler para apoiar um workflow de bioinformática, intitulada “Um ambiente para gerenciamento e execução de experimentos de expressão gênica com microarranjos” [17]. Este trabalho explorou recursos do pacote de estatísticas R, embutido no Kepler e também recursos de sistemas de gerência de bancos de dados.

Em 2008, foi apresentada a dissertação de mestrado “MiningFlow: um sistema de Workflow para apoiar o Processo de Descoberta do Conhecimento em Textos” [18]. Este trabalho propõe um sistema de definição de *workflows* baseado em ontologias e permite também representar e armazenar a proveniência. Os *workflows* científicos estão sendo utilizados em um problema de mineração de textos. Os *workflows* definidos sob o MiningFlow podem ser executados automaticamente pelo SGWfC Kepler ou Taverna.

Em 2008, foi apresentada a dissertação de mestrado “Gerência de Dados Genômicos em Sistemas de *Workflows* de Bioinformática” [19]. Este trabalho propõe uma arquitetura para auxiliar os SGWfC no gerenciamento de dados genômicos na execução de *workflows* em bioinformática, visando o apoio à proveniência. O sistema Kepler foi utilizado para gerenciar os *workflows*.

Em 2009, foi apresentada a dissertação de mestrado “Gerência de Distribuição na Execução de *Workflows* Científicos em Bioinformática” [20]. Este trabalho apresenta uma arquitetura para a gerência de *workflows* de bioinformática em ambientes distribuídos, tendo como funcionalidades a definição e o controle da execução remota dos *workflows* em clusters de PC. O desenvolvimento também foi realizado com o Kepler.

1.6 Organização do Texto

Este trabalho está organizado da seguinte maneira. Este capítulo 1 apresentou a introdução da dissertação, o contexto da exploração e produção de petróleo e gás que é onde está inserida, a motivação e os objetivos que levaram a desenvolvê-la e apresentou o histórico dos trabalhos acadêmicos relacionados às áreas desta pesquisa.

O capítulo 2 apresenta os tipos de estruturas utilizadas na exploração *offshore* de petróleo, as metodologias e os conceitos de engenharia utilizados em projetos de modelagem dessas estruturas; e a descrição dos sistemas computacionais utilizados pelos engenheiros para modelá-las.

O capítulo 3 apresenta os conceitos dos *workflows* científicos; ressalta a importância da proveniência dos dados em experimentos científicos e apresenta as linhas de pesquisas que estão sendo feitas a fim de apresentar modelos para implementar a proveniência; apresenta a abordagem utilizada para modelar o workflow; descreve brevemente os sistemas gerenciadores de *workflows* mais utilizados no meio acadêmico; e apresenta uma descrição detalhada da ferramenta Kepler utilizada para modelar e executar os *workflows* desenvolvidos neste trabalho.

O capítulo 4 descreve as etapas da modelagem do workflow de análise de fadiga de *riser*, detalhando cada uma das suas atividades e apresenta os produtos resultantes dessa modelagem que contêm as especificações necessárias para a implementação do workflow.

O capítulo 5 descreve a implementação do workflow no sistema Kepler; apresenta as soluções adotadas para controlar a proveniência dos dados envolvidos no processo; e, finalmente, apresenta o sistema web que constituiu uma ferramenta de apoio aos experimentos científicos, onde os engenheiros podem consultar a proveniência armazenada no banco de dados; e acompanhar a execução do workflow, assim como, têm acesso ao histórico das análises realizadas.

O capítulo 6 apresenta um estudo de caso real de análise de fadiga que foi executada no workflow científico implementado no Kepler. Aborda todas as tarefas necessárias para a configuração do ambiente e execução do workflow científico de um problema de engenharia offshore, tendo como produto final a proveniência completa do projeto realizado. Finalmente, o capítulo 7 apresenta as conclusões desta dissertação assim como sugestões para trabalhos futuros.

Capítulo 2 - Metodologias de projeto de *riser*

2.1 Sistemas flutuantes e *risers*

Os Sistemas *Offshore* dedicados à produção de petróleo no mar compreendem plataformas de exploração, de produção e de armazenamento do óleo [11]. Atualmente, as atividades de produção *offshore* de petróleo são executadas em diferentes profundidades. Cada cenário exige a utilização de estruturas otimizadas para operar sob suas determinadas condições ambientais.

Inicialmente, a produção *offshore* de petróleo no Brasil era realizada em lâminas d'água com profundidades entre 100m e 500m; hoje estas são consideradas profundidades rasas. Eram utilizadas plataformas marítimas fixas, compostas por estruturas rígidas que podem ser constituídas de aço ou concreto. Em função da sua elevada rigidez, os efeitos dinâmicos submetidos às condições ambientais eram pouco significativos inicialmente em lâminas d'água rasas. O comportamento não-linear da estrutura também não é significativo, devido aos pequenos deslocamentos que ela sofre. A Figura 2.1 [21] ilustra uma plataforma fixa.

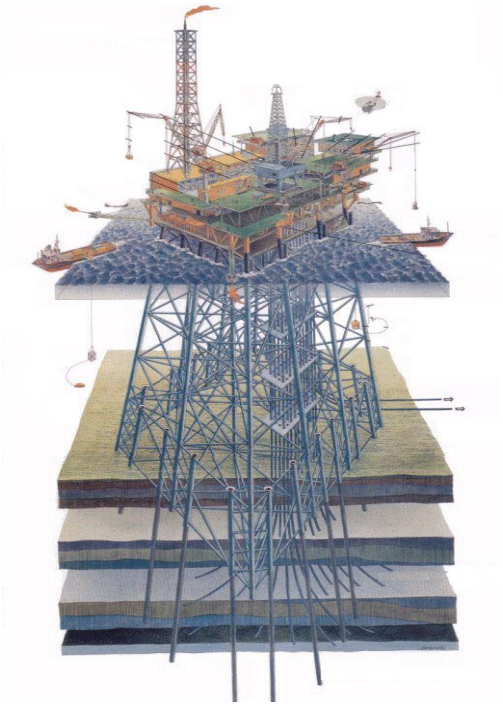


Figura 2.1: Plataforma Fixa

À medida que foram sendo descobertos novos reservatórios de petróleo em águas mais profundas com lâminas d'água entre 500m e 1000m, fazia-se necessário construir estruturas maiores e muito rígidas para suportar as condições ambientais mais extremas, o que se mostrou economicamente inviável. Então, foram concebidas novas estruturas para tornar realidade a exploração e produção de petróleo em águas profundas. Foram introduzidos os sistemas flutuantes ancorados no fundo do mar por meio de cabos de aço ou poliéster e/ou amarras (correntes de aço). Estes sistemas são descritos a seguir.

2.1.1 Tipos de unidades flutuantes

As unidades flutuantes, que possuem dutos cuja vida útil é de aproximadamente 20 a 30 anos, apresentam grandes deslocamentos como resposta às ações ambientais impostas [10]. Dentre os sistemas flutuantes destacam-se as seguintes embarcações: as semi-submersíveis, os navios FPSOs e TLPs.

As **plataformas semi-submersíveis** são estruturas flutuantes com um convés de grande porte, podendo ter várias colunas, braces e flutuadores (*pontoons*), os quais estão localizados na base das colunas. Tanto as colunas quanto os *pontoons* são compartimentos em tanques com a finalidade de oferecer estabilidade à estrutura em casos de avaria ou problemas no controle do lastro da unidade. As imagens da Figura 2.2 [10, 22] ilustram exemplos de plataformas semi-submersíveis.

Figura 2.2: Plataforma Semi-submersível

Os **FPSOs (*Floating Production, Storage and Offloading*)**, como seu próprio nome sugere, são unidades que além de realizar a atividade de produção, também possuem as condições que possibilitam o armazenamento e o escoamento do petróleo e gás. Em muitos casos são utilizados cascos de navios já existentes que foram convertidos e adaptados para atender esta nova finalidade.

A principal vantagem dos *FPSOs* é a sua viabilidade econômica, pois apresenta simplicidade de conversão ou construção, facilidade de instalação e re-locação, custo de investimento reduzido e curto prazo entre o projeto e o início da produção. Podem ser reaproveitados navios com cerca de 20 anos de uso e/ou com alto custo de operação para o transporte do petróleo. Eles apresentam algumas desvantagens operando como unidades estacionárias, isto devido ao seu grande porte que gera grande amplitude de movimentos sob tormenta, maior do que em outros tipos de sistemas flutuantes. Os sistemas de ancoragem são projetados para permitir um melhor alinhamento do navio com a condição ambiental dominante. A Figura 2.3 [23] é uma foto de um FPSO da Petrobras.

Figura 2.3: Navio FPSO

As **TLPs (*Tension Leg Platform*)** possuem estrutura semelhante às semi-submersíveis, diferindo-se basicamente em relação ao sistema de ancoragem. Esta unidade flutuante é mantida em posição por meio de tendões que estão submetidos a elevadas trações aplicadas em seu topo, provenientes da força resultante de restauração hidrostática (diferença entre peso e empuxo) em que a mesma foi projetada. Esta tração deve ser mantida ao longo de todo seu comprimento a fim de evitar a a flambagem dos tendões. A Figura 2.4 [11] ilustra uma TLP.

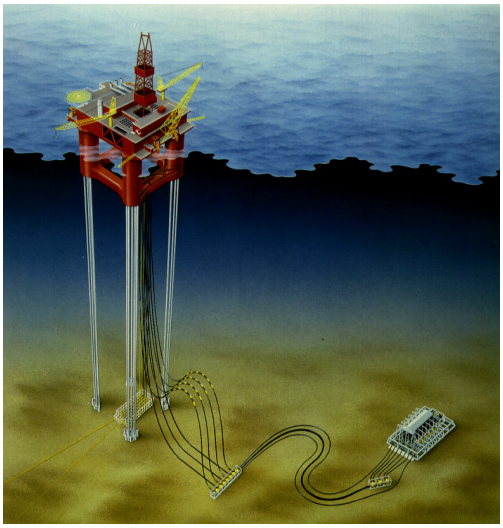


Figura 2.4: TLP

As **plataformas Spar** são unidades flutuantes formadas principalmente por uma estrutura cilíndrica vertical com grande calado. Elas são projetadas para que seu centro de gravidade fique abaixo do centro de flutuação, o que lhe fornece estabilidade. As imagens da Figura 2.5 [24] representam as plataformas Spar.

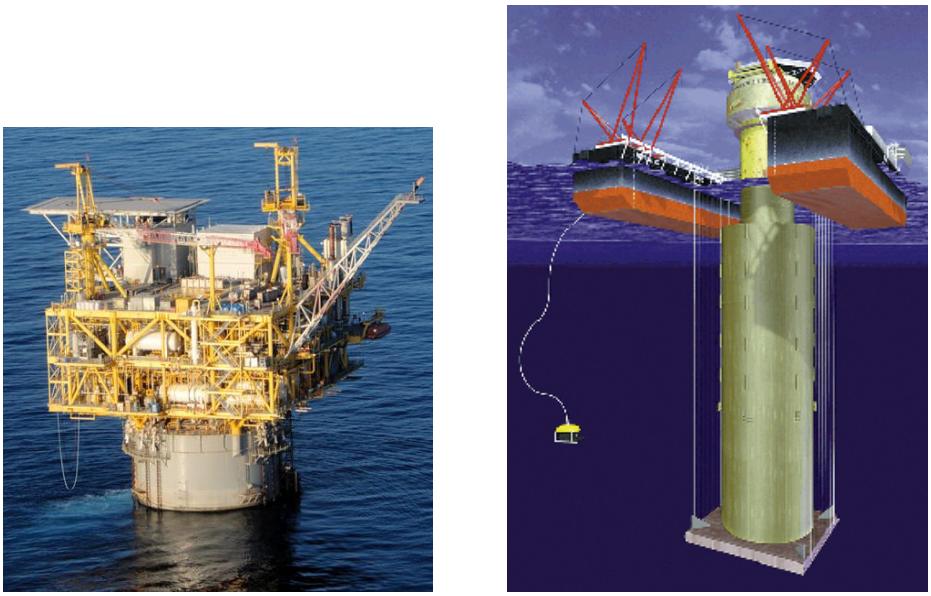


Figura 2.5: Plataforma Spar

2.1.2 Tipos de *risers*

Os *risers* são estruturas tubulares que fazem a ligação entre o poço produtor ou injetor de petróleo e a unidade flutuante, ou ainda entre a unidade flutuante e um duto de exportação (*pipeline*) [13]. Tendo como principal função o transporte dos fluidos (óleo, gás ou água). Existem ainda *risers* utilizados em operações de perfuração e

completação de poços. Conforme o tipo de material utilizado na fabricação do *riser,* este pode ser caracterizado como rígido ou flexível.

Os ***risers* rígidos** são dutos formados apenas por uma camada de metal. Podem ser constituídos de aço ou titânio. Eles contêm uma série de juntas de aproximadamente 12 metros de comprimento geralmente unidas por solda. Podem adotar a configuração de catenária livre, ou outras configurações com a utilização de flutuadores intermediários. A Figura 2.6 [10] ilustra um *riser* rígido empregado para atividades de perfuração.

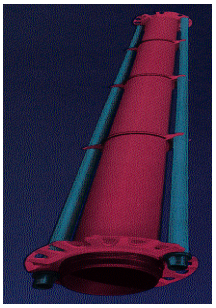


Figura 2.6: *Riser* Rígido

Os ***risers* flexíveis** são compostos pela superposição de camadas plásticas que garantem a integridade do fluido, e de camadas metálicas espiraladas responsáveis pela resistência à ação dos diversos carregamentos mecânicos. A Figura 2.7 [11] ilustra um *riser* flexível. Geralmente são dispostos em configurações em catenária, mas assim como nos rígidos, podem assumir diferentes configurações que possuem seções intermediárias com flutuadores, cujo empuxo alivia o peso suportado pelo sistema flutuante e, quando sob solicitação lateral, contribui com momentos restauradores.

Figura 2.7: *Riser* Flexível

A escolha de uma configuração é principalmente baseada em critérios econômicos e na localização dos poços. A Figura 2.8 ilustra as diferentes configurações assumidas pelos *risers*.

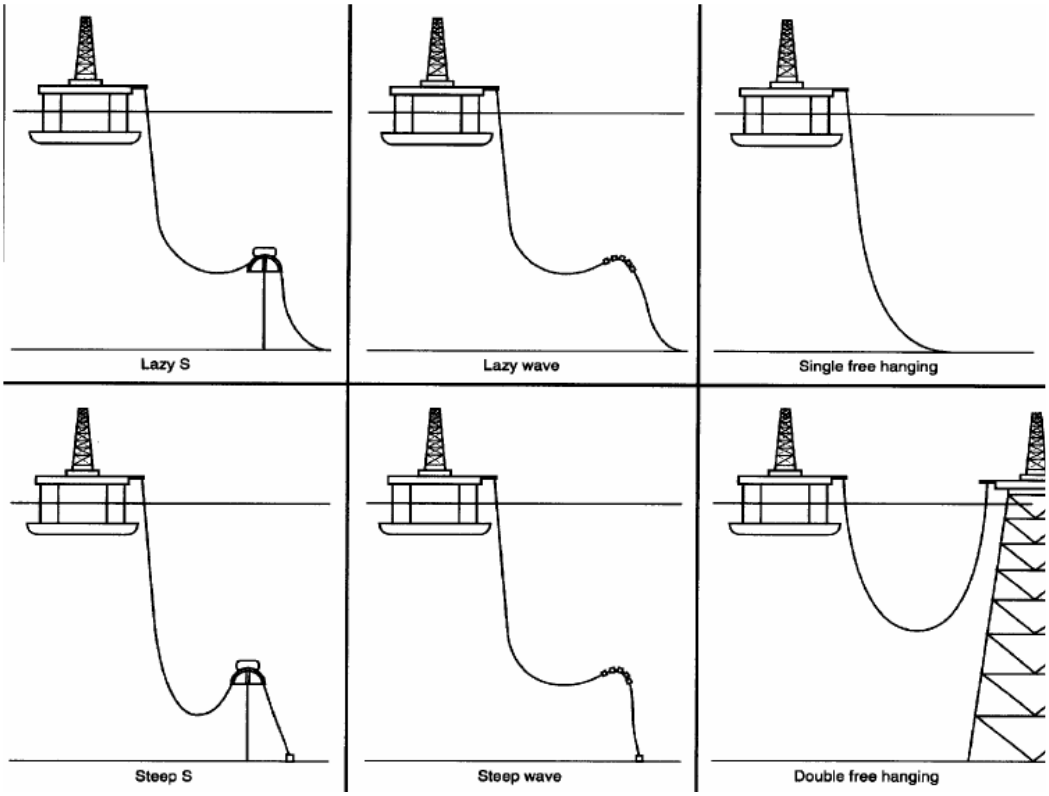


Figura 2.8: Configurações dos *Risers* Flexíveis.

Os ***risers* híbridos** constituem uma configuração de *risers* flexíveis associados a *risers* rígidos. Geralmente conecta-se um *riser* flexível em catenária livre à unidade flutuante. A extremidade inferior do *riser* flexível é conectada ao *riser* rígido, através de um dispositivo de flutuação ou bóia. O trecho flexível permite desacoplar os movimentos da unidade flutuante com o trecho rígido. A Figura 2.9 ilustra uma configuração híbrida de *risers*.

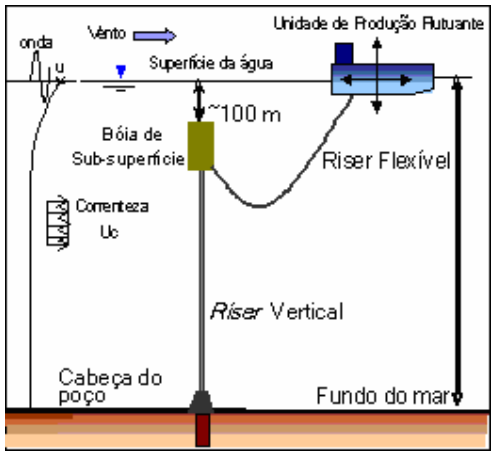


Figura 2.9: Configuração Híbrida de *Risers*

2.2 Metodologias de Análise

As metodologias de projetos de sistemas *offshore* acompanham a evolução das atividades de exploração e produção da indústria do petróleo [8]. A seguir são apresentadas as metodologias mais comumente utilizadas, suas condições de aplicações, e suas vantagens e desvantagens.

2.2.1 Metodologia Desacoplada

A metodologia de projeto que tradicionalmente se usava em plataformas flutuantes era baseada em análises desacopladas [8, 10, 13]. Os movimentos do casco e o comportamento estrutural dinâmico não linear das linhas de ancoragem e *risers* são analisados separadamente. Entretanto, essa metodologia caracteriza-se por não levar em consideração com rigor a influência do comportamento estrutural das linhas em termos de massa adicionada, rigidez, amortecimento estrutural e hidrodinâmico, e cargas hidrodinâmicas, junto à análise do comportamento global do sistema.

A análise é feita em duas etapas. Na primeira etapa, realiza-se uma análise hidrodinâmica para determinar os movimentos da unidade flutuante e conceder uma estimativa das trações das linhas de ancoragem, que nesta etapa são representadas de forma simplificada por coeficientes escalares de massa, rigidez e amortecimento. Na segunda etapa, os movimentos da unidade flutuante obtidos anteriormente são aplicados no topo de cada *riser*, agora representados por um modelo rigoroso de elementos finitos, para a avaliação de suas respostas estruturais.

Esta metodologia tem obtido resultados aceitáveis quando o projeto é destinado à exploração e produção em águas rasas e possui uma quantidade de *risers* pequena, de modo que, a sua contribuição estática e dinâmica no comportamento global não seja relevante. Contudo, as mesmas simplificações podem induzir erros graves na estimativa do comportamento de plataformas com grande número de *risers* e/ou localizadas em águas profundas [8]. Mesmo assim, essa metodologia vinha sendo largamente empregada em projetos *offshore* em virtude do tempo de processamento ser bem reduzido, até que trabalhos realizados recentemente [10, 11, 12, 13] indicaram o comprometimento destas simplificações em projetos atuais, sugerindo que simuladores baseados em formulações acopladas sejam utilizados.

2.2.2 Metodologia Acoplada

A metodologia de análise acoplada é a mais completa e adequada para ser aplicada em projetos de plataformas flutuantes situadas em cenários de exploração e produção de petróleo em águas profundas e com uma grande quantidade de linhas conectadas [8, 10, 13]. Ela é configurada totalmente em simuladores baseados em análises acopladas que incorporam em um único modelo computacional tridimensional, o modelo hidrodinâmico do casco da unidade flutuante acoplado ao modelo de elementos finitos das linhas de ancoragem e *risers*. Considerando as condições ambientais a que a plataforma provavelmente estará sujeita, são realizadas análises no domínio do tempo para considerar a interação não linear do comportamento hidrodinâmico do casco com o comportamento estrutural e hidrodinâmico das linhas de ancoragem e *risers*. Os resultados são fornecidos com alto nível de precisão e são mais confiáveis porque consideram participação das linhas de ancoragem e *risers*.

Entretanto, para fornecer a resposta estrutural detalhada dos *risers*, é necessário empregar malhas muito refinadas de elementos finitos. Este fato constitui uma desvantagem desta metodologia que é o elevado tempo de processamento. Geralmente, isto pode tornar inviável a sua utilização em projetos na prática.

Eventualmente, esta desvantagem pode ser contornada com a evolução natural dos hardwares; esta implementação possibilita que o processamento seja distribuído em computadores com arquitetura paralela; além disso, são desenvolvidos estudos para se obter algoritmos otimizados para a solução de problemas dinâmicos não-lineares

2.2.3 Metodologia Híbrida

As metodologias híbridas são estratégias que combinam as vantagens das metodologias desacopladas e acopladas. Basicamente, os movimentos da unidade flutuante são analisados de modo acoplado e a resposta estrutural das linhas de ancoragem e *risers* de modo desacoplado [8, 13]. O principal objetivo é diminuir o custo computacional e de memória, tornando viável a execução das análises. Por exemplo, no cálculo dos movimentos da unidade flutuante efetuado por um modelo acoplado, as linhas e *risers* podem ser modelados por uma malha relativamente pobre de elementos finitos. Estes movimentos, então, são aplicados numa análise desacoplada para a análise estrutural das linhas de ancoragem e *risers* que podem ser analisadas individualmente e modeladas por uma malha refinada o bastante para fornecer com maior precisão todos os parâmetros da resposta estrutural.

Atualmente, são cada vez mais utilizadas as metodologias híbridas para o projeto do sistema de ancoragem e *risers*, e como conseqüência do seu desenvolvimento, está-se avançando gradualmente na direção de um procedimento totalmente integrado ou acoplado.

2.3 Seqüência de Análises

Sistemas computacionais têm sido desenvolvidos pela indústria do petróleo com o objetivo de efetuar simulações numéricas para obter a resposta dos sistemas *offshore* sob diversas solicitações. Eles também acompanham a evolução das atividades de exploração e produção, uma vez que são utilizados para simular e implementar as metodologias de projetos de sistemas *offshore*.

Os programas descritos a seguir são utilizados no contexto da metodologia híbrida descrita no item 2.2.3, que será a metodologia abordada nesse estudo de *workflows* científicos. Os movimentos da unidade flutuante serão analisados de forma acoplada e a resposta estrutural das linhas de ancoragem e *risers* de forma desacoplada.

2.3.1 Análise hidrodinâmica do casco da unidade flutuante

A análise hidrodinâmica fornece os coeficientes de força de onda que atuam nos sistemas *offshore* e calcula as funções de transferência dos movimentos do sistema [8]. Utiliza uma metodologia desacoplada, uma vez que os efeitos não-lineares das linhas de ancoragem e *risers* não são considerados. Algumas vezes é levada em consideração apenas a rigidez das linhas de ancoragem e *risers*, mas de forma linear. Esses resultados são obtidos através de cálculos de difração de ondas e dinâmica dos fluidos que analisam a interação da água do mar com o casco da unidade flutuante. Essa analise pode ser feita computacionalmente no programa comercial WAMIT [25].

2.3.2 Análise Acoplada de Movimentos da unidade flutuante

Na análise acoplada de movimentos, o sistema *offshore* é representado pelo casco da unidade flutuante, tomando os coeficientes de força de onda, amortecimento potencial e massa adicionada, acoplado a um modelo de representação das linhas de ancoragem e *risers* construído com malhas de elementos finitos relativamente pobres. O tempo de simulação dever ser suficientemente longo para obter a estabilidade estatística da resposta de movimentos do casco [12]. A Figura 2.10 representa o sistema acoplado.

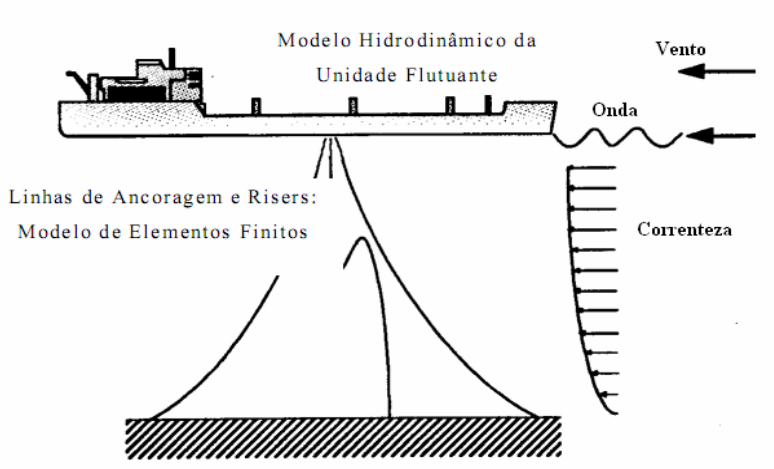


Figura 2.10: Sistema Acoplado (casco + linhas)

O objetivo da análise de movimentos acoplada é obter a resposta de movimentos da unidade flutuante nos seis graus de liberdade do casco (*surge, sway, heave, roll, pitch* e *yaw*), sob condições ambientais definidas de projeto, considerando a influência das linhas de ancoragem e *risers*. Esta análise é feita de maneira integrada à análise estrutural das linhas. A Tabela 2.1 descreve as direções dos movimentos dos sistemas *offshore* e esses movimentos são ilustrados na Figura 2.11.

Tabela 2.1: Descrição dos movimentos dos sistemas *offshore*

Movimento	Direção
Surge	Eixo longitudinal
Sway	Eixo transversal do navio
Heave	Vertical
Yaw	Rotação no plano horizontal
Pitch	Rotação em torno do eixo transversal do navio (passando pelo CG).
Roll	Rotação em torno do eixo longitudinal do navio (passando pelo CG).

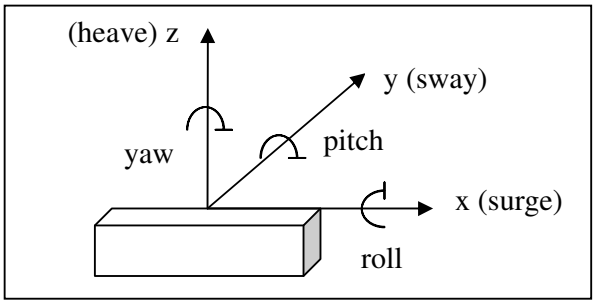


Figura 2.11: Sentido dos movimentos do sistema *offshore*

Neste contexto, pode-se mencionar o programa Prosim [11, 13, 12, 26] que vem sendo desenvolvido desde 1997 em parceria entre a Petrobras e o LAMCSO – Laboratório de Métodos Computacionais e Sistemas *Offshore*, do PEC/COPPE/UFRJ, e baseia-se em um modelo acoplado para a avaliação de sistemas flutuantes.

Ele emprega uma formulação acoplada onde, a cada instante do processo de integração no tempo das equações de movimento do casco, efetua-se uma análise não-linear dinâmica de um modelo de elementos finitos de cada uma das linhas, sob ação da onda, correnteza, peso próprio e das componentes de movimento transmitidas pelo casco.

O Prosim também possui uma interface gráfica de pré e pós-processamento para gerar automaticamente os arquivos com modelos de análise e para visualizar os resultados. Este módulo de pré-processamento é denominado SITUA. Seu nome foi inspirado na sua disponibilidade de recursos de análise estática e geração de modelos para **S**ituações de **I**nstalação e **A**varia de **U**nidades **A**ncoradas. O conjunto composto pela interface gráfica e pelo programa de análise é denominado: SITUA-Prosim.

2.3.3 Análise estrutural dos *risers*

Na metodologia híbrida, a análise estrutural dos *risers* é feita de forma desacoplada. Ela consiste em efetuar análises de modelos de elementos finitos de um *riser* (ou de uma outra linha qualquer), aplicando no topo os movimentos da unidade flutuante calculados na análise dos movimentos e aplicando ao longo da linha os carregamentos de onda e correnteza.. O objetivo principal destas análises é obter a resposta estrutural de um *riser*, de forma que os esforços nos trechos mais críticos possam ser comparados aos critérios de projeto.

Neste contexto, pode-se mencionar o programa ANFLEX (*Static and Dynamic Analysis for Mooring Systems and Risers*) [27, 28, 29], que desde 1991 vem sendo desenvolvido e utilizado pela Petrobras para análise estrutural não-linear estática e dinâmica de estruturas esbeltas, tais como *risers* e linhas de ancoragem. O programa tem sido atualizado continuamente para atender à implementação de novas metodologias, e desenvolver interfaces gráficas para melhor simular as situações exploração e produção *offshore*.

No procedimento atual de análise, inicialmente efetua-se uma análise não-linear estática para determinar a configuração de equilíbrio. Os carregamentos de peso

próprio, correnteza e *offset* estático da embarcação são normalmente impostos sobre a estrutura e são aplicado incrementalmente. No primeiro passo, aplica-se apenas a carga total do peso próprio, e nos demais, aplica-se incrementalmente a correnteza e o *offset* estático, que representa o deslocamento do topo do *riser*.

Em seguida, é feita uma análise não-linear dinâmica no domínio do tempo, que se inicia a partir dos resultados obtidos no último passo da análise estática. Esta análise inclui todas as parcelas estáticas do carregamento e se acrescenta as parcelas dinâmicas, que pode ser determinística ou aleatória. Na metodologia híbrida as parcelas dinâmicas do movimento da embarcação, previamente calculadas por um modelo acoplado, são efetivamente consideradas. As ondas também podem ser consideradas atuando diretamente na linha estudada.

Vale observar que a metodologia híbrida pode, no contexto das análises acopladas, ser utilizada para fornecer totalmente ou parcialmente os movimentos da unidade flutuante, bem como os *offsets* estáticos de cada caso de carregamento. Num contexto mais específico apenas os movimentos de baixa freqüência calculados nas análises acopladas poderiam ser utilizados e superpostos com RAOs de movimento (movimento de alta freqüência), previamente calculados pelo WAMIT, por exemplo. De maneira geral, os movimentos completos (de baixa e alta freqüência) provenientes das simulações acopladas são considerados na metodologia híbrida de análise de *risers*.

2.3.4 Análise de fadiga de onda

A fadiga é um estado limite bastante importante no projeto e na manutenção de estruturas *offshore*. O dano por fadiga pode ser entendido como a falha que ocorre nos materiais submetidos a variações cíclicas de tensões e deformações, através de um processo de acumulação de dano, ciclo a ciclo, ou seja, a carga aplicada não é suficiente para provocar a ruptura imediata da estrutura, mas a ruptura acontece após certo número de variações de carregamento, quando o dano acumulado na estrutura atinge um determinado nível [30].

O dano devido à fadiga depende da história completa das tensões ao longo da vida útil da estrutura, devendo-se considerar todo o conjunto de condições de carregamento que podem ocorrer em todas as fases do projeto do sistema *offshore*: fabricação, transporte, instalação e operação.

O dano é calculado em função dos valores das amplitudes de variação das tensões que ocorrem ao longo da vida útil prevista. A equação a seguir ilustra o dano associado a um certo valor de amplitude de variação de tensão **i** e uma dada condição de carregamento **j**.

$$\text{Dano}_{ij} = \frac{n_{ij}}{N_i}$$

Sendo:

n_{ij} → Número de ciclos de ocorrência da amplitude de variação das tensões **i**, considerando-se 1 ano de atuação da condição de carregamento **j**.

N_i → Número de ciclos admissível associado à variação de tensão **i**. obtido nas curvas S-N.

Para calcular o dano total de um carregamento j, durante 1 ano, associado a todas as amplitudes de variação de tensões, utiliza-se a Regra Cumulativa de Miner multiplicada pelo percentual de ocorrência de cada caso de carregamento.

$$\text{Dano}_j = \sum_i \left(\frac{n_{ij}}{N_i} \right) . \gamma_j$$

Sendo:

γ_j → Número de ciclos de ocorrência da amplitude de tenão **i**, considerando-se 1 ano de atuação da condição de carregamento **j**.

O dano total acumulado para todas as amplitudes de varição de tensões é dada pela equação abaixo.

$$\text{Dano}_T = \sum_i \sum_j \left(\frac{n_{ij}}{N_i} \right) . \gamma_j$$

O valor correspondente à fadiga é o inverso do dano total.

$$\text{Fadiga} = \frac{1}{\text{Dano}_T}$$

Finalmente, a vida útil representa 10% do valor da fadiga.

$$\text{Vida Útil} = \frac{\text{Fadiga}}{10}$$

.

Para a verificação da fadiga, podem ser utilizados modelos de análise no domínio do tempo ou no domínio da freqüência. No caso das análises de fadiga aleatória de SCR's no domínio do tempo, uma melhor representação das características não-lineares do *riser* é obtida. Em estruturas *offshore*, um tipo de avaliação da fadiga deve ser feita através de análises dinâmicas determinísticas ou aleatórias da estrutura, considerando-se as condições ambientais da locação. Desta forma, é possível determinar o histórico de longo prazo das tensões locais nos diversos pontos da estrutura e identificar todos os ciclos de tensões e as suas respectivas amplitudes.

Os resultados da análise de fadiga são apresentados em termos de vida útil do *riser*. São efetuados diversos estudos qualitativos e paramétricos, procurando verificar a existência de relação entre uma componente de movimento e o dano. Os resultados são apresentados separadamente para as regiões mais críticas do *riser*, que para um SCR são as regiões do topo que tem contato com a unidade flutuante e a região do TDP (*Touch Down Point*), que sofre contato com o solo marinho.

Na sequencia de análises considerada aqui para a implementação do workflow, os resultados da análise estrutural do ANFLEX são pós-processados para o cálculo da fadiga dos *riser* através do programa POSFAL [31, 32], também desenvolvido pela Petrobras.

Capítulo 3 - Conceitos sobre Workflow

3.1 *Workflows* científicos

Em sistemas ou atividades baseadas em processos, onde um processo pode ser definido como um conjunto de um ou mais procedimentos ou atividades relacionadas, que coletivamente atingem um objetivo dentro de uma estrutura organizacional que define papéis funcionais e relações, podem ocorrer problemas como a má distribuição de trabalho, o surgimento de atividades que atrasam as demais ("gargalos"), dificuldade de acompanhamento, sincronização e avaliação de atividades, entre outros. Uma linha de pesquisa que visa solucionar ou minimizar tais problemas e melhorar o fluxo de atividades baseadas em processos é a área que estuda o Workflow [33, 34].

Um workflow pode ser compreendido pela seguinte definição: É a automação do processo de negócio onde documentos, informações ou tarefas são passadas de um participante para o outro de acordo com um conjunto de regras de procedimentos para se atingir um objetivo comum de negócio [35]. Atualmente, este conceito é largamente aplicado em ambientes científicos. Um workflow científico é a especificação de um processo que descreve um experimento científico [36]. Entretanto, diferentemente dos ambientes de negócio, os ambientes científicos visam a análise e a manipulação de grande volume de dados. A Figura 3.1 [37] destaca as diferenças entre o *workflows* de negócio e workflows científicos.

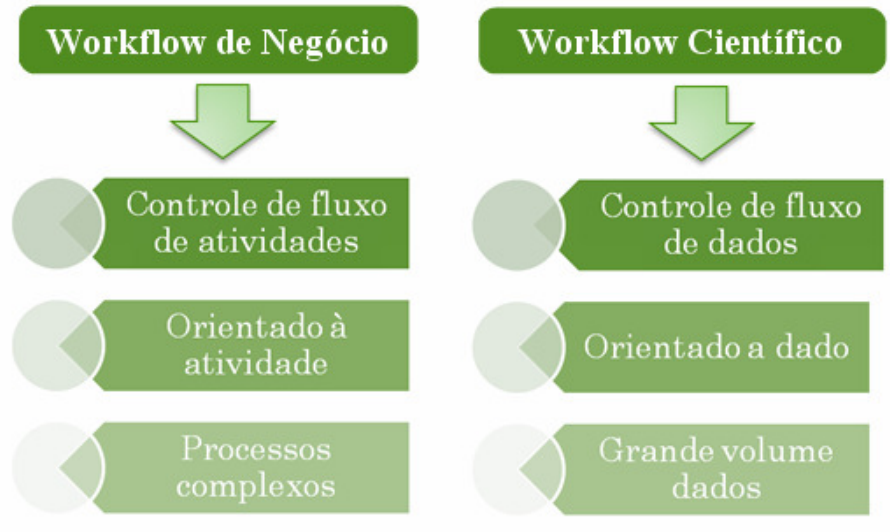


Figura 3.1: Diferenças entre *workflows* de negócio e *workflows* científicos

A modelagem computacional é um dos passos de uma abordagem particular para o processo de experimento científico, onde o fenômeno a ser estudado é simulado em um computador. Para esse tipo de experimento foi definido o termo "*in silico*".

O experimento *in silico* pode ser feito utilizando-se um ambiente para modelagem computacional que auxilie em todas as etapas da elaboração do experimento científico [14, 15]. Dentre os requisitos desejáveis de um ambiente para modelagem computacional, destacam-se: reutilização de programas e sistemas já desenvolvidos; encadeamento dos sistemas computacionais usados nas diversas etapas do experimento; auxílio no gerenciamento dos vários processos envolvidos nas etapas de construção e execução do experimento; e meios de estender as funcionalidades e tarefas básicas do ambiente para modelagem computacional [38].

De um modo geral, *workflows* científicos são ferramentas apropriadas para acessar e tratar dados científicos [39], resultantes de medições experimentais; processamento de imagens (por exemplo, imagens médicas e de satélites); ou de simulações numéricas, como no caso deste estudo. O sistema de gerência de workflow científico corresponde ao processo automatizado que gerencia processos e dados para criar soluções computacionais para um problema científico, representado pelo workflow. Alguns exemplos de etapas desse processo são: a definição dos dados do experimento, a simulação do fenômeno e a análise dos resultados [17].

Cada workflow é constituído de um conjunto de passos que podem envolver acesso a banco de dados (incluindo consultas, análise e mineração de dados), e/ou processamento computacional intensivo. Cada passo do workflow é representado por uma atividade que pode ser realizada por um ou mais atores, de acordo com a concepção do SGWfC. Nesse contexto um ator é um elemento do workflow que executa uma atividade durante uma instância do workflow, que pode ser tanto uma pessoa como um sistema automatizado. Uma instância representa uma ocorrência de uma execução de um workflow.

Assim, o workflow permite ao cientista inspecionar e exibir os dados corretamente conforme são computados, alterando parâmetros de execução quando necessário. Com isso os experimentos podem ser executados várias vezes, reproduzindo os resultados das análises com a mesma metodologia. Essa sistematização facilita a comparação dos resultados além da própria invocação dos programas a serem

executados. A Figura 3.2 [40] ilustra um exemplo de workflow científico que recebe dados científicos, analisa e gera uma saída gráfica.

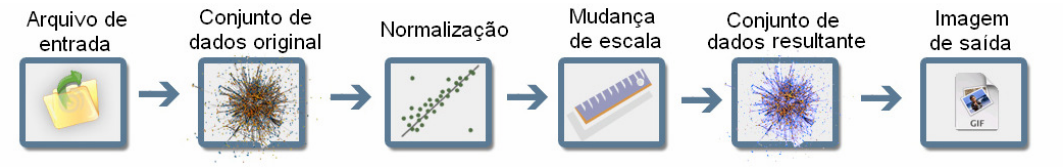


Figura 3.2: Exemplo de *Workflow* científico

Os *workflows* científicos, ao contrário dos comerciais, são normalmente centrados em dados e de execução linear [41]. No entanto, com a maior abrangência de aplicações científicas, começam a surgir projetos que demandam a combinação de diversos modelos de computação. Estes modelos são implementados através de estruturas de controle, iterações, registros de dados intermediários, sincronizações, entre outros [42]. O uso de estruturas de controle é importante para representarem condições, isto é, tratarem expressões lógicas que alteraram o fluxo de execução do workflow. As estruturas de repetição, comumente conhecidas como loops, são necessárias em diversos *workflows* científicos. Por exemplo, através de execuções iterativas, um cientista busca um parâmetro de convergência, em que o experimento vai sendo relaxado até que um determinado número de soluções seja encontrado.

Workflows podem representar modelos teóricos ou análises experimentais; podem ser simples e lineares, ou complexos e não-lineares. Uma característica que é importante de ser ressaltada dos sistemas de workflow é que eles podem ser agrupados hierarquicamente. Um workflow pode conter sub-*workflows* que executam tarefas embutidas. Desta maneira, o diagrama fica mais claro e organizado. Ele pode ser compreendido mais facilmente uma vez que as suas etapas estão melhores distribuídas. Esses sub-*workflows* podem ser reutilizados em outros experimentos e podem ser tornar novos componentes desses sistemas.

Os atores conectados entre si (e outros componentes que serão apresentados a seguir) formam um workflow, permitindo ao cientista inspecionar e exibir os dados corretamente conforme são computados, alterando parâmetros de execução quando necessário. Então os experimentos podem ser executados várias vezes, reproduzindo os resultados das análises. Podem ser destacadas as seguintes características que contribuem para a realização do experimento científico: documentação em todos os

aspectos de uma análise; representação visual de cada passo; interoperabilidade entre sistemas operacionais; e possibilidade para reutilização em diferentes projetos.

Os *workflows* científicos são geralmente compostos por um conjunto de poucas tarefas, mas algumas destas tarefas demandam um processamento de alto desempenho, fazendo com que sua execução necessite de diferentes ambientes de computação. Devido ao elevado esforço computacional, estas tarefas são candidatas a serem executadas em um ambiente de processamento de alto desempenho, como clusters de PC, que é composto por vários processadores interligados em uma rede, pertencentes a uma única unidade; ou por grades computacionais, que é um ambiente amplamente distribuído e heterogêneo, que tem como objetivo compartilhar e agregar recursos geograficamente [20].

3.2 Proveniência

3.2.1 Definição

A proveniência pode ser compreendida no contexto da arte ou do mundo digital, onde se refere, respectivamente, à documentação da história de um objeto de arte, ou à documentação dos processos em um ciclo de vida de um objeto digital [43]. Já no dicionário inglês Oxford, a proveniência é definida como "A fonte ou origem de um objeto, a sua história e seu *pedigree*; um registro da última atribuição e passagem de um item através dos seus vários proprietários." Em experiências científicas, a proveniência nos ajuda a compreender e interpretar os resultados através da análise da seqüência de passos que conduziu a um resultado. Podemos entender a linha de raciocínio utilizada na sua obtenção, e verificar que o experimento foi realizado de acordo com procedimentos aceitáveis, identificar os parâmetros de entrada do experimento, e, em alguns casos, reproduzir o resultado [44].

O interesse em pesquisas sobre proveniência tem crescido na comunidade científica, porque ela é entendida como um componente crucial dos sistemas de workflow, que pode ajudar os cientistas garantir reprodutibilidade das suas análises científicas e processos [45]. Os cientistas e engenheiros gastam um tempo e esforço substancial no gerenciamento manual dos dados e registros de informações apenas para responder a questões básicas, como, por exemplo, quem produziu estes dados e quando? Quem o modificou e quando? Qual processo produziu esses dados? Os mesmos dados brutos conduziram a produção de dois dados distintos? A Figura 3.3 [45] apresenta duas

imagens geradas através de um processo de visualização científica. Se não tivermos nenhuma informação adicional ao nome dos arquivos jpg, somos induzidos a fazer as seguintes perguntas: como estas imagens foram geradas? Elas pertencem a um mesmo paciente?

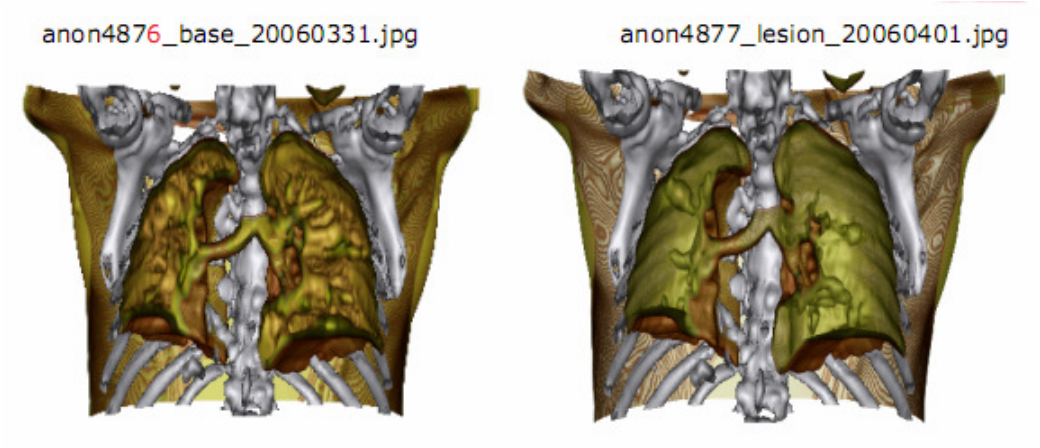


Figura 3.3: Imagens produzidas a partir do SGWfC Vistrails

Nesse contexto a proveniência pode ser compreendida como uma auditoria, um processo para rastrear os dados e as ferramentas de cada experimento. As pesquisas científicas acadêmicas são muito dinâmicas. Constantemente novos dados e ferramentas são criados. Além disso, as análises exigem uma intensa comparação entre os dados novos e os dados já produzidos. Ela fornece importante documentação que é fundamental para preservar os dados, para assegurar a qualidade dos dados e determinar as suas autorias, e de reproduzir, bem como validar os resultados [46]. Estes são importantes requisitos dos processos científicos. É necessário que um sistema, além de conter anotações sobre a proveniência dos dados, possua o armazenamento detalhado da execução de cada atividade no workflow. [16]. A Figura 3.4 adaptada de [45] ilustra a importância dos dados envolvidos na proveniência em *workflows* científicos.

As informações obtidas pela proveniência podem ser recolhidas através dos diferentes recursos nas atividades de um workflow e em múltiplos níveis de detalhes. Nesse sentido, outro conceito importante relacionado a tarefas computacionais é a abstração, o que nos permite dividir tarefas complexas e representá-las em diferentes níveis de granularidade.

A utilidade de proveniência, em um determinado domínio está relacionada com a granularidade em que é coletada. A extensão dos requisitos varia da proveniência nos atributos a tuplas em um banco de dados para a proveniência arquivos gerados durante o

processo. Aumentar o nível da abstração permite o desenvolvimento de um sistema de proveniência mais flexível. O custo do registro e do armazenamento dos dados pode ser inversamente proporcional à granularidade da proveniência [47].

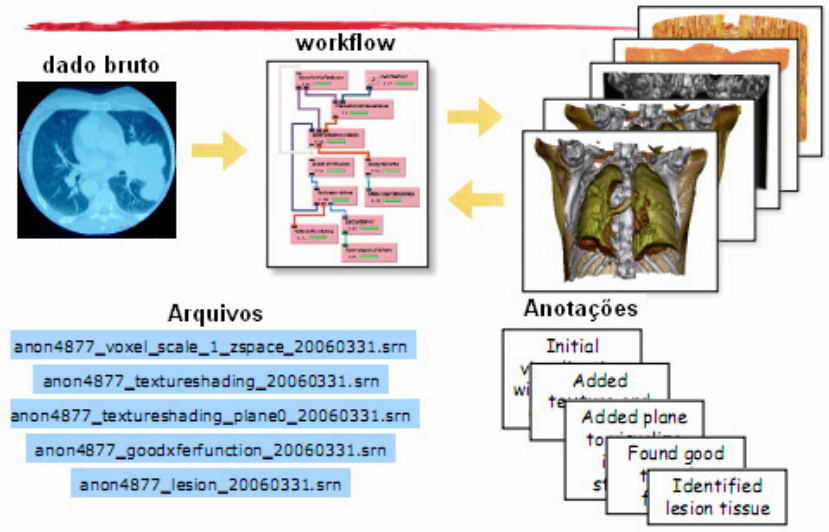


Figura 3.4: Dados envolvidos na proveniência de *workflows* científicos

3.2.2 Modelos de representação

Apesar da grande quantidade de pesquisas nesse tema, ainda não há um modelo formal de proveniência para sistemas de workflow que define com precisão o significado de proveniência, tendo em conta a estrutura de classes e passo dos dados produzidos [48]. Uma solução de gerência de proveniência é constituída por três componentes principais: um mecanismo de captura, um modelo de representação, e uma infra-estrutura de armazenamento, acesso e consultas [49].

O mecanismo de captura da proveniência precisa ter conhecimento dos detalhes mais relevantes das atividades que serão processadas, tais como os seus passos, informações da execução, e anotações específicas dos usuários [50, 51]. As atenções desse tipo de mecanismo podem estar concentradas ou divididas em três níveis principais: processo, workflow, e sistema operacional; envolvem, respectivamente, a documentação das atividades envolvidas no workflow, os recursos disponibilizados no sistema de gerenciamento de workflow, e a necessidade de haver modificação de scripts ou programas para executarem corretamente.

Os pesquisadores têm proposto vários modelos para representar a proveniência na literatura. Todos estes modelos suportam de alguma forma a proveniência retrospectiva, e a maioria dos modelos que os sistemas de *workflows* usam, fornece meios para capturar a proveniência de modo prospectivo. Embora esses modelos se diferenciem em várias formas, incluindo a sua utilização de estruturas e estratégias de armazenamento, todos compartilham um tipo essencial de informação: a dependência entre as atividades e os dados envolvidos no workflow.

Apesar de uma base comum, os modelos de proveniência tendem a variar em função do domínio e das necessidades dos usuários. Embora a maioria dos modelos se empenhe para armazenar conceitos gerais, os estudos de caso frequentemente utilizados influenciam o desenvolvimento do modelo. Por exemplo, o Taverna [52] foi desenvolvido para apoiar a criação e o gerenciamento de *workflows* no domínio da bioinformática, então, portanto, prevê uma infra-estrutura que inclui suporte para ontologias disponíveis neste domínio. Já o Vistrails [53] foi desenvolvido visando ao apoio de *workflows* de visualização científica, por isso possui as bibliotecas que implementam as atividades desse domínio.

Quando se trata de tarefas computacionais, existem duas formas de proveniência: prospectiva e retrospectiva [49]. A primeira captura uma atividade da especificação do workflow e faz a associação com os passos que devem ser seguidos para gerar os dados. A segunda captura as etapas executadas, assim como as informações sobre o ambiente utilizado para a geração dos dados, em outras palavras, é um log detalhado de uma tarefa computacional que foi executada. A proveniência retrospectiva não depende da presença da proveniência prospectiva, uma vez que é possível capturar informações como o processo foi executado, quem o executou, em quanto tempo foi executado, sem ter conhecimento prévio da seqüência das atividades envolvidas.

Um componente importante de proveniência é a informação sobre a causalidade, que corresponde à descrição do processo ou da seqüência de atividades, que, juntamente com dados de entrada e de parâmetros, levou à criação dos dados em questão. Podemos inferir causalidade em ambas as proveniências: prospectiva e retrospectiva. Podemos também representar causalidade graficamente onde os nós correspondem aos processos e produtos e as bordas correspondem a dependências entre dados ou entre processos. As dependências entre processos podem ser úteis para documentar o processo de criação de

dados, mas também podemos usá-los para reproduzir ou validar um processo. Outro componente-chave da proveniência é a documentação das informações definidas pelos usuários que não são capturadas automaticamente, mas que registram importantes decisões e notas. Estes dados muitas vezes são criados sob a forma de anotações.

Muitos congressos têm sido organizados com o objetivo de desenvolver um modelo para representar a proveniência, podemos destacar o Provenance Challenge [54]. Nesse contexto foi elaborado o Open Provenance Model (OPM) [43] que possui os seguintes requisitos:

- Permitir que as informações da proveniência sejam trocadas entre os sistemas, por meio de uma camada de compatibilidade com base em um modelo compartilhado proveniência;
- Permitir aos programadores desenvolverem e compartilharem ferramentas capazes de operar no próprio modelo de proveniência;
- Fornecer uma representação digital da proveniência para qualquer sistema, seja computacional ou não;
- Definir um conjunto de regras que identificam as inferências válidas que podem ser feitas nos grafos de proveniência.

Na especificação do OPM, também foram definidos alguns não-requisitos:

- Não tem como propósito especificar as representações internas que sistemas devem adotar para armazenar e manipular a proveniência internamente; os sistemas estarão livres para adotar suas próprias representações de acordo com o seu propósito;
- Não tem como propósito definir uma sintaxe para este modelo através de linguagens de marcações; a implementação do modelo em XML, RDF ou outros serão especificados futuramente em outros documentos;
- Não será especificar os protocolos para armazenar essas informações nos repositórios de dados;
- Não serão especificados os protocolos para consulta aos repositórios de dados.

O OPM é baseado em três entidades básicas: artefatos, processos e agentes. O Artefato representa um objeto físico ou digital. O Processo é definido como uma atividade ou série de atividades realizadas em artefatos ou causada por eles, e que resulte na produção de novos artefatos. O Agente é uma entidade que atua como um

catalisador de um processo, permitindo, acelerando, controlando e afetando a sua execução. O OPM define uma notação gráfica e uma definição formal dos grafos de proveniência. Os artefatos são representados por círculos, os processos são representados por retângulos e, por último, os agentes são representados por octógonos.

O grafo de proveniência tem como finalidade representar as dependências causais entre as entidades definidas no modelo. Portanto, um grafo de proveniência é um grafo direcionado, cujos nós são artefatos, processos e agentes, e as linhas que os unem representam os relacionamentos entre ele. A ponta com a seta representa a fonte, o efeito da dependência; e a outra extremidade representa o seu destino, denotando a sua causa, conforme ilustrado na Figura 3.5 [44].

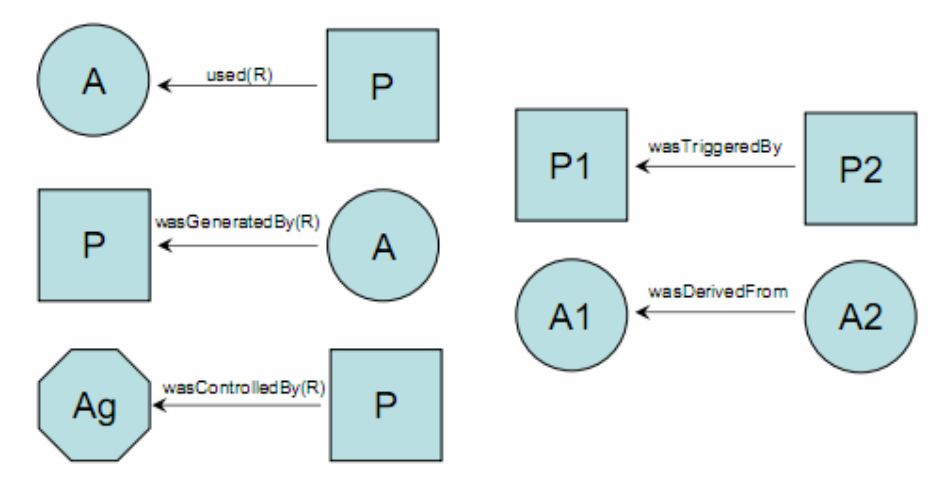


Figura 3.5: Relacionamentos entre as entidades do OPM

As primeiras duas dependências exibidas na Figura 3.5 expressam um processo que utiliza um artefato, e um artefato que foi gerado por um processo. Uma vez que um processo pode ter diversos artefatos utilizados, é importante identificar os papéis que esses artefatos desempenham no workflow. Do mesmo modo, um processo pode ter gerado muitos artefatos, e cada um teria um papel específico. Por exemplo, o processo da divisão algébrica usa dois números, com papéis dividendo e divisor, e produz dois números, com papéis e quociente restante. Os papéis são significativos apenas no contexto do processo em que são definidos. Essas últimas relações são representadas, respectivamente, pelas expressões **used** e **wasGeneratedBy**.

Um processo é causado por um agente, atuando essencialmente como um catalisador ou controlador: esta dependência causal é expressa pela expressão **wasControlledBy**. Levando em consideração que um processo pode ter sido catalisado por vários agentes, também é necessário identificar os seus papéis como catalisadores.

Pode ser constatado que a dependência entre um agente e um processo representa um relacionamento de controle, e não uma de derivação. Ela é introduzida no modelo para facilmente expressar como um usuário ou um agente externo controla um processo.

Ainda pode haver casos em que não estamos cientes do processo que gerou um artefato A2, mas sabemos que o artefato A2 foi derivado de um outro artefato A1. Da mesma maneira, podemos não estar cientes do artefato exato que o processo P2 utilizou, mas que sabemos que alguns artefatos foram gerados por um outro processo P1. Podemos dizer então que o processo P2 foi desencadeado a partir do processo P1. Essas últimas relações são representadas, respectivamente, pelas expressões **wasDerivedFrom** e **wasTriggeredBy**.

As entidades e as dependências causais definidas em um grafo do OPM são os pilares para a construção de um modelo de entidade relacionamento de banco de dados para armazenar a proveniência dos dados. As entidades representam as tabelas que contém os artefatos, processos, agentes, papéis. As dependências causais irão representar as tabelas associativas que constituem os relacionamentos entre as tabelas das entidades. O quadro da Figura 3.6 [44] contém as definições que são obtidas a partir do grafo de proveniência que constituem o núcleo-base para a modelagem do OPM em um banco de dados relacional.

$$
\begin{array}{rcll}
ProcessId & : & primitive\ set & \text{(Process Identifiers)} \\
ArtifactId & : & primitive\ set & \text{(Artifact Identifiers)} \\
AgentId & : & primitive\ set & \text{(Agent Identifiers)} \\
Role & : & primitive\ set & \text{(Roles)} \\
Account & : & primitive\ set & \text{(Accounts)} \\
Process & = & ProcessId \rightarrow \mathbb{P}(Account) & \\
Artifact & = & ArtifactId \rightarrow \mathbb{P}(Account) & \\
Agent & = & AgentId \rightarrow \mathbb{P}(Account) & \\
Used & = & ProcessId \times Role \times ArtifactId \times \mathbb{P}(Account) & \\
WasGeneratedBy & = & ArtifactId \times Role \times ProcessId \times \mathbb{P}(Account) & \\
WasTriggeredBy & = & ProcessId \times ProcessId \times \mathbb{P}(Account) & \\
WasDerivedFrom & = & ArtifactId \times ArtifactId \times \mathbb{P}(Account) & \\
WasControlledBy & = & ProcessId \times Role \times AgentId \times \mathbb{P}(Account) & \\
Alternate & = & Account \times Account & \\
OPMGraph & = & Artifact \times Process & \\
 & & \times Agent \times \mathbb{P}(Used) & \\
 & & \times \mathbb{P}(WasGeneratedBy) \times \mathbb{P}(WasTriggeredBy) & \\
 & & \times \mathbb{P}(WasDerivedFrom) \times \mathbb{P}(WasControlledBy) & \\
 & & \times \mathbb{P}(Alternate) &
\end{array}
$$

Figura 3.6: Núcleo da modelagem do OPM em um banco de dados

Existem várias abordagens para a captura e modelagem da proveniência, mas só recentemente o problema de armazenamento, acesso e consulta começou a receber atenção [55]. Os pesquisadores têm utilizado uma ampla variedade de modelos [56, 57, 58] e sistemas de armazenagem de dados [59, 60], que vão desde XML e sistema de arquivos a registros armazenados em tabelas de bancos de dados relacionais. Uma das vantagens dos sistemas de arquivos é que os usuários não precisam de infra-estruturas adicionais para armazenar informações proveniência. Por outro lado, um banco de dados relacional provê um armazenamento centralizado e eficiente, em que um grupo de usuários pode compartilhar de forma eficaz e eficiente, toda uma infra-estrutura de consulta a proveniência de dados, a qual é uma componente necessária de um sistema de gerenciamento de proveniência, especialmente quando grandes volumes de informação são capturados. Um outro estudo que pode ser destacado para sistemas de armazenamento de proveniência é o PASS (Provenance-Aware Storage System) que é um sistema que automaticamente coleta, armazena, gerencia e provê consulta à proveniência [61]. A Figura 3.7 adaptada de [62] resume a arquitetura das pesquisas desenvolvidas para as pesquisas de gerência de proveniência.

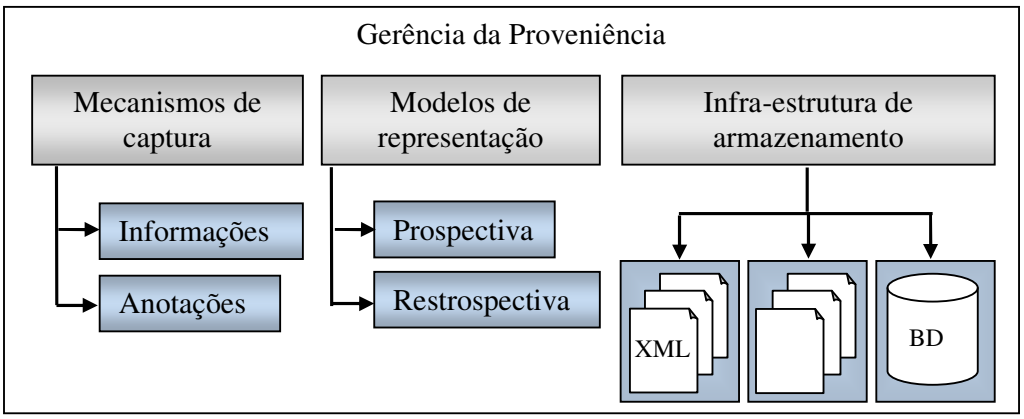


Figura 3.7: Arquitetura da gerência de proveniência

3.2.3 Proveniência Evolutiva

Embora a questão da proveniência no contexto dos *workflows* científicos tem recebido uma considerável atenção recentemente, a maioria das pesquisas tem foco na proveniência dos dados, isto é, conservando a informação de como um determinado dado foi gerado. Esta informação tem muitos usos, de simplesmente informativo até permitir a recriação dos dados produzidos possivelmente com diferentes parâmetros. No entanto, enquanto estão resolvendo um problema particular, os cientistas muitas vezes

criam diversas variações de um workflow em um processo de tentativa e erro [63]. Estas versões dos *workflows* podem variar tanto nos valores de parâmetros utilizados como nas suas especificações. Se apenas a proveniência dos dados individuais é mantida, informações úteis sobre a relação dentre os *workflows* são perdidas.

Um modelo de proveniência orientado a mudanças é capaz de capturar as modificações de um workflow. A evolução de um workflow pode ser representada como uma árvore, onde cada nó é uma versão da especificação workflow e cada ramo representa uma ação. Dentre os benefícios desse modelo podemos destacar a capacidade de compreende a linha de raciocínio de sequência para frente e para trás, e comparar os diferentes resultados.

O VisTrails é um exemplo de SGWfC que captura informações detalhadas sobre este processo de refinamento: Conforme um usuário modifica um workflow, o Vistrails transparentemente captura as mudanças de atividades, por exemplo, a inclusão ou retirada de uma atividade, a modificação de um parâmetro, a adição de uma ligação entre as atividades, semelhantemente a um log de uma base de dados.

A representação orientada à mudança da evolução de um workflow é concisa e usa substancialmente menos espaço do que uma alternativa de armazenamento de múltiplas versões de um workflow [64]. Entretanto, apenas o SGWfC Vistrails permite a gerência de versões. O modelo também é extensível. A álgebra subjacente das ações pode ser personalizada para suportar mudanças em diferentes níveis de abstração. Além disso, permite construção de uma interface intuitiva na qual a evolução de um workflow é apresentada como uma árvore [45], ilustrada na Figura 3.8 [45], permitindo aos engenheiros retornarem a uma versão anterior de uma forma intuitiva, para desfazer alterações incorretas, e serem lembrados das ações que levaram a um determinado resultado [65].

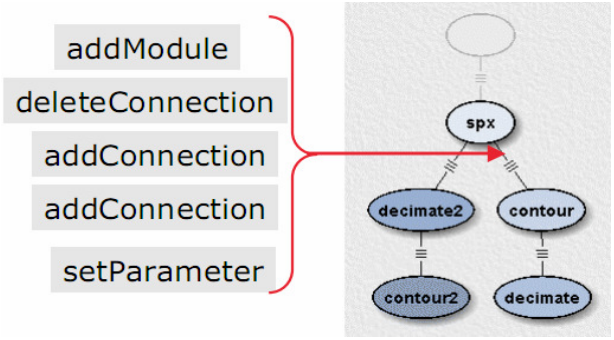


Figura 3.8 : Árvore da proveniência da evolução de um workflow

Como já foi dito neste capítulo, os *workflows* científicos são frequentemente utilizados como um meio para criar complexos procedimentos computacionais para resolver problemas científicos em diversas áreas. A concepção de um workflow em áreas de pesquisas multidisciplinares, como bioinformática e modelagem ambiental, muitas vezes exigem uma cooperação entre vários cientistas em localizações geográficas diferentes. Atualmente, existem poucas ferramentas disponíveis para apoiar a concepção de *workflows* colaborativos. Os pesquisadores são frequentemente limitados a trocar especificações dos *workflows* através de mensagens eletrônicas, consequentemente, o processo tende a ser lento. Em alguns casos, pode ser possível dividir o trabalho de tal modo que colaboradores possam trabalhar de forma independente e, em seguida, combinar os seus trabalhos para atingir um resultado final.

Para apoiar a concepção de *workflows* em ambientes de trabalho colaborativo, também há estudos para desenvolver um mecanismo que permite que os cientistas possam trabalhar simultaneamente em uma tarefa, e acompanhar as alterações feitas em tempo real. Entretanto, estas alterações não são automaticamente incorporadas para este recurso não funcionar como um sistema de controle de versão. Cada alteração gera um novo ramo de exploração e permite ao usuário escolher quais ramos utilizar, independentemente de quem os criou. Desta maneira, é possível representar visualmente uma árvore contendo todas as contribuições. Os cientistas podem compartilhar e receber atualizações e, ao mesmo tempo, têm a opção de desconsiderar atualizações que não os interessam. Faz-se necessário desenvolver um algoritmo para a sincronização e discutir como ela pode ser utilizada na prática [66].

A fim de apoiar a concepção de *workflows* em ambientes colaborativos, é necessário utilizar uma arquitetura de proveniência que suporte um conjunto de workflow versionados e um repositório centralizado de proveniência em que todos os cientistas tenham acesso. Contudo, é necessário um sistema de versões em nível de usuário, pois cada um precisa acompanhar seu próprio desenvolvimento, assim como, o dos outros colaboradores. Mais importante ainda, é preciso proteger os trabalhos; não deve ser permitido apagar ou sobrescrever as suas próprias atualizações. Também é necessário um repositório centralizado para gerenciar todos os *workflows* e fornecer os meios de comunicação para notificar os cientistas quando ocorrerem alterações. A união destes dois recursos não apenas permite aos usuários compartilhar de forma eficiente o conjunto de *workflows*, mas também lhes permite acompanhar todo o histórico das

especificações do workflow, independentemente da quantidade de cientistas que estão colaborando no desenvolvimento do projeto.

3.3 Modelagem de *Workflows* Científicos

Como mencionado nos capítulos anteriores, engenheiros freqüentemente realizam experimentos e simulações em um conjunto de sistemas de análise dinâmica para simulação do comportamento de sistemas *offshore* flutuantes de exploração e produção de petróleo, muitos deles desenvolvidos por pesquisadores da COPPE. O objetivo deste estudo é modelar e implementar esses experimentos como *workflows* científicos.

No entanto, poucos destes engenheiros têm a percepção de que já executam um workflow. Para o contexto deste estudo, é de suma importância que estes *workflows* sejam muito bem identificados e explicitados. Esta seção apresenta uma abordagem para apoiar a identificação destes *workflows* de forma a tornar explícitas as atividades executadas ao longo do workflow, assim como os produtos utilizados e gerados por estas atividades. Resumindo, esta abordagem fornece apoio para capturar os requisitos relevantes à composição de um workflow, e oferece meios para que se realize a verificação e validação dos requisitos capturados.

Na ciência da computação, em especial nas áreas de Engenharia de Software e Banco de Dados, existem técnicas que podem ser adaptadas para fornecer o apoio necessário ao gerenciamento de experimentos científicos em larga escala [67]. Diversos trabalhos vêm sendo propostos no intuito de prover formalismo para descrições de *workflows*. Eles fazem uso de diferentes técnicas, tais como:máquinas de estados finitos, álgebra de processos [68], máquinas de estados abstratos, lógica temporal, lógica transacional e redes de Petri [69]. Durante a etapa de concepção de um workflow, é necessário entender bem o domínio da aplicação. Nesta etapa, são necessárias reuniões entre os membros do grupo de pesquisa, as quais envolvem a tomada de decisão e análises refinadas sobre a modelagem do workflow. Para executar esta etapa, foi adotado o uso de entrevistas, que na Engenharia de Software é uma das técnicas de levantamento de requisitos mais utilizadas.

Então, é importante ressaltar que a abordagem que está sendo utilizada nessa identificação de informações não é paradigma obrigatório e exclusivo a ser adotado. Na verdade trata-se de uma abordagem inovadora proposta por Mattoso et al [62] do

Programa de Engenharia de Sistemas e Computação (PESC- COPPE-UFRJ). Não foram encontrados outros trabalhos na literatura que indiquem como deve ser feito o processo de concepção de *workflows*. Essa abordagem caracteriza-se por definir uma forma mais clara e intuitiva para o levantamento de requisitos de *workflows* científicos. Ela se mostrou muito adequada para representar o contexto deste workflow de sistemas *offshore* e, portanto, foi adotada nessa pesquisa. O intercâmbio de conhecimento entre os pesquisadores do PEC e do PESC foi muito importante para a modelagem e implementação do workflow.

Esta abordagem também está sendo utilizada no projeto da Rede Galileu da Petrobras em pareceria com os Programas de Engenharia Civil e de Engenharia de Sistemas e Computação da COPPE. A Rede Galileu é uma das redes temáticas da Petrobras que têm como objetivo compartilhar os conhecimentos adquiridos entre a empresa e as universidades em áreas de pesquisa que tragam benefícios a indústria do petróleo e gás. Nesse contexto será desenvolvido um meta-workflow que irá contemplar outros experimentos científicos da engenharia *offshore*.

Os trabalhos relacionados ao levantamento dos requisitos de um workflow científico utilizam a Metodologia de Reuso [70, 71, 72]. Primeiramente são definidas as atividades e os tipos de dados envolvidos. As atividades serão criadas apenas em um nível abstrato, assim define-se o workflow abstrato. Nessa etapa somente os requisitos e as regras do experimento científico são relevantes, o workflow não leva em consideração aspectos do modo de execução. Este workflow pode ser adaptado para situações específicas, e, conseqüentemente, serem instanciados para serem executados. No workflow concreto, entretanto, há a definição de características tecnológicas, como a indicação de programas e de recursos necessários para se executar as atividades. Cada uma das atividades é associada aos respectivos softwares que desempenharão as suas funções no fluxo. O workflow concreto é uma instância específica de um workflow abstrato para resolver um determinado problema [62]

Em cada etapa alguns procedimentos precisam ser executados, visando à geração de artefatos que contenham a especificação do workflow. A primeira etapa corresponde à realização das seguintes atividades: reuniões para aquisição das informações, preenchimento dos formulários e geração da modelagem. Os produtos gerados são: modelagem – atualmente representada por diagramas de atividades UML 2.0; e formulários preenchidos – formulários de atividades, resultados e ferramentas. A

segunda etapa corresponde à verificação e validação dos produtos gerados. Está prevista a revisão dos formulários preenchidos e dos diagramas. Em seguida, deve ser feita uma reunião para validação da modelagem criada e dos formulários preenchidos. No caso de serem necessárias correções nos formulários e nos diagramas, a repetição da verificação e validação deve ser realizada. Os produtos gerados são: a modelagem do workflow científico validada; e os formulários preenchidos e agora validados.

A transformação do conhecimento subentendido em explícito é importante, pois permite a captura das informações relevantes ao experimento científico gerado. Desta maneira, torna-se necessário especificar um conjunto de características pertinentes às atividades do experimento científico. A principal abstração que representa o experimento é o próprio workflow científico.

Para o levantamento dos requisitos necessários à modelagem do workflow é necessário que os modeladores interajam com os especialistas do experimento. Portanto, pode-se pensar, inicialmente, em formulários para representar os dois tipos de elementos da modelagem do workflow, isto é, um para atividades e outro para produtos. Cada formulário reúne um conjunto de características que necessitam ser especificadas e auxiliam na descrição e representação destes elementos.

Entretanto, uma vez que é necessário um contexto e a infra-estrutura computacional para a execução e suporte de um workflow, também se pode definir um formulário que permita descrever as informações relativas às ferramentas utilizadas nas atividades modeladas.

3.4 Sistemas de gerência de workflow

Os sistemas de gerência de *workflows* científicos (SGWfC) são as ferramentas computacionais utilizadas para gerenciar, definir, implementar, executar e, em alguns casos, monitorar os *workflows* científicos [19]. A maioria destes utiliza linguagem de programação gráfica (GPL). Uma GPL (Graphic Programing Language) representa a construção de programas manipulando elementos gráficos, mas há alguns softwares que apenas especificam os *workflows* textualmente através de scripts. Utilizam expressões visuais e símbolos gráficos: ícones, flechas, caixas, arcos, círculos. Esses elementos se conectam, formando o ciclo de execução do programa. A GPL promove a reutilização dos componentes de interfaces e segue o paradigma de desenvolvimento baseado em componentes. Como exemplos de ambientes de desenvolvimento que utilizam GPL

podemos citar o Microsoft Robotics Studio; o LabView, utilizado em instrumentação eletrônica; e os sistemas de wokflow Vistrails, Taverna, Triana, Kepler, dentre outros. Estes SGWfC serão descritos nesta seção, e também serão apresentados os motivos pelos quais o Kepler foi escolhido para implementar o workflow deste estudo.

Nos SGWfC, os *workflows* são representados como grafos cíclicos ou acíclicos. Aqueles baseados em grafos acíclicos incorporam alguns padrões importantes, como por exemplo, seqüência, dependência temporal entre as tarefas; paralelismo, execução concorrente entre as tarefas; e escolha, execução de uma tarefa mediante uma dada condição. Os *workflows* baseados em grafos cíclicos, além de contemplar todos os padrões anteriores, também incluem o padrão iteração. Esses padrões representam um grande desafio para o projetista de *workflows* científicos. Se um cientista deseja executar um loop, ele deve escolher um SGWfC que admita tal característica

Poucos SGWfC apresentam as estruturas necessárias para habilitar a execução não linear de um workflow. O Triana e o Kepler atendem essa necessidade, porém não de maneira fácil e intuitiva. Os atores que representam estas estruturas poderiam ser melhores representados graficamente, e otimizados para melhorar o desempenho dos laços. Deste modo, os SGWfCs deveriam ser reprogramados para atender às novas demandas da comunidade científica.

3.4.1 Vistrails

O VisTrails [53] representou a primeira tentativa de organizar o processo de visualização. A motivação para o seu desenvolvimento [73] era apresentar um suporte à exploração de dados através de um recurso de proveniência. Este recurso o diferenciava dos sistemas de visualização anteriores que não o possuíam, como, por exemplo, MayaVi [74] e ParaView[75], que na verdade eram ferramentas mais semelhantes a *workflows* de negócio. Atualmente, ele é a ferramenta de referência para ser usada em visualização científica desenvolvido pela Universidade de Utah nos Estados Unidos. Possui a biblioteca VTK (Visualization Tool Kit) com todos os recursos para visualização, além da gerência eficiente de *workflows* científicos.

Até março de 2009 o Vistrails não apresentava suporte a estruturas de controle nem de repetições, fluxos lineares. Ele também possuía outra desvantagem em relação ao Kepler uma vez que não era capaz de executar um programa externo através de um

componente nativo, para isto é necessário desenvolver um script em Python, que é a linguagem em que ele é implementado.

No VisTrails há uma nítida separação entre a definição do workflow e suas instâncias, ou seja, suas execuções, o que representa uma grande vantagem na gerência de experimentos. A definição do workflow pode ser utilizada como um modelo, para ele ser executado com um conjunto de parâmetros com valores diferentes [76]. Seu mecanismo de proveniência também é bastante sofisticado, pois captura as mudanças nos valores dos parâmetros de execução. Ele consegue capturar o usuário que está logado no sistema operacional para registrar na proveniência quem executou o workflow. Seu objetivo em longo prazo é de fornecer a infra-estrutura necessária para melhorar o processo de descoberta científica e reduzir seu tempo de execução. Ele gerencia os dados e os metadados de um produto resultante da visualização. Ao capturar a proveniência dos processos de visualização e dos dados que são manipulados, o Vistrails permite aos cientistas explorarem de forma eficiente e eficaz os dados do processo.

Ele também foi o primeiro SGWfC a ter controle de versão, ou seja, possuir suporte para registrar a evolução do desenvolvimento das definições de um workflow [77]. Deste modo, ele possui a capacidade de comparar os resultados atuais das análises com os obtidos em versões anteriores do workflow. Apesar de o Vistrails ter sido desenvolvido para sistemas de visualização, sua arquitetura permite que novas bibliotecas sejam integradas para desempenhar novas tarefas como integração e mineração de dados.

A arquitetura do Vistrails pode ilustrar claramente como funciona um sistema de gerenciamento de workflow científico [78]. A Figura 3.9 [45] representa a arquitetura do Vistrails. Os usuários criam e editam o fluxo de dados usando uma interface visual, o Vistrail Builder. As especificações do workflow estão salvas no Vistrail Repository. Os usuários também podem interagir com *workflows* salvos através de um servidor, o Vistrail Server, ou podem importá-los a partir da planilha Visualization Spreadsheet. Cada célula da planilha representa uma instância do workflow. Então, os usuários podem modificar os parâmetros.

A execução do workflow no Vistrails é controlada pela Vistrail Cache Manager, que mantém um registro das operações que são invocadas e os respectivos parâmetros. Somente novas combinações de operações e parâmetros são requisitados do Vistrail

Player, que executa as operações, invocando as funções de visualização: Visualization API e Script API. O Player também interage com o otimizador Optimizer Module, que analisa e otimiza as especificações do workflow. O log com o registro da execução dos *workflows* é mantido no Vistrail Log.

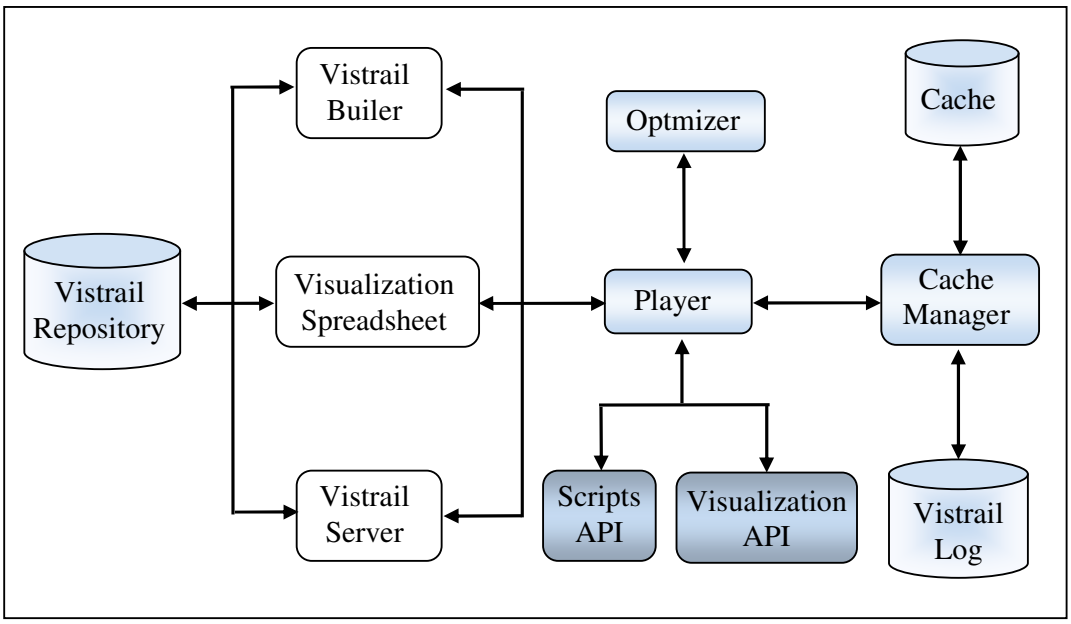


Figura 3.9: Arquitetura do Vistrails

Por ser um sistema muito recente quando do início do desenvolvimento desta dissertação, o VisTrails não foi escolhido. Entretanto, devido a suas evoluções e atual maturidade, ele também pode ser considerado um SGWfC adequado para futuras evoluções do workflow desenvolvido nessa dissertação.

3.4.2 Taverna

O Taverna [52] é um projeto desenvolvido pelo myGrid que envolve o *European Bioinformatics Institute* (EBI) e universidades britânicas. Ele pode ser utilizado nas plataformas Windows, Linux, Mac ou Solaris, foi desenvolvido em Java e *scripts shell* e seu código é aberto e orientado a serviços.

O projeto myGrid [79] é um middleware utilizado para simplificar a análise dos dados nos experimentos científicos, especialmente em bioinformática. O ambiente onde ele é aplicado é muito heterogêneo, os dados e os recursos estão distribuídos. Ele é constituído de *workflows*, web services e tecnologias de web semântica. Os web services podem ser descritos como tecnologias e padrões para expor códigos ou bases de dados em uma máquina através de uma API que pode ser consumida por terceiros

remotamente; As tecnologias de web semântica são responsáveis por encontrar o serviço apropriado durante a modelagem do workflow, encontrar workflow para reuso e armazenar o processo ou o resultado de um experimento em um repositório de proveniência [80].

A partir de uma interface gráfica de usuário, o *workflow* é modelado pelo cientista e traduzido para uma linguagem de definição de *workflows* do Taverna, chamada **Scufl** (*Simple Conceptual Unified Flow Language*) [81]. Em sua arquitetura possui uma máquina de *workflow*, chamada **FreeFluo** desenvolvida pela equipe do Taverna e utilizada para gerenciar, de modo desacoplado da interface, a execução dos *workflows*.

Ele captura as informações de proveniência implicitamente através de um log de eventos que grava eventos referentes ao início e ao fim de cada passo da execução e nos eventos de escrita e de leitura de dados. As dependências entre os dados processados ou criados durante a execução podem ser definidas através da ordem lógica dos eventos. Ele armazena informações de proveniência prospectiva como Scufl, e as de proveniência retrospectiva como um objeto RDF (Resource Description Framework) [82] em um banco de dados MySQL.

3.4.3 Triana

O Triana [83] é um projeto desenvolvido pela Universidade de Cardiff na Inglaterra que combina uma interface visual intuitiva com poderosas ferramentas da análise de dados. Está sendo utilizado por cientistas para uma série de tarefas, tais como processamento de textos, sinais e imagens. É escrito em Java, tem suporte aos recursos de programação distribuída: grids, web services e P2P [84]. Inclui uma grande biblioteca de análise de dados e permite aos usuários facilmente integrar as suas próprias ferramentas.

O Triana é uma aplicação constituída de um conjunto de camadas. A Figura 3.10 ilustra a arquitetura em camadas do Triana. A GUI (Graphic User Interface) representa uma interface gráfica fácil de usar para construir um ambiente de rede; também há uma interface de linha de comando para este fim. O Triana Service é responsável por validar e exectar o workflow, que nele é denominado TaskGraph. Oferece suporte a fluxos paralelos, de controle e de iteração; controla as instâncias dos *workflows*; gerencia os dados envolvidos no workflow; registra a proveniência como uma auditoria e detalhes

do histórico da execução; e distribui os *workflows* para outros Trianas Services que estão sendo executados em outros servidores no ambiente do grid através do GAT (Grid Application Toolkit) [85]. O Triana Controller contém as tarefas que devem ser executadas pelo Triana Service.

No Triana, o workflow é representado no formato WSFL (Web Services Flow Language), que é um linguagem em XML para definir e gerenciar fluxos em web services. Essa representação do workflow é direcionada a um middleware responsável por disparar e gerenciar a execução do processo.

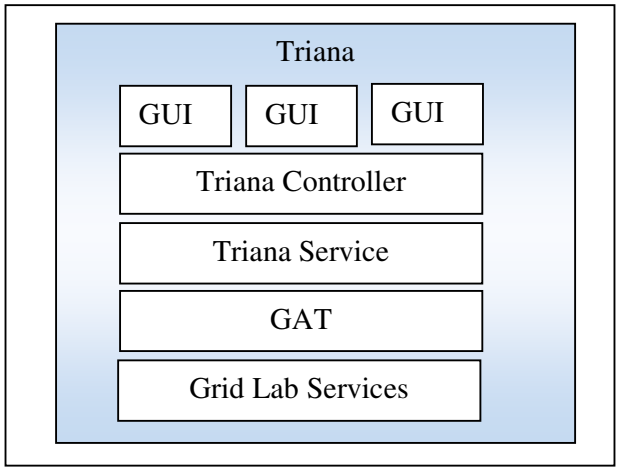


Figura 3.10: Arquitetura do Triana

3.4.4 Kepler

O Kepler [86] é um programa desenvolvido para apoiar as atividades de modelagem e análise de dados científicos. Através de uma interface gráfica com sua caracterização de processos e *workflows* como atores e diretores, cientistas com pouca experiência em ciência da computação podem definir a execução de *workflows* científicos, ferramentas essenciais para o acesso e manipulações complexas de grandes quantidades de dados científicos.

Ele inclui tecnologias de computação distribuída que permitem o compartilhamento de dados e *workflows* entre a comunidade científica [87]. Também oferece acesso a um conjunto em contínua expansão de recursos computacionais, repositórios de dados geograficamente distribuídos e bibliotecas de *workflows*.

O sistema Kepler suporta diferentes tipos de workflow, desde aqueles de baixo nível, do tipo que interessa aos engenheiros de grids computacionais [88, 89], passando pelos analíticos de descoberta de conhecimento para cientistas, até projetos de nível

conceitual, que podem se tornar executáveis somente após alguns passos de refinamento.

O sistema foi construído sobre o *framework* do software livre Ptolemy II, desenvolvido na Universidade de Berkeley na Califórnia. Entretanto, ele é desenvolvido em parceria por várias universidades: UC Davis, UC Santa Bárbara e UC San Diego Ele é escrito na linguagem Java é possui uma interface gráfica denominada Vergil. O Kepler está disponível para os seguintes sistemas operacionais: Window, OSX e Linux. Através dele, os cientistas podem capturar *workflows* em um formato XML facilmente intercambiável, que pode ser arquivado, versionado, executado e reusado [90].

O Ptolemy II faz parte do projeto Ptolemy que faz estudos na modelagem, simulação, e concepção de sistemas concorrentes em tempo real. Seu foco é sobre os sistemas integrados que misturam tecnologias, incluindo, por exemplo, eletrônica analógica e digital, hardware e software, aparelhos e instrumentos mecânicos e eletrônicos. O foco está também em sistemas complexos, no sentido de que mistura muito diferentes operações, tais como ambiente de redes, processamento de sinais, controle *feedback*, processos de tomada de decisões [42,91].

O Ptolomeu II possui uma área onde os modelos são construídos visualmente como um conjunto de componentes que interagem entre si. Um modelo computacional rege a semântica da interação e, portanto, impõe uma disciplina sobre a interação dos componentes. O fluxo de dados contém tokens que são passados de um ator ao outro através das suas conexões. A semântica da interação do workflow não é determinada por atores, ela é definida por um componente separado chamado diretor [92].

Ele oferece uma infra-estrutura unificada para a implementação de uma série de modelos computacionais. A arquitetura global consiste em um conjunto de pacotes que fornecem suporte para os modelos computacionais mais comuns, como: bibliotecas matemáticas, algoritmos gráficos, interpretação de expressões lingüísticas, gráficos de sinais, interfaces multimídias, tais como áudio; e em um outro conjunto de pacotes mais especializados para modelos particulares, como representações de modelos graficamente.

O Kepler 1.0 é baseado no Ptolemy II 7.0.2. Em sua arquitetura, o Kepler herda do Ptolemy II o paradigma de modelagem orientada a ator, que separa os componentes, ditos atores, da orquestra geral do workflow, conduzida por diretores, tornando os

componentes mais facilmente reutilizáveis. Da mesma maneira, ele herda todos esses modelos computacionais do Ptolemy II descritos acima. Os *Workflows* podem ser organizados visualmente em sub-*workflows*. Cada um encapsula um conjunto de passos executáveis, que conceitualmente representam uma atividade do trabalho [93].

Utiliza a linguagem Modeling Markup Language (MoML) como modelo para os arquivos de *workflows* salvos pelo sistema. Esse formado é herdado do Ptolemy II, é corresponde a um formato Extensible Markup Language (XML). Nestes arquivos são especificados os componentes, suas respectivas parametrizações e as conexões entre eles que irão compor o workflow. Este modelo representa um mecanismo primário para construir modelos cuja definição e execução é distribuída em um ambiente de rede.

Uma vez que o Kepler fora desenvolvido em Java, é necessária a instalação de uma máquina virtual Java para ele ser executado na máquina no usuário. Os seus desenvolvedores aconselham a instalação do JDK 5 (jdk-1_5_0_15-windows-i586-p.exe) , que pode ser obtido no site oficial da Sun Microsystems.

O instalador do Kepler (kepler-win-1.0.0.exe) pode ser obtido no site oficial da ferramenta: http://www.kepler-project.org/. Sua instalação é praticamente automática e muito intuitiva. Abaixo estão listados os requisitos que compõem a configuração mínima e/ou recomendada para a sua instalação:

- 300 MB de espaço em disco;
- 512 MB de RAM(mínimo), 1 GB ou mais (recomendado);
- 2 GHz CPU (mínimo);
- Java 1.5.x (não usar Java 1.6)
- Ambiente de rede (opcional). Embora não represente um requisito para executar o Kepler, muitos *workflows* exigem uma conexão para acessar recursos remotos.
- Software R (opcional). R é uma linguagem e um ambiente para o processamento de dados estatísticos e gráficos. É necessário para algumas funcionalidades do Kepler.

A Figura 3.11 apresenta a interface do Kepler. Ela possui uma barra de menu, um barra de ferramentas, um árvore de componentes, um painel de referência e uma área de canvas onde será modelado o workflow.

A barra de menu contém as seguintes funcionalidades principais: abrir arquivos ou endereços de *workflows*; salvar, imprimir e fechar *workflows*; copiar, colar, apagar atores; desfazer ou refazer um comando; opções de zoom para melhor visualizar o workflow; ferramentas para configurar o ambiente do aplicativo; recursos para controlar o funcionamento do workflow e permite adicionar portas e multiportas ao workflow.

A barra de ferramentas contém as mesmas opções de zoom da barra de menu, bem como, os recursos para controlar o workflow. Os botões da barra de ferramentas são apresentados na Tabela 3.4.

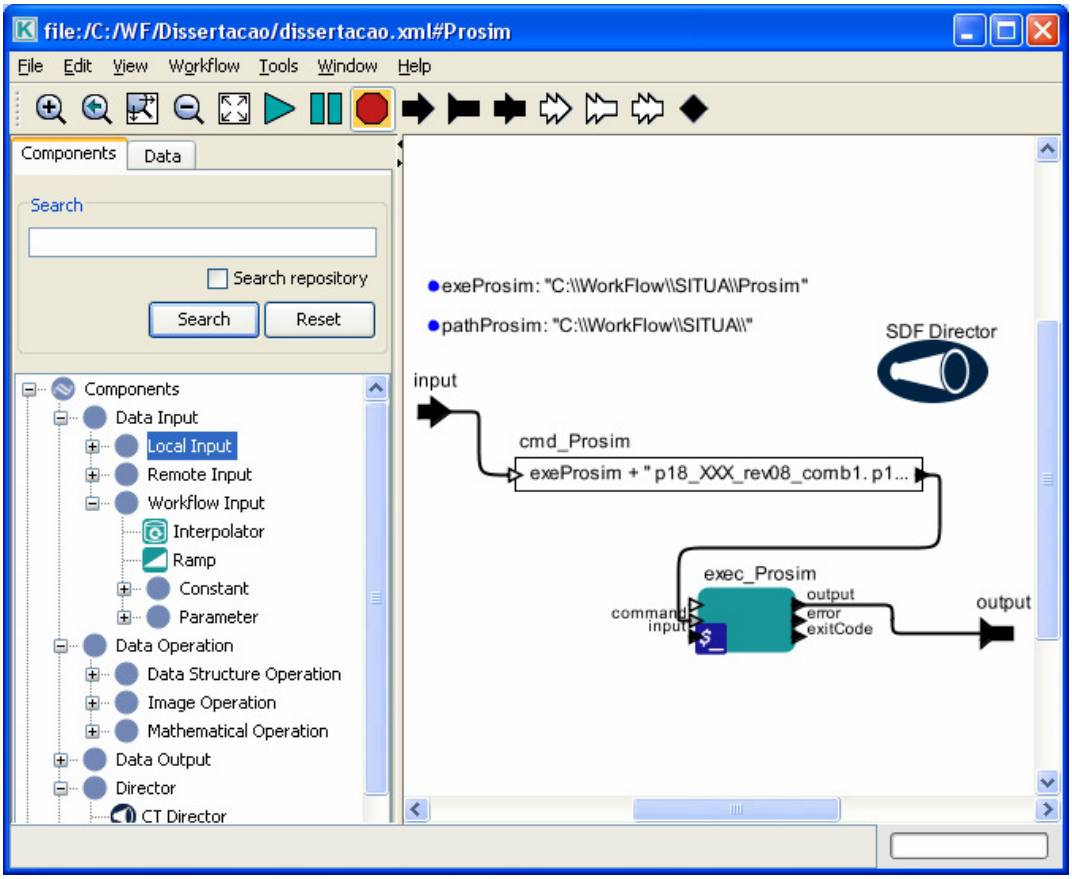


Figura 3.11: Interface do Kepler

Tabela 3.4: Barra de ferramentas do Kepler

Botão	Descrição
	Aumenta o zoom do canvas do workflow.
	Retorna o workflow ao zoom inicial.
	Define um zoom que ajusta todo o workflow na tela.
	Diminui o zoom do canvas do workflow.
	Apresenta a área do canvas em tela cheia no monitor.
	Inicia a execução de um workflow ou o reinicia, caso tenha sido paralisado.
	Pausa a execução de um workflow.
	Termina a execução de um workflow.
	Insere uma porta de entrada no workflow.
	Insere uma porta de saída no workflow.
	Insere uma porta de entrada/saída no workflow.
	Insere uma multiporta de entrada no workflow.
	Insere uma multiporta de saída no workflow.
	Insere uma multiporta de entrada/saída no workflow.
	Insere um ponto de junção entre conexões no workflow.

O canvas é uma interface gráfica onde os engenheiros podem construir os *workflows* através de sua interface gráfica intuitiva. Os componentes (atores) são arrastados para a área do diagrama do Workflow, onde podem ser conectados, configurados e executados.

Essa interface gráfica também pode ser utilizada pelos engenheiros para modelar os *workflows* antes de implementar o código e definir os parâmetros necessários para a execução. O Kepler possui um ator chamado Composite que representa um componente sem um papel (função) definido. Ele pode ser utilizado para apresentar informações básicas tais como nome e descrição tanto do próprio ator bem como de suas portas

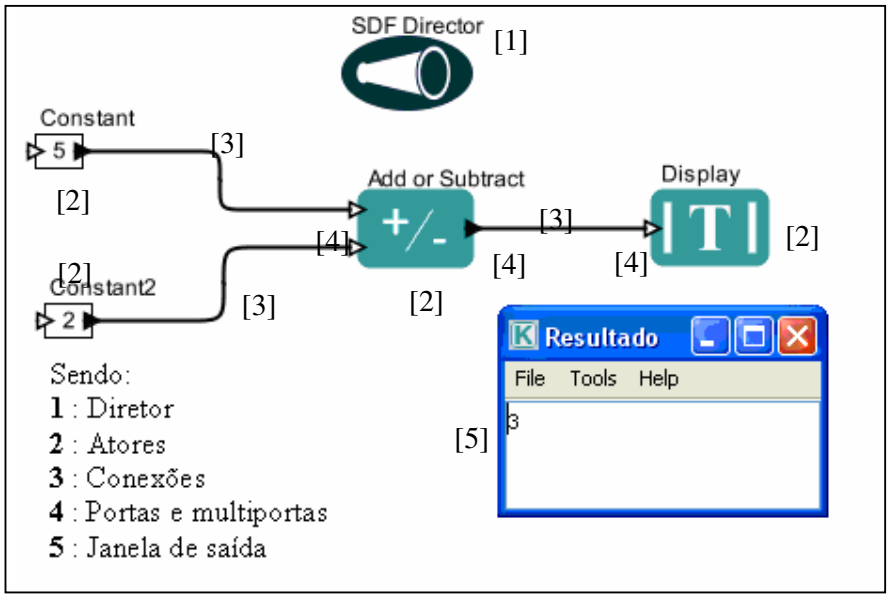


Figura 3.12: Componentes do Workflow

Os atores são disponibilizados em uma árvore de componentes que são agrupados de acordo com os papéis que desempenham no workflow, por exemplo, há os diretores, componentes de entrada de dados, de saída de dados, estruturas de controle, funções matemáticas, acesso remoto, etc. O Kepler também disponibiliza uma ferramenta de busca para encontrar os atores nessa árvore.

A Figura 3.12 ilustra um simples exemplo de uma operação de soma. Nele estão representados os principais elementos do workflow: diretores; atores; as conexões entre os atores; portas e multiportas; e uma janela de saída textual com o resultado do processamento.

O Kepler faz uma analogia do workflow com um filme de cinema. Ambos são visualmente representados pelos seus componentes: diretores e atores. O diretor controla (ou dirige) a execução do workflow e os atores realizam o processamento de acordo com as diretivas do diretor, assim como um diretor de um filme supervisiona o elenco e o suporte [37]. Os atores encenam sua atuação seguindo as instruções do diretor. Em outras palavras, os atores especificam o que processar, enquanto que o diretor define quando e como este processamento ocorrerá.

Em todo workflow científico deve haver um diretor que controlará a sua execução usando um modelo computacional característico. Por exemplo, a execução do workflow pode ser síncrona, com um processamento ocorrendo em um componente de cada vez, em uma seqüência predefinida. De outra maneira, os componentes do

workflow também podem ser executados em paralelo, com um ou mais componentes executando simultaneamente. A Figura 3.12 apresenta os diretores disponíveis no Kepler, e suas respectivas descrições para utilização.

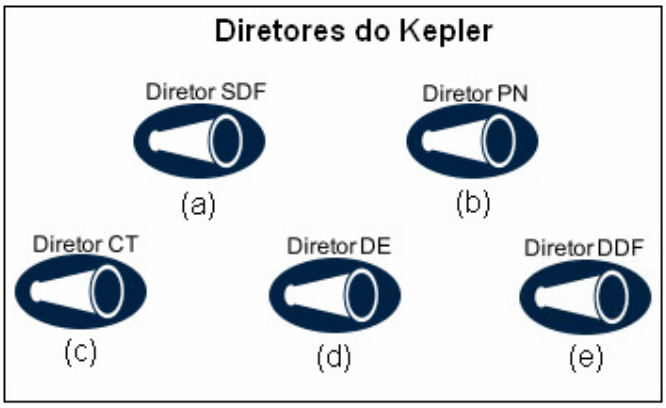


Figura 3.13: Diretores do Kepler

O diretor SDF (Figura 3.13 (a)) define que a comunicação entre os atores será feita de modo síncrono, eles enviam e recebem tokens, que indicam quando eles serão disparados. Ele é freqüentemente usado para supervisionar *workflows* bem simples e seqüenciais, que podem realizar, por exemplo, processamento, conversão ou formatação de dados.

O diretor PN (Figura 3.13 (b)) permite que cada ator execute um *thread* ou um processo de modo isolado. As conexões podem conter uma fila de tokens em espera para disparar um determinado ator. Ele é usado para gerenciar *workflows* que requerem processamento paralelo em sistemas distribuídos.

O diretor CT (Figura 3.13 (c)) define que as conexões entre os atores representam o valor de um sinal contínuo no tempo em um determinado momento. Em cada ponto, os atores calculam a sua saída com base nas entradas anteriores e na atual entrada, até que o sistema se estabiliza. Muitas vezes são utilizados para modelar processos físicos dinâmicos.

O diretor DE (Figura 3.13 (d)) define que os atores se comunicam através de uma fila de eventos dependentes do tempo. Os eventos são processados numa escala de tempo global, e em resposta a um evento, o ator é autorizado a chamar eventos no presente ou no futuro, mas não no passado. Ele é utilizado em sistemas de filas, comunicações em rede, em modelos assíncronos de circuitos e em reações instantânea de sistemas físicos.

O diretor DDF (Figura 3.13 (e)) é freqüentemente usado para *workflows* que utilizem loops ou condições particulares ou outras estruturas de controle, mas que não contenham processamento paralelo.

É possível ter mais de um diretor dentro de um workflow, porém eles devem estar alocados em sub-*workflows* diferentes. O ator CompositeActor (Figura 3.14 (a)) é o responsável por criar sub-*workflows* dentro de um workflow. Caso um workflow tenha muitas atividades que envolvam a participação de muitos atores, é aconselhável utilizar o CompositeActor para encapsulá-los, e, desta forma, melhor organizar visualmente o workflow, tornando seu layout mais claro para o usuário do Kepler.

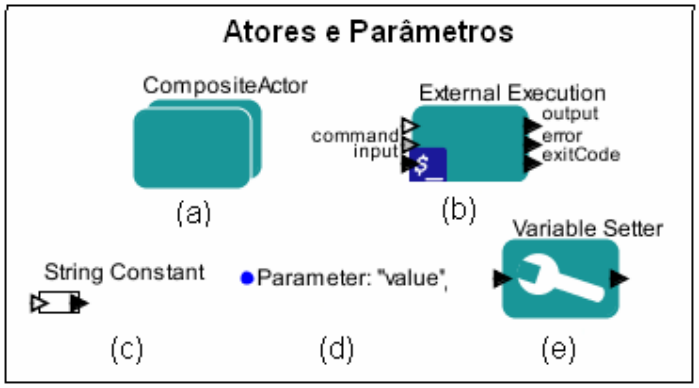


Figura 3.14: Atores e parâmetros do Kepler

A base principal para o funcionamento do Kepler é a utilização dos atores que são responsáveis por encapsular as atividades do workflow. Na maioria dos casos, essas atividades são exercidas por programas específicos. O ator External Execution (Figura 3.14 (b)) é capaz de executar programas iniciados pela linha de comando. Continuando, os atores analisam os dados de entrada, executam um processamento e fornecem uma saída. Esses dados de entrada podem ser transferidos internamente durante a execução do workflow ou podem estar armazenados em arquivos. Neste caso, o Kepler possui alguns atores que fazem leitura de arquivos, dentre eles podemos destacar o FileReader (Figura 3.15(a)). Da mesma maneira, os resultados produzidos pelos processamentos podem ser repassados internamente pelo workflow e servir de entrada para a atividade seguinte, ou ele podem gerar um saída. Neste caso o resultado pode ser armazenado em arquivo pelo ator FileWriter (Figura 3.15(b))ou pode se exibido na tela. Caso o resultado seja um texto, ele pode ser exibido pelo ator Display (Figura 3.15(c)); caso ele seja um imagem, pode ser exibido pelo ator ImageDisplay (Figura 3.15(d)).

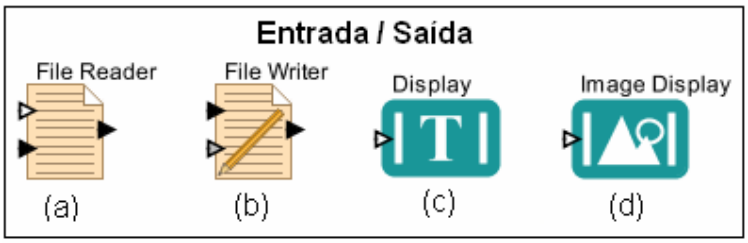


Figura 3.15: Atores de entrada e saída de dados

O Kepler também possui atores que controlam o fluxo de execução do workflow de acordo com os valores dos parâmetros que são passados ao longo do processamento. Nesse sentido, há atores que realizam comparações aritméticas (Figura 3.16 (a, b)) e lógicas (Figura 3.16 (c)), atores que geram iterações no fluxo (Figura 3.16 (d)), e atores que definem o fluxo a ser seguido a partir de uma condição (Figura 3.16 (e, f)).

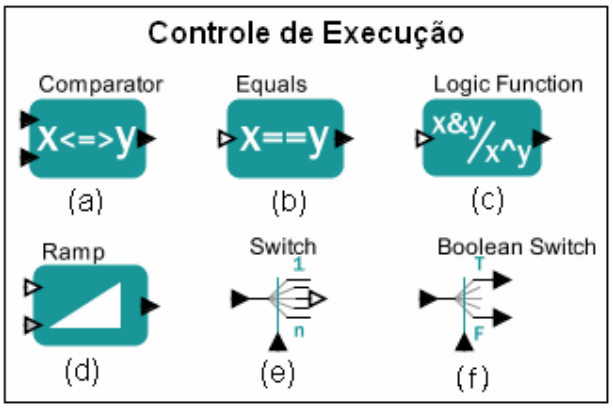


Figura 3.16: Atores de controe de execução

Durante a execução do workflow pode haver a necessidade de formatar os parâmetros envolvidos e as saídas das atividades. O Kepler possui atores que são capazes de manipular valores numéricos e strings. A Figura 3.6 apresenta alguns atores que realizam essas manipulações. O papel desempenhado por esses atores geralmente representam funções existentes na maioria das linguagens de programação mais utilizadas no desenvolvimento de sistemas, por exemplo: converter os caracteres para maiúsculo ou minúsculo (Figura 3.17 (a)), obter o comprimento de uma string (Figura 3.17 (b)), extrair uma parte da string (Figura 3.17 (c)), converter uma string para inteiro (Figura 3.17 (d)) ou XML (Figura 3.17 (e)).

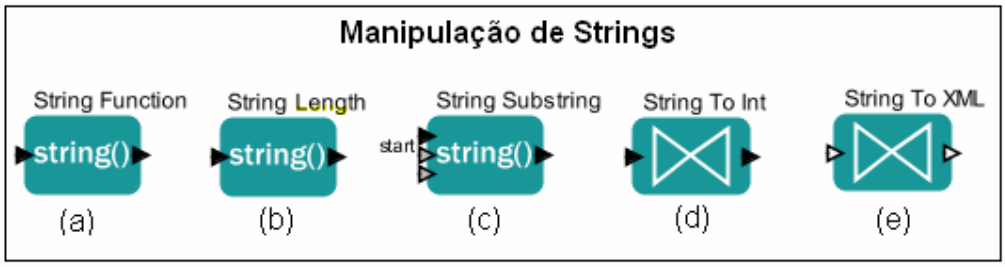


Figura 3.17: Atores de manipulação de strings

Os atores e o próprio workflow contém parâmetros (Figura 3.14 (d)) que são valores configuráveis, que podem, por exemplo, controlar aspectos da simulação. Os valores desses parâmetros podem ser alterados durante a execução do workflow, para isso é necessário defini-los no ator VariableSetter (Figura 3.14 (e)). Os parâmetros dos atores podem ser pré-definidos por constantes (Figura 3.14 (c)). Cada ator pode conter uma ou mais portas usadas para consumir ou produzir dados e permitir a comunicação com os outros atores no workflow. A linha que representa o fluxo de dados entre as portas dos atores é denominada de canal. As portas podem ser divididas em três tipos: portas de entrada, para os dados consumidos pelo ator; portas de saída, para os dados produzidos pelo ator; e porta de entrada/saída, para os dados que são consumidos e produzidos pelo ator. Cada porta pode ser configurada para ser singular ou múltipla. A porta singular é conectada a apenas um canal, é representada por um triângulo preto. A porta múltipla pode ser conectada a muitos canais, é representada por um triângulo branco. Os dados envolvidos nessa comunicação são heterogêneos e podem ser recebidos e enviados para saídas em arquivos, saídas gráficas, boxplots, FTP, SSH, páginas web e bancos de dados.

O Kepler contém atores específicos para tarefas de acesso remoto a dados, transformações de dados, análise de dados, interface com outras aplicações, por exemplo, atores que executam funções do Matlab e do R (RExpression, Correlation, RMean, RMedian, etc) estão contidos na sua biblioteca padrão. Uma importante característica do Kepler é a possibilidade de acesso a bases de dados científicas. A biblioteca de componentes do Kepler contém uma *Ecological Metadata Language* (EML) representada pelo ator EML2Dataset utilizada para acessar fontes de dados com informações ecológicas, desta forma permite ao Kepler importar uma grande quantidade de dados heterogêneos, o que representa uma ferramenta muito útil e flexível para os engenheiros que utilizam muitos dados em formatos de arquivos diferentes.

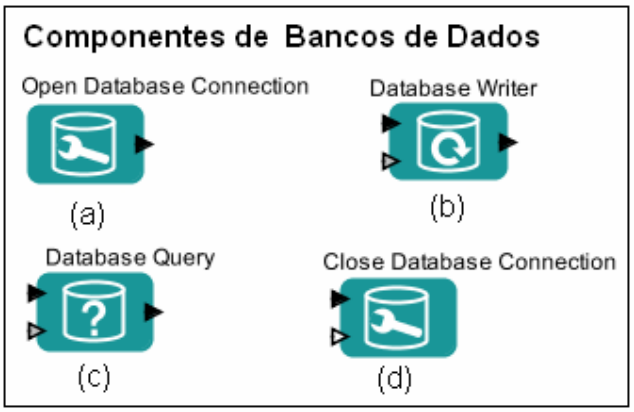


Figura 3.18: Componentes de acesso a bancos de dados

Atualmente a principal fonte de dados utilizada pelos sistemas computacionais são os bancos de dados. O Kepler não poderia deixar de possuir uma biblioteca de acesso e consulta aos bancos de dados. Os componentes de consulta possuem um parâmetro que corresponde à instrução SQL que realizará uma operação no banco.

O Open Database Connection (Figura 3.18 (a)) é o ator utilizado para abrir uma conexão ao banco de dados. Devem ser definidos quatro parâmetros de conexão: o sistema de gerenciador de banco de dados, o endereço do banco, o usuário e a sua respectiva senha. Sua porta de saída contém a referência para a conexão ao banco, que uma vez estabelecida, pode ser utilizada pelos outros componentes de acesso a banco de dados utilizados no workflow.

O Database Writer (Figura 3.18 (b)) é o ator que executa um comando que realiza uma operação no banco de dados. Possui como parâmetros de entrada a referência da conexão aberta pelo ator OpenDatabaseConnection e o comando SQL. A porta de saída exibirá a quantidade de linhas afetadas pelo comando ou um erro caso o comando esteja incorreto.

O ator Database Query (Figura 3.18 (c)) é o ator que executa um comando que realiza uma consulta no banco de dados. Os parâmetros de entrada são: a referência da conexão aberta pelo ator OpenDatabaseConnection; o comando SQL; e o formato do resultado da consulta, que pode ser:XML, *record, array, string*

O ator Close Database Connection (Figura 3.18 (d)) desconecta a conexão aberta pelo ator OpenDatabaseConnection. Os parâmetros de entrada são: a referência para a conexão; e um *trigger* (gatilho), não obrigatório, porém quando conectado, é responsável pelo disparo do ator quando recebe um *token*.

As bibliotecas de componentes do Kepler são formadas por objetos configuráveis. Utilizando características de orientação a objetos, os usuários podem criar suas próprias bibliotecas de atores, herdando as características básicas dos atores mais genéricos e estendendo suas funcionalidades. Os usuários também podem fazer download de novos atores dos repositórios do Kepler, assim como podem enviar os atores que eles próprios desenvolveram para estes repositórios. Estes novos atores serão adicionados na árvore de componentes.

Os desenvolvedores do Kepler estão trabalhando numa biblioteca para acessar novas fontes de dados, os web services e Grids: Estes atores irão permitir aos engenheiros utilizar recursos computacionais na web e em *workflows* científicos distribuídos. O Kepler ainda possuirá atores especializados para executar jobs em grids, atores para certificação de autenticação (SProxy or GlobusProxy), para enviar uma job para um Grid (GlobusJob), e um acesso a dados baseado em Grid (GridFTP).

Capítulo 4 - Especificação do workflow

4.1 Ferramentas Utilizadas na Análise de Fadiga

O workflow que será implementado tem como objetivo realizar a Análise de Fadiga de *Risers*, considerando a metodologia híbrida de projeto de *risers* descrita na seção 2.2.3 e a sequência de análises descrita na seção 2.3.

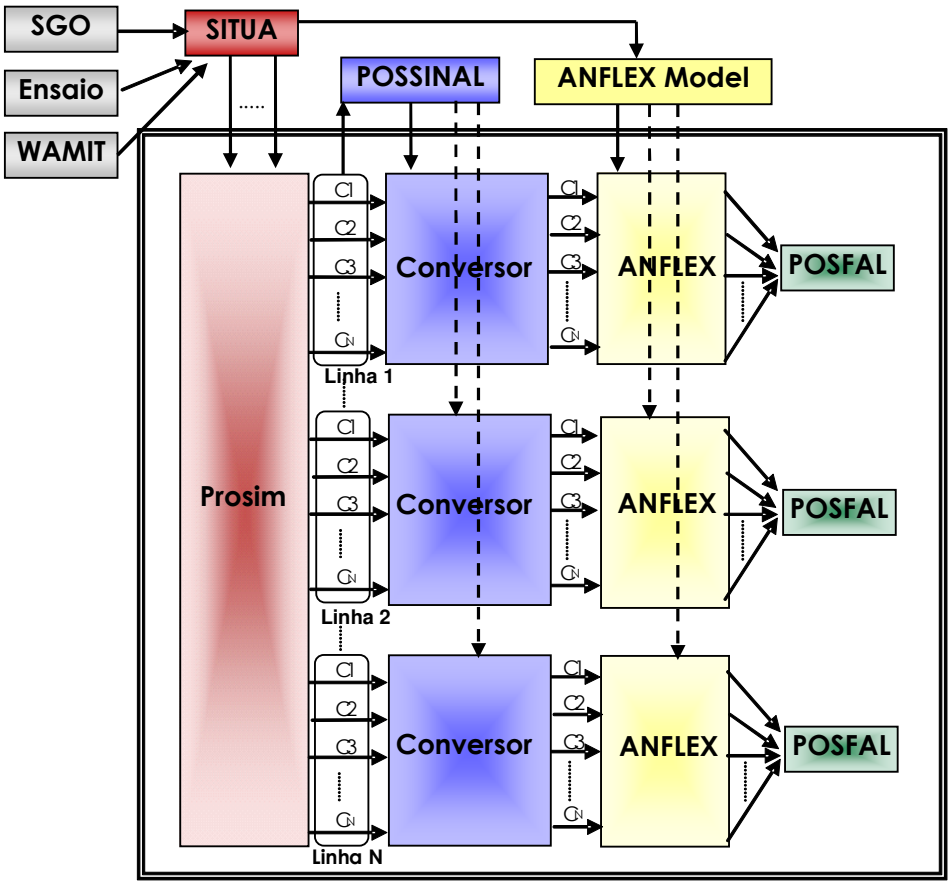


Figura 4.1: Ferramentas utilizadas na Análise de Fadiga dos Risers

Recapitulando, em uma primeira etapa, a análise de movimentos acoplada do conjunto (casco + linhas de ancoragens + *risers*) é realizada aplicando-se os carregamentos ambientais correspondentes à operação normal da plataforma durante um determinado período de tempo.

Na segunda etapa, na análise estrutural desacoplada de cada *riser*, as séries de movimento obtidas na primeira etapa são aplicadas no topo de cada linha em conjunto com os carregamentos ambientais correspondentes (correnteza e onda).

Como produto da análise estrutural desacoplada do *riser* na segunda etapa, para posterior análise de fadiga, são obtidos os histogramas de amplitude de variações de tensões das juntas críticas ao longo do *riser*. Finalmente, numa terceira etapa, a partir dos histogramas de amplitudes de variaçãoes de tensões, avalia-se a vida útil do *riser* em questão através da análise da fadiga.

Desta forma, até chegar à determinação da vida à fadiga propriamente dita, que é efetuada através do programa POSFAL, existe uma série de outras etapas e ferramentas de análise que geram os resultados e informações necessárias. Além disso, em cada uma destas etapas diferentes ferramentas poderiam ser empregadas. A Figura 4.1 ilustra uma das seqüências possíveis, representando a dependência entre as ferramentas envolvidas no processo de avaliação da vida útil através das análises de fadiga dos *risers*.

Esta seqüência se inicia com o programa WAMIT, que é empregado para gerar coeficientes hidrodinâmicos do casco da plataforma flutuante ao qual os *risers* estão conectados. A seqüência envolve também a determinação de coeficientes de força de vento e de arrasto/correnteza no casco, usualmente efetuada através de Ensaios em túnel de vento, bem como a utilização de dados do fundo do mar (batimetria e obstáculos), obtidos a partir do banco de dados SGO (Sistema de Gerenciamento de Obstáculos) da Petrobras.

Na implementação preliminar apresentada aqui, esses sistemas não estarão incorporados no gerenciamento do workflow científico, mas poderão ser incluídos em desenvolvimentos futuros. Os componentes que estão sendo considerados atualmente são os contidos dentro do quadro indicado pela linha dupla na Figura 4.1: Prosim, Conversor, ANFLEX e POSFAL. A Figura 4.1 ainda inclui outras ferramentas que não estão formalmente inseridas dentro do workflow uma vez que são ferramentas com interface gráfica, portanto não podem ser disparadas e executadas automaticamente. Trata-se do SITUA, que faz parte do sistema SITUA-Prosim; o ANFLEX Model, pré-processador do ANFLEX, e do POSSINAL, que pode tratar séries temporais fornecidas tanto pelo Prosim quanto pelo ANFLEX para efetuar o tratamento estatístico dos sinais de resposta.

É interessante ressaltar que a Figura 4.1 ilustra apenas uma das seqüências possíveis: por exemplo, ao invés do WAMIT poderia ser empregado outro programa de análise hidrodinâmica, por exemplo, o Wadam; ao invés do Prosim poderia ser empregado o TPN, Dynasim ou DeepC e assim por diante.

O programa Prosim utiliza como dados de entrada os dados de obstáculos e batimetria de fundo do SGO; os coeficientes gerados pelo WAMIT, SGO e Ensaio, e que foram convertidos no SITUA para a análise acoplada de movimentos da plataforma; e os coeficientes de vento e correnteza de ensaios, para o modelo acoplado com todas as linhas e considerando cada caso de condição ambiental.

Em seguida, cada arquivo de movimentos da unidade flutuante gerado no Prosim é tratado em um programa Conversor, que os converte para o formato de leitura do ANFLEX. O Conversor também separa o arquivo de movimentos em outros seis novos arquivos, podendo separar as parcelas de baixas freqüências e altas freqüências, cada um contendo a série de movimentos de uma determinada direção de movimento da unidade flutuante: *Surge, Sway, Heave, Yaw, Pitch* e *Roll*.

O workflow prossegue com o programa ANFLEX para executar a análise estrutural detalhada de cada *riser* de interesse (Linha 1, Linha 2 ... Linha N), para cada caso de carregamento ambiental. São aplicados os movimentos calculados pelo Prosim e tratados pelo Conversor quanto os carregamentos de onda e correnteza aplicados diretamente sobre as linhas. Finalmente, a resposta estrutural do ANFLEX, para um conjunto de condições ambientais de carregamento, serve de entrada para a análise da fadiga dos *risers* com a determinação da vida útil dos *risers* pelo POSFAL. A Figura 4.2 apresenta somente as ferramentas da análise de fadiga que serão implementadas no workflow científico.

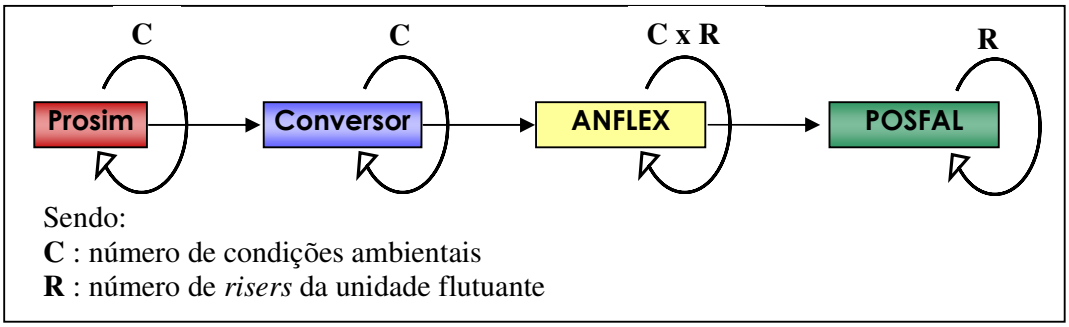


Figura 4.2: Ferramentas utilizadas no workflow científico

A seguir descrevem-se as diferentes fases da seqüência de análises do workflow apresentadas na Figura 4.1.

4.1.1 WAMIT: Análise hidrodinâmica do casco da unidade flutuante

O WAMIT é um programa que analisa o comportamento hidrodinâmica dos sistemas *offshore* na sob a ação de ondas utilizando uma metodologia desacoplada. Ele calcula a velocidade potencial e a pressão que as ondas exercem sobre o casco através de análises de difração, que calculam os efeitos das ondas incidentes sobre o casco. Os resultados obtidos são parâmetros hidrodinâmicos incluindo massa adicional, coeficientes de amortecimento, momentos das forças e *response-amplitude operators* (RAOs).

Em seguida, os resultados do WAMIT são empregados no programa SITUA/Prosim para a análise de movimentos acoplada do sistema.

4.1.2 SITUA: Geração dos Modelos Numéricos

O sistema SITUA-Prosim engloba uma interface gráfica (SITUA) e um programa de análise (Prosim). O SITUA incorpora um conjunto de ferramentas gráficas e numéricas para pré-processamento e geração de modelos, incluindo a leitura de dados batimétricos e de obstáculos de fundo oriundos do SGO, leitura de dados meteoceanográficos para definição de carregamentos ambientais, e leitura dos coeficientes de ondas gerados pelo WAMIT. Por sua vez, o Prosim é o programa que executa as análises dinâmicas não-lineares com os modelos gerados pelo SITUA. A Figura 4.3 ilustra o modelo tridimensional de uma unidade flutuante semi-submersível gerada no SITUA pela opção híbrida, com a modelagem através de cilindros equivalentes ao casco real.

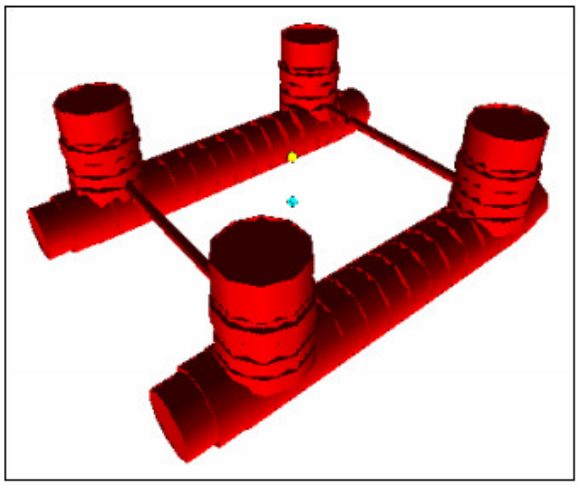


Figura 4.3: Unidade Flutuante modelada no SITUA

Apesar de o usuário normalmente enxergar o SITUA-Prosim como um sistema único, no contexto do workflow apresentado na Figura 4.3, a interface gráfica do SITUA está representada separadamente dos módulos de análise do Prosim. Evidentemente, como a interface interativa não pode ser utilizada através de linha de comando, sua execução não é controlada pelo workflow. No entanto, no contexto da atual implementação preliminar, a primeira aplicação a ser executada pelo usuário é a interface gráfica interativa do SITUA, com a qual o usuário poderá realizar todos os procedimentos necessários à geração dos modelos numéricos e dos dados geométricos e físicos que descrevem a unidade flutuante, as linhas de ancoragem e *risers*. Além disso, o SITUA também pode ser empregado para exportar os modelos dos *risers* no formato dos arquivos de entrada lidos pelo ANFLEX.

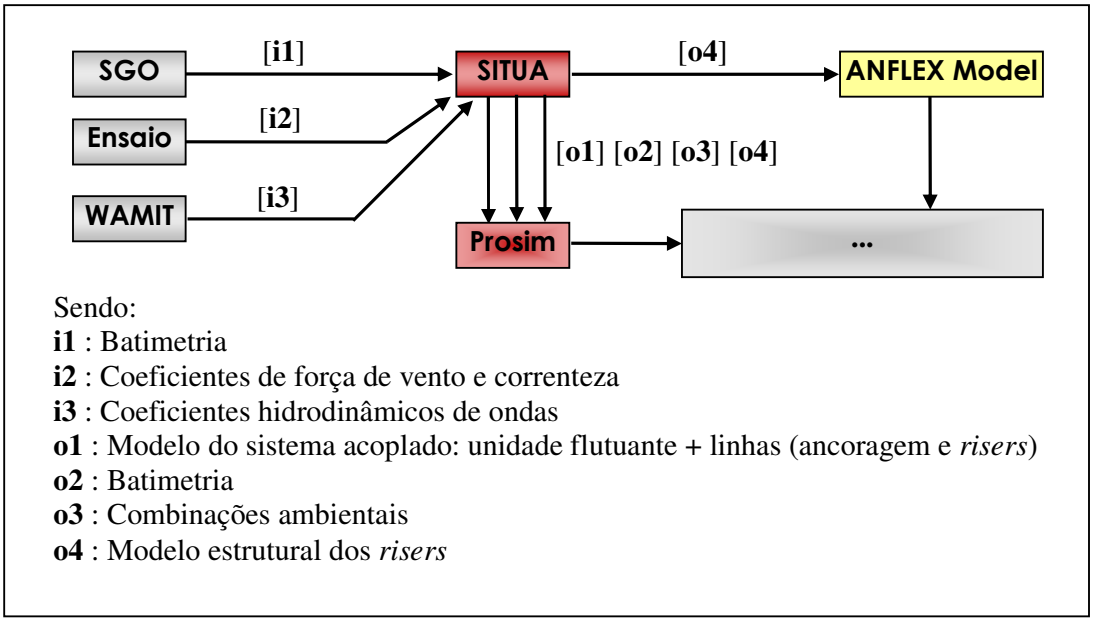


Figura 4.4: Ênfase no papel do SITUA no workflow

O diagrama da Figura 4.4 ilustra uma visão do workflow do diagrama da Figura 4.1, dando destaque ao papel desempenhado pelo SITUA, mostrando suas entradas e saídas, assim como os outros programas com os quais interage diretamente. Os dados de batimetria do SGO estão contidos no arquivo com extensão '**dgn'**. Os coeficientes de onda do WAMIT estão contidos no arquivo com extensão '**out**', convertido pelo SITUA em um outro formato com extensão '**wnf**' para leitura no Prosim. O arquivo contendo o modelo do sistema acoplado possui extensão '**prm**', estando definidas as condições ambientais sob as quais os movimentos do sistema serão calculados no Prosim. O componente Workflow do diagrama representa uma abstração das outras ferramentas

que também constituem o workflow de análise de fadiga, mas que não se comunicam com o SITUA.

4.1.3 Prosim: Análise Acoplada de Movimentos da Plataforma

Lembrando que nesta etapa preliminar não está sendo considerada a fase de análise hidrodinâmica através do WAMIT, o primeiro aplicativo a ser inserido no workflow é o Prosim. Como já mencionado, este programa é o responsável por realizar as análises acopladas de movimentos.

O Prosim emprega uma formulação acoplada que incorpora, em uma única estrutura de código e de dados, um modelo hidrodinâmico para a representação do casco da unidade flutuante, e um modelo estrutural e hidrodinâmicos com elementos finitos para a representação rigorosa das linhas de ancoragem e *risers*. A cada instante do processo de integração no tempo das equações de movimento do casco, o programa efetua passos da análise não-linear dinâmica das linhas. Como resultado desta formulação acoplada, o programa é capaz de fornecer os movimentos da unidade flutuante considernado a contribuição das linhas e todos os efeitos não-lineares e dinâmicos decorrentes da interação entre o casco e as linhas.

No contexto do projeto de fadiga de *risers*, o Prosim pode ser empregado para gerar as séries temporais de movimentos da unidade flutuante, quando o sistema é submetido às várias condições ambientais especificadas.

As informações geradas pelo WAMIT e pelo SGO foram transformadas no SITUA, que gerou novos formatos de arquivos para que fossem lidos no Prosim. Assim, os coeficientes hidrodinâmicos oriundos do WAMIT agora estão contidos em arquivos com extensão "**wnf**". Os dados de fundo batimétrico e obstáculos oriundos do SGO agora estão contidos em arquivos com extensão "**fun**", e dados de coeficientes de vento e arrasto do casco, determinados por ensaios, devem estar contidos em arquivos respectivamente com extensão "**cvs**" e "**cds**".

Para realizar seu processamento, o Prosim recebe esses arquivos descritos acima e também interpreta um arquivo com extensão "**prm**" gerado pela interface gráfica SITUA, contendo os dados gerais do modelo do casco da unidade flutuante e das linhas.

Cada simulação/análise no Prosim, correspondente a um caso de carregamento, que representa combinações ambientais de vento, onda e correntes, e como produtos de cada simulação gera diversos tipos de arquivos contendo resultados de movimentos de

tração das linhas, dentre outros. No contexto da análise de fadiga de *risers* considerado aqui, o resultado de interesse são os históricos de movimentos da unidade flutuante, que ficam armazenados em arquivos de saída com extensão "**pld**", para cada caso de carregamento. Desta forma, o número de arquivos com esta extensão corresponde ao número de casos de carregamento avaliado no projeto.

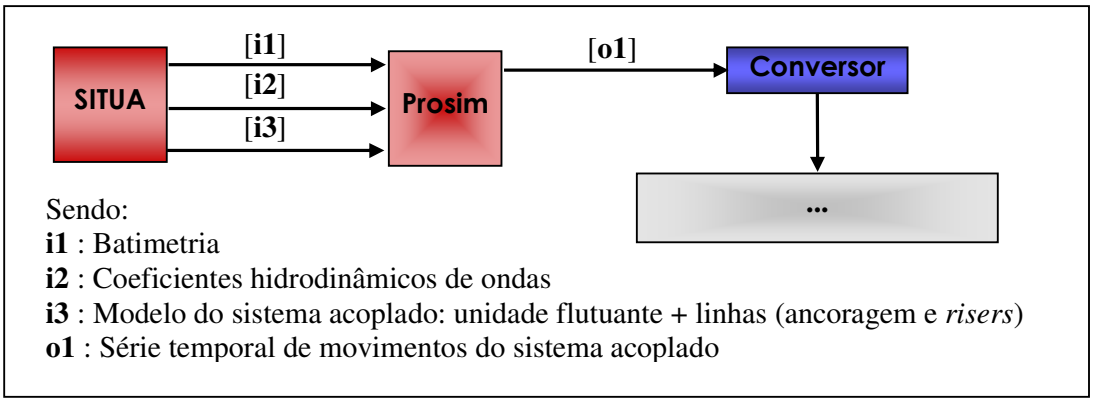


Figura 4.5: Ênfase no papel do Prosim no workflow

O diagrama da Figura 4.5 também ilustra uma visão do workflow onde é dado destaque ao papel desempenhado pelo Prosim, mostrando suas entradas e saídas, assim como sua interação com os seguintes programas: WAMIT, SITUA e Conversor.

4.1.4 POSSINAL: Tratamento do Sinal de Resposta de Movimentos

Opcionalmente, a critério do projetista, o histórico de movimentos do casco da unidade flutuante pode passar por um processo de filtragem dos termos de baixa freqüência nas componentes de *surge* e *sway* e, caso tiverem, *rool e pitch*. Para tal, utiliza-se o programa POSSINAL, como indicado na Figura 4.5. Neste processo, as entradas do POSSINAL seriam os arquivos de extensão "**pld"** do Prosim; as saídas seriam arquivos semelhantes, contendo os sinais filtrados de baixa freqüência.

É interessante observar que apenas a versão "batch" do POSSINAL (ou de qualquer outro programa equivalente capaz de efetuar o tratamento estatístico de sinais e disparado por linha de comando) pode ser inserida formalmente no workflow; a versão disparada através de interface gráfica não pode ser gerenciada pela ferramenta de workflow.

O diagrama da Figura 4.6 ilustra uma visão do workflow onde é dado destaque ao papel desempenhado pelo POSSINAL, mostrando suas entradas e saídas, assim como sua interação com o Prosim e com o Conversor.

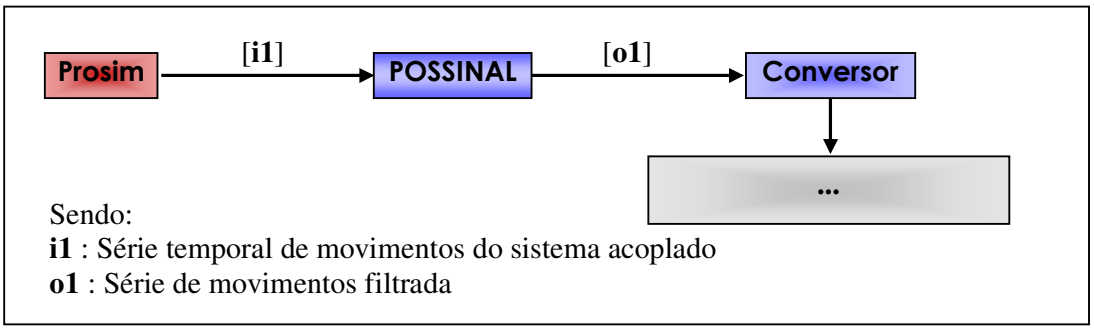


Figura 4.6: Ênfase no papel do POSSINAL no workflow

4.1.5 ANFLEX: Análise Estrutural dos *Risers*

A fase seguinte do processo do workflow consiste no uso do programa ANFLEX para executar a análise estrutural de cada *riser*, considerando a ação de carregamentos ambientais de ondas e corrente, e a série temporal dos movimentos da unidade flutuante, calculadas pelo Prosim, tratados pelo Conversor e que podem ter sido filtrados no POSSINAL.

Atualmente, o ANFLEX não é capaz de importar os dados de movimento a partir de arquivos com extensão "**pld**" gerados por uma simulação no Prosim. Para contornar dessa incompatibilidade de comunicação entres as aplicações Prosim e ANFLEX, foi desenvolvido um software denominado Conversor, indicado na Figura 4.7, para converter os arquivos "**pld**" para o formato atualmente adotado para a entrada de dados de movimento pelo ANFLEX. Assim, a partir de cada arquivo "pld", esse Conversor gera outros seis arquivos com extensão "**thf**", cada um com uma das seis componentes de movimento da unidade flutuante: *surge, sway, heave, roll, pitch e yaw*.

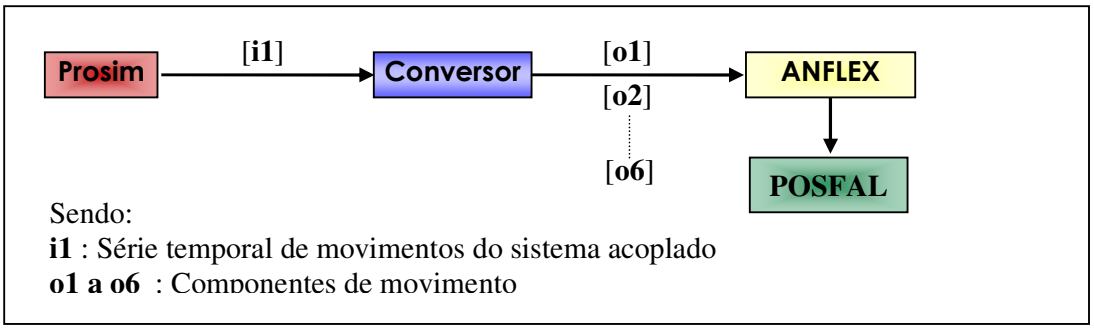


Figura 4.7: Ênfase no papel do Conversor no workflow

O diagrama da Figura 4.7 ilustra uma visão do workflow onde é dado destaque ao papel desempenhado pelo Conversor, mostrando suas entradas e saídas, assim como sua interação com o Prosim e o ANFLEX. A tendência é que futuramente o Conversor

seja eliminado e que esses dois programas se comuniquem de modo inteligente com o Prosim adequando seus dados de saída para a leitura no ANFLEX, ou o ANFLEX atualizar seu modo de leitura para o formato do Prosim. Atualmente encontra-se em andamento, no contexto da Rede Temática de Computacao e Visualizacao Científica – Galileu, gerenciada pela Petrobras, um projeto de implementação de um padrão de arquivo único para comunicação entre todos os simuladores baseado no formato XML. Com isso, futuramente não será mais necessário empregar este programa Conversor.

O XML é um formato para a criação de documentos com dados organizados de forma hierárquica. Ele tem sido cada vez mais utilizado no desenvolvimento e integração de sistemas principalmente pela sua portabilidade, já que é um formato que não depende das plataformas de hardware ou de software. A sua filosofia parte do princípio que a semântica e o valor dos dados devem estar claramente separados. A apresentação de seu conteúdo pode ser caracterizada pela simplicidade e legibilidade, tanto para humanos quanto para computadores.

Caso a opção de análise pelo ANFLEX seja a aplicação apenas dos componentes de baixa freqüência dos movimentos, filtrados pelo POSSINAL, não haverá necessidade de usar este programa conversor porque o próprio POSSINAL é capaz de ler as séries de movimento no formado "**pld**" do Prosim, e gravar o resultado da filtragem no formato de entrada do ANFLEX.

Entretanto, antes do ANFLEX é necessário executar o seu pré-processador, o ANFLEX Model. Ele também é um programa com interface gráfica, portanto não pode ser utilizado através de linha de comando. Nele é realizada uma modelagem desacoplada dos *risers*. Os arquivos de saída resultantes do seu processamento apresentam os esforços a que o *risers* estão submetidos no acoplamento com o casco nas seguintes situações: estática (**[arquivo]_S.dat**) e dinâmica (**[arquivo]_D.dat**).

Existe ainda a possibilidade do ANFLEX apresentar como saída para o POSFAL os histogramas de amplitude de variação de tensões no lugar dos esforços. Isto faz com que o ANFLEX exporte um conjunto de dados menor para o POSFAL.

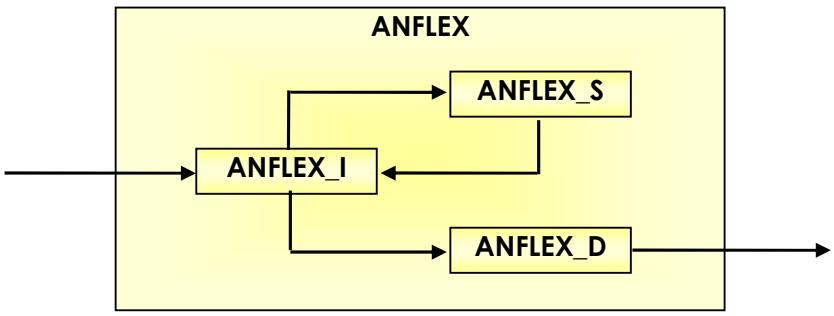
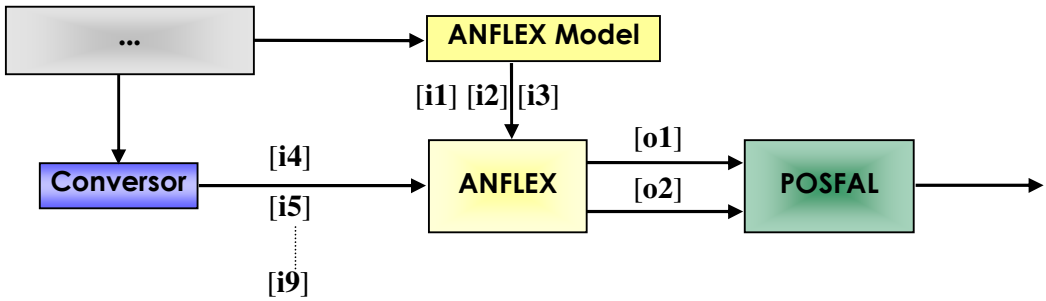


Figura 4.8: Componentes do ANFLEX

O ANFLEX é disparado para cada *riser* de interesse e para cada combinação ambiental, efetuando a análise estrutural através de um modelo de elementos finitos mais refinado do que o utilizado na análise de movimentos acoplada pelo Prosim. Ele é constituído de três programas: ANFLEX_I, ANFLEX_S e ANFLEX_D que, respectivamente, são responsáveis pelo carregamento dos dados de entrada, análise estática e análise dinâmica dos *risers*. Primeiramente o ANFLEX_I é executado e verifica se os dados estão corretos para a execução do ANFLEX_S. Após a execução do ANFLEX_S, o ANFLEX_I é chamado novamente, porém, dessa vez ele verifica se os dados estão corretos para a execução do ANFLEX_D. Então, a última atividade desempenhada no workflow será feita no POSFAL. A Figura 4.8 detalha todo esse processamento dos programas do ANFLEX.



Sendo:
i1 : Modelo do *riser*
i2 : Parâmetros da análise estrutural estática
i3 : Parâmetros da análise estrutural dinâmica
i4 a i9 : Componentes de movimento
o1 : Séries de Esforços (FAA) ou Histogramas de amplitude de Variação de Tensões (FAH)
o2 :Arquivo com as juntas críticas selecionadas para a análise de fadiga.

Figura 4.9: Ênfase no papel do ANFLEX no workflow

Os principais resultados do cálculo realizado pelo ANFLEX a serem fornecidos para o POSFAL são esforços (arquivos com extensão **'FAA'**) ou os histogramas de das amplitudes de variação de tensões (arquivos com extensão '**FAH'**) aos quais cada *riser* é submetido ao longo do tempo. O arquivo principal gerado para a posterior análise de fadiga corresponde ao de extensão "fah", contendo um histograma de tensões em alguns pontos de cada seção referente a posições específicas ao longo do comprimento do *riser*. O diagrama da Figura 4.9 ilustra uma visão do workflow onde é dado destaque ao papel desempenhado pelo ANFLEX, mostrando suas entradas e saídas, assim como sua interação com o Conversor,ANFLEX Model e POSFAL.

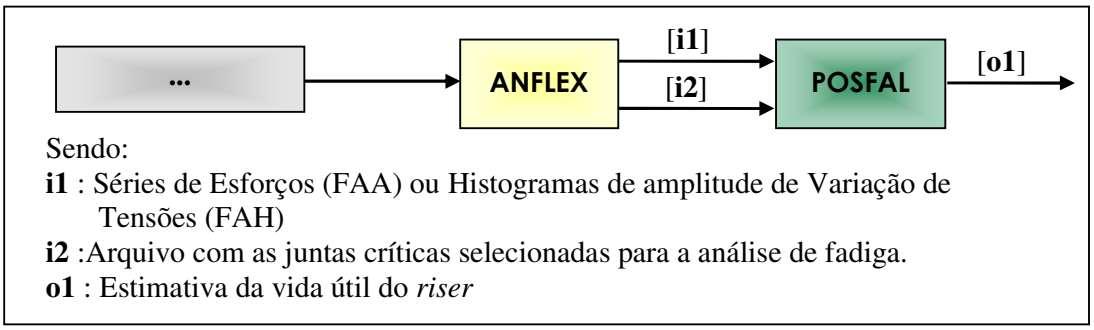
4.1.6 POSFAL: Análise de Fadiga

Como mencionado anteriormente, a análise de fadiga propriamente dita de cada *riser* é efetuada através do programa POSFAL. Para calcular o dano acumulado e a vida útil à fadiga, o POSFAL toma os resultados das variações de tensões ao longo do *riser* fornecidos pelo ANFLEX para os diversos casos de carregamento, associados as suas respectivas percentagens de ocorrência.

Os arquivos de entrada do POSFAL são gerados pelo ANFLEX para todos os casos de carregamento: um arquivo com extensão "**fah"** que corresponde a um histograma de análise de fadiga; um arquivo "**j21**" contendo as juntas do *riser* e informações adicionais de projeto, e um arquivo de configuração de execução, "**dat**". Os resultados do POSFAL, incluindo a vida útil ao longo do *riser*, são armazenados em um arquivo com extensão "**sai".**

Nesse arquivo "**sai**" também são informadas as juntas mais críticas do riser. Diante disso, o engenheiro pode executar o POSFAL novamente, informando estas juntas no arquivo "**fah**". O POSFAL irá gerar um novo arquivo "**sai**" e também um arquivo com extensão "**hst"** que contém um histórico de amplitude de variação de tensão para um *riser,* entre outras informações estatísticas. Entretanto, o *workflow* de análise de fadiga desta dissertação não foi projetado para realizar esta segunda execução do POSFAL.

O diagrama da Figura 4.10 ilustra uma visão do workflow onde é dado destaque ao papel desempenhado pelo POSFAL, mostrando suas entradas e saídas, assim como sua interação com o ANFLEX.



Sendo:

i1 : Séries de Esforços (FAA) ou Histogramas de amplitude de Variação de Tensões (FAH)

i2 :Arquivo com as juntas críticas selecionadas para a análise de fadiga.

o1 : Estimativa da vida útil do *riser*

Figura 4.10: Ênfase no papel do POSFAL no workflow

4.2 Procedimentos Atuais para Execução das Análises

Esta seção descreve os atuais procedimentos que os engenheiros *offshore* realizam para cada um dos programas envolvidos em uma análise de fadiga de *risers*. Os parâmetros de execução através da linha de comando e os arquivos envolvidos nos processos são apresentados a seguir. Atualmente esses programas são executados isoladamente, a implementação do workflow permitirá que eles sejam disparados automaticamente e sequencialmente.

As atividades da análise de fadiga dos *risers* que o workflow irá contemplar têm início com a execução do programa Prosim, que corresponde ao executável **Prosim.exe**. É necessário que alguns arquivos estejam presentes no mesmo diretório do **Prosim.exe** para que ele seja executado. Estes arquivos foram gerados por outros programas através de interfaces gráficas, portanto não podem ser disparados automaticamente pelo workflow. O arquivo com extensão "**prm**", gerado pelo SITUA, contém os dados gerais do modelo do casco da unidade flutuante e das linhas. Os coeficientes de ondas gerados pelo WAMIT, em um arquivo com extensão "**out**", são transformados no SITUA para que sejam lidos no Prosim em um arquivo com extensão "**wnf**". Os dados de batimetria estão contidos em um arquivo com extensão "**fun**" (que também foi convertido pelo SITUA a partir do arquivo "**dgn**" vindo do SGO, como já mencionado), e os coeficientes de vento e correntezas estão contidos, respectivamente, em arquivos com extensão "**cvs**" e "**cds**".

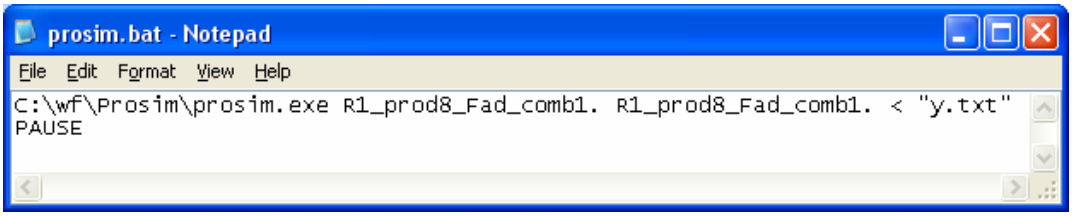


Figura 4.11: Conteúdo de um arquivo bat do Prosim

Os engenheiros encapsulam a execução do **Prosim.exe** em um arquivo "**bat**". Ele possui dois parâmetros de entrada: o primeiro corresponde ao nome do arquivo "**prm**" que será processado e o segundo corresponde ao nome do arquivo "**pld**" que irá conter a série de movimentos do sistema acoplado. Geralmente o nome do arquivo "**pld**" é atribuído ao arquivo "**prm**" para melhor organizar os arquivos das análises. A linha de comando ainda possui o operador "**<**" que redireciona o conteúdo do arquivo

y.txt, que é constituído de apenas do caracter '**y**' para permitir concluir a execução do **Prosim.exe**. A Figura 4.11 apresenta o conteúdo de um arquivo prosim.bat

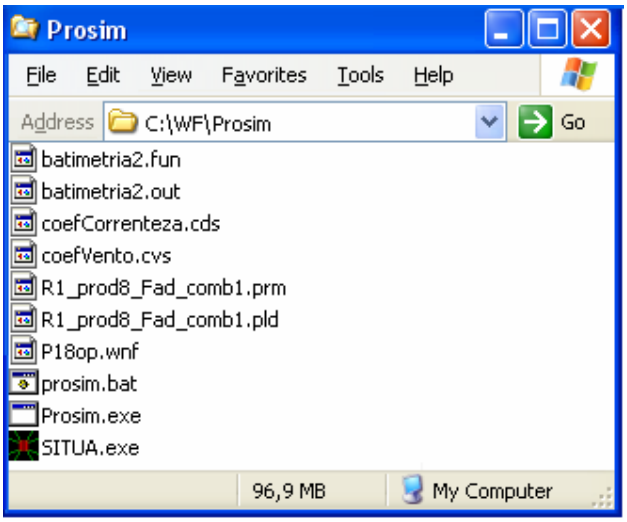


Figura 4.12: Arquivos envolvidos na execução do Prosim

A Figura 4.12 apresenta os arquivos mais relevantes envolvidos na execução do Prosim, listados no mesmo diretório. Além do arquivo "**pld**" também são gerados outros resultados no processamento, entretanto eles não são utilizados em um experimento de análise de fadiga. A análise de um arquivo "prm" pode gerar mais de um arquivo de movimento "**pld**", dependendo do número de combinações ambientais contidas/ativadas em cada modelo. Esta configuração, porém, é feita no SITUA. Cada arquivo "**pld**" possuirá no seu nome um sufixo que representa a condição ambiental ao qual o sistema *offshore* foi submetido, para gerar a respectiva série de movimentos. Usualmente, cria-se um "prm" para cada combinação ambiental, o que significa dizer que se teria um único "**pld**" associado a cada "**prm**".

O programa Conversor corresponde ao executável **PLD_TO_ANF.exe**. Os engenheiros podem encapsulá-lo em um arquivo "**bat**" que o aciona. Ele gera seis arquivos de extensão "**thf**" que contêm o movimento do sistema acoplado nas respectivas direções de movimento, a partir de um arquivo de movimentos de extensão "**pld**". Quando o Conversor inicia a sua execução, ele lê os parâmetros que estão contidos no arquivo **LE_ANF.dat**, que obrigatoriamente deve estar no mesmo diretório do executável. Este arquivo possui quatro parâmetros que estão representados na Figura 4.13, que ilustra um exemplo do arquivo.dat.

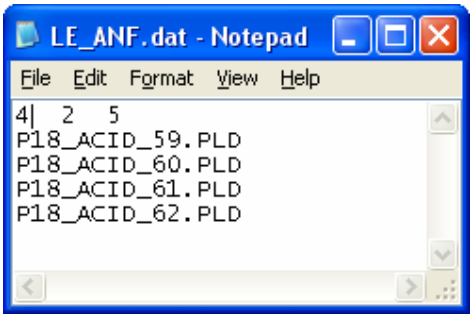


Figura 4.13: Arquivo de configuração do conversor

A primeira linha do arquivo contém três destes parâmetros. O primeiro número indica a quantidade de arquivos de movimentos que serão convertidos. O segundo número corresponde ao tipo de sinal que será utilizado para gerar a nova série de movimentos: o valor padrão é **2**, que indica que o sinal gerado terá somente o regime permanente; já o valor **1** indica que o sinal será composto pela combinação do movimento gerado por uma análise estática, com o regime permanente da análise dinâmica. O terceiro número indica o intervalo em que o sinal é capturado no tempo no arquivo "**pld**". O quarto parâmetro é composto pelo conjunto de arquivos de movimentos que serão processados pelo conversor; cada um é definido em uma linha diferente no arquivo.

Os arquivos de movimentos "**thf**" são gerados com o mesmo nome do arquivo de movimentos "**pld**". A Figura 4.14 apresenta todos os arquivos envolvidos na execução do Conversor listados no mesmo diretório. A importância do Conversor pode ser destacada pela diferença entre os tamanhos do arquivo **pld** em relação aos arquivos **thf.** Estes arquivos tiveram o conteúdo formatado a partir do arquivo "**pld"** para serem utilizados no ANFLEX.

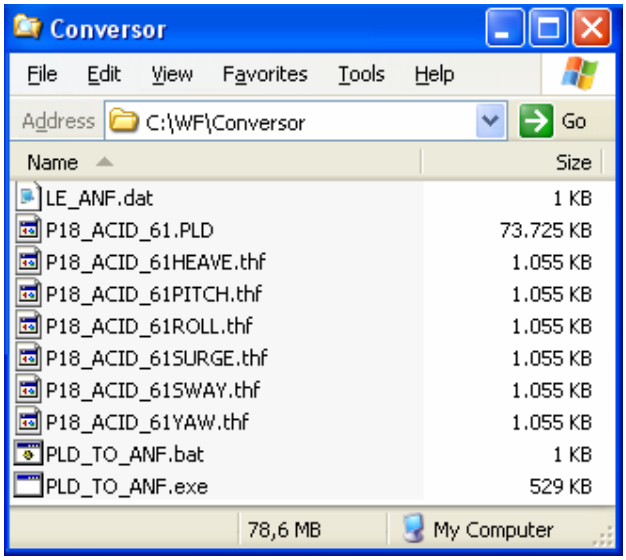


Figura 4.14: Arquivos envolvidos na execução do Conversor

O programa ANFLEX é composto por um conjunto de executáveis. O arquivo anflex.exe corresponde ao ANFLEX Model, que é utilizado através de uma interface gráfica e não está incluído no workflow. Os arquivos **Anf_I_704.exe, Anf_S_704.exe e Anf_D_704.exe** são responsáveis, respectivamente, por carregar os dados de entrada para as análises, executar a análise estática dos *risers* e executar a análise dinâmica dos *risers*; o número 704 representa a versão mais nova do programa. Estes executáveis são encapsulados em um arquivo "**bat**" e disparados sequencialmente. A Figura 4.15 apresenta o conteúdo deste arquivo cujo trecho destacado exibe a análise estrutural de um *riser* sob uma condição ambiental.

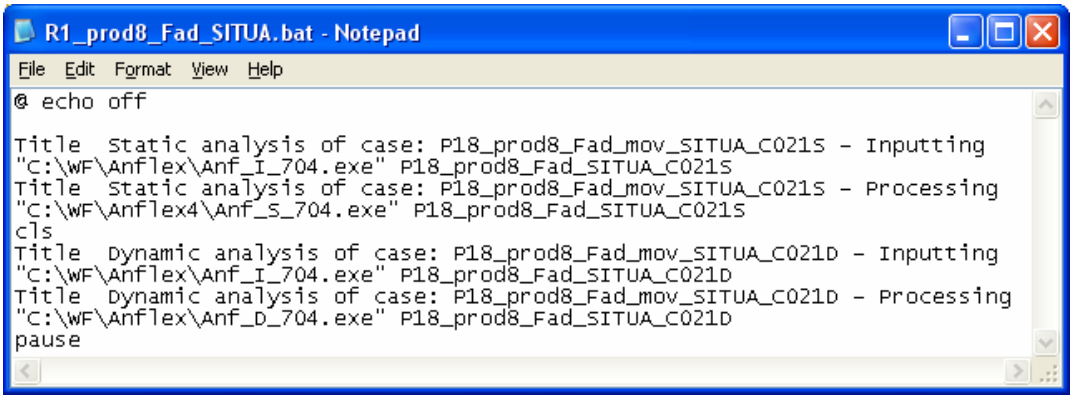


Figura 4.15: Conteúdo doe um arquivo bat do ANFLEX

Primeiramente o **Anf_I_704.exe** é executado passando como parâmetro o nome do arquivo **S.dat** com o modelo da linha para análise estática, e prepara os dados para o **Anf_S_704.exe**, que é executado passando este mesmo **S.dat** como parâmetro. Após a

execução do **Anf_S_704.exe**, **o Anf_I_704.exe** é chamado novamente, porém, dessa vez o parâmetro passado é o nome do arquivo**D.dat** com os carregamentos dinâmicos e prepara os dados para o **Anf_D_704.exe**, que é executado passando este mesmo **D.dat** como parâmetro.

Os arquivos **J21** e **FAH** são gerados com o mesmo nome do arquivo **dat** que contém a geometria das linhas. A Figura 4.16 apresenta todos os arquivos envolvidos na execução do ANFLEX listados no mesmo diretório. Opcionalmente, os arquivos **D.dat** e **S.dat** podem ser particionados em arquivos includes. Neste caso, usualmente cria-se um diretório "**include**" contendo todos os arquivos que compõem o **S.dat** e **D.dat**

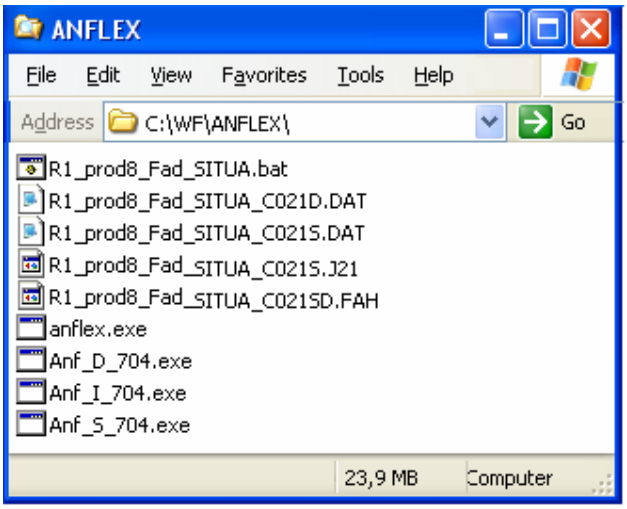


Figura 4.16: Arquivos envolvidos na execução do ANFLEX

O programa POSFAL corresponde ao executável **posfal15.exe**. Ele gera a estimativa de vida útil do *riser* no arquivo "**sai**", e pode gerar o histograma de amplitude de variação de tensão associado no arquivo "**hst**" numa segunda execução. Geralmente este executável é acionado pelos engenheiros através de um arquivo "**bat**" que apenas o encapsula. Ao iniciar sua execução ele verifica o conteúdo do arquivo **runfile.psf** que obrigatoriamente deve ficar no mesmo diretório dele. Este arquivo contém o nome de um ou mais arquivos de configuração (**.dat**) de execução do posfal sem a sua extensão "**dat**", conforme ilustrado na Figura 4.17. O arquivo de configuração contém vários parâmetros que representam as diretrizes para a execução do POSFAL, incluindo curvas S-N do *riser* e as condições ambientais em que os *risers* foram analisados com as suas respectivas probabilidades de ocorrências. Cada combinação ambiental está associada a um arquivo "**fah**" e cada riser a um arquivo "**J21**", ambos devem estar no mesmo diretório do **runfile.psf** no momento da execução.

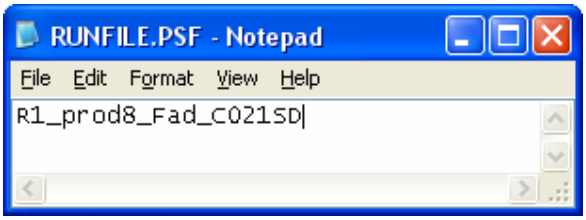


Figura 4.17: Conteúdo de um arquivo runsfile.psf

O POSFAL é executado uma vez para cada arquivo "**dat**" informado no **runfile.psf**. Os arquivos **sai** e **hst** são gerados com o mesmo nome do arquivo de configuração **dat**. A Figura 4.18 apresenta todos os arquivos envolvidos na execução do POSFAL listados no mesmo diretório.

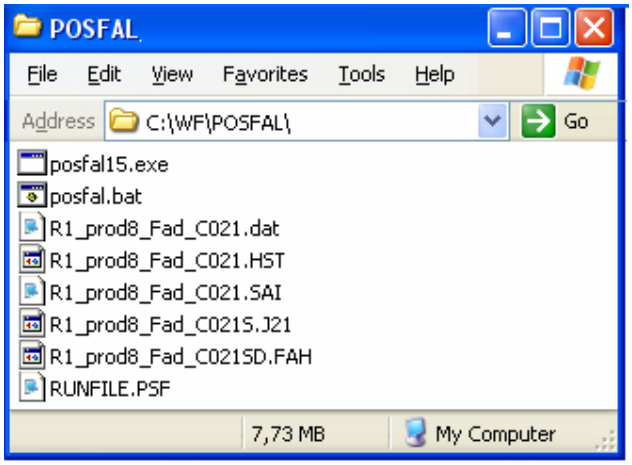


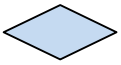
Figura 4.18: Arquivos envolvidos na execução do POSFAL

4.3 Modelagem do workflow

Uma outra maneira de representar a modelagem de workflow científico pode ser feita utilizando a UML (Unified modeling Language). Ela é uma linguagem gráfica que tem como objetivo documentar, especificar e estruturar a modelagem de um sistema, abstraindo como ele será implementado.

A UML possui diagramas que ilustram a estrutura e o comportamento do sistema. O diagrama de atividade é um exemplo de diagrama que ilustra o comportamento de sistema. Pode ser visto como uma extensão dos fluxogramas porque além de possuir toda a semântica existente em um fluxograma, também possui notação para representar ações concorrentes, juntamente com a sua sincronização. A fim de facilitar o entendimento do diagrama de atividades representados nas figuras 4.11 e 4.12, os seus elementos são descritos em uma legenda na Tabela 4.1.

Tabela 4.1: Legenda do Diagrama de Atividades

Elemento	Descrição
	Início do processo
	Atividade
	Ponto de decisão entre fluxos
	Junção de Fluxos
	Final do processo

O diagrama da Figura 4.19 apresenta a modelagem do workflow abstrato, representando as suas atividades e os dados envolvidos durante a análise de fadiga dos *risers*. Já o diagrama da Figura 4.20 apresenta a modelagem do workflow concreto. As atividades foram substituídas pelos respectivos softwares que executam as suas funções e os dados foram substituídos pelos arquivos cujas extensões representam os tipos de dados.

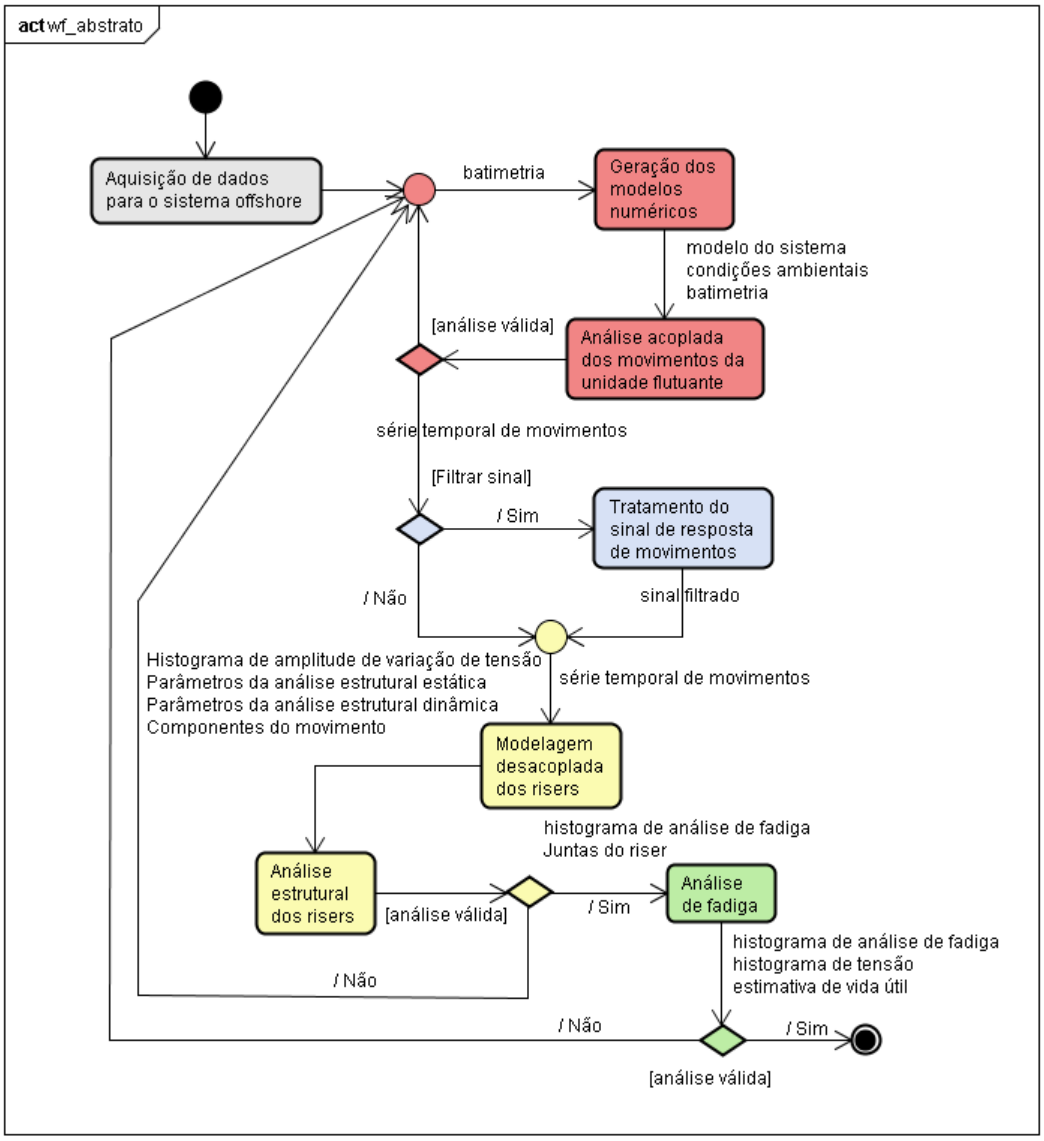


Figura 4.19: Workflow Abstrato

Uma diferença que pode ser claramente notada entre os diagramas de *workflows* abstrato e concreto é a presença do Conversor no workflow concreto. Uma vez que o formato dos dados produzidos pelo Prosim não são compatíveis com o formato de leitura do ANFLEX faz-se necessário a utilização do Conversor. Então, ele representa um recurso do workflow concreto para interligar as atividades presentes no workflow abstrato. Através desse recurso a especificação do workflow pode ser reutilizada.

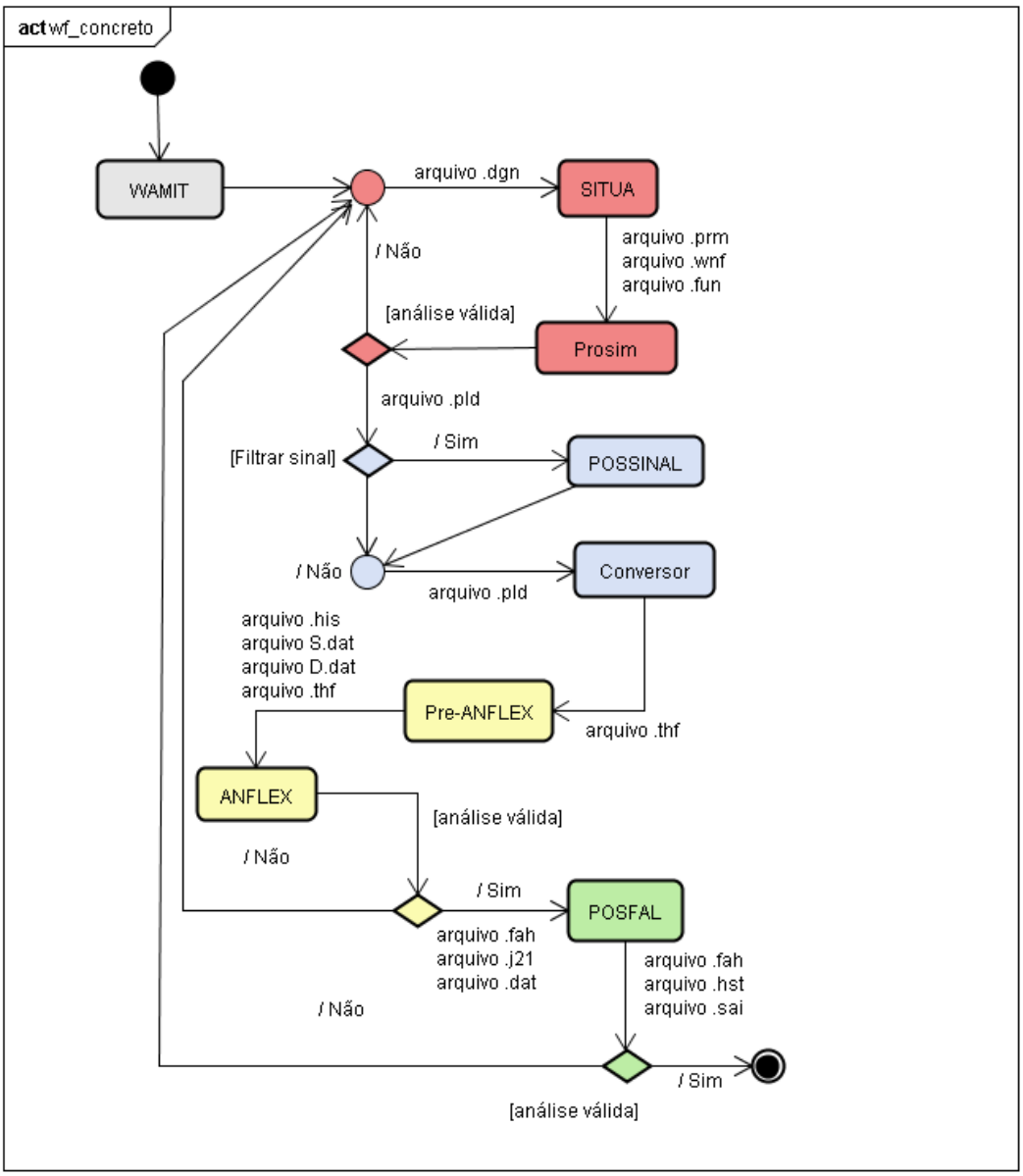


Figura 4.20: Workflow Concreto

Capítulo 5 - Implementação do Workflow

5.1 Introdução

Este capítulo descreve os procedimentos para a implementação do workflow de análise de fadiga de risers, considerando a especificação descrita no Capitulo 4, empregando a ferramenta Kepler descrita na seção 3.3.4.

Inicialmente, a Seção 5.2 trata das soluções adotadas para registrar a proveniência dos dados, utilizando como base os modelos descritos na seção 3.2.2, uma vez que o Kepler ainda não dispõe de um recurso para tal finalidade. Em seguida, a Seção 5.3 irá descreve a implementação do workflow no Kepler, detalhando os atores envolvidos no processo. Finalmente, as seções 5.4, 5.5 e 5.6 apresentam um tutorial contendo as instruções para configurar um ambiente, executar um workflow em um projeto de sistemas offshore e analisar os resultados.

5.2 Soluções para o Registro da Proveniência

O Kepler não possui uma ferramenta própria para gerenciar a proveniência dos *workflows*, portanto era necessário propor uma solução para esta necessidade. Então foi feito um estudo sobre as pesquisas dos modelos de proveniência existentes na literatura [94]. Uma vez que não há um modelo formal padrão para representar a proveniência em experimentos científicos, as proveniências prospectivas e retrospectivas e o OPM foram importantes para ajudar a direcionar o desenvolvimento de um modelo próprio que atendesse às necessidades dos *workflows* de sistemas *offshore*.

Em muitas aplicações científicas, foi observado que a manipulação de bancos de dados coexiste com a execução de módulos workflow. Os dados podem ser selecionados a partir de um banco de dados, tratados e utilizados em uma análise. Os resultados da análise podem então ser registrados em um banco de dados e potencialmente utilizados em outras análises. Portanto, para compreender a proveniência de um resultado, é preciso se conectar à proveniência cujas informações foram armazenadas em bases de dados e nos próprios workflow. Para combinar essas formas distintas de proveniência será exigido um framework que seja constituído de operações no banco de dados e que haja interação entre a estrutura de dados e a estrutura do workflow.

A partir desse contexto foi decidido armazenar a proveniência do workflow em um banco de dados. Então, primeiramente, era necessário desenvolver uma modelagem de dados que fosse capaz de comportar os dados envolvidos no processo [95], bem como responder as seguintes perguntas definidas pelos engenheiros que trabalhariam no experimento científico: Quem executou o workflow? Quando ele foi executado? O sistema *offshore* foi submetido a quais condições ambientais? Na análise estrutural dos *risers* foram feitas análises estáticas ou estáticas seguidas de dinâmicas? Como foi composto o movimento aplicado nas linhas? As atividades do workflow foram executadas com sucesso? É importante também registrar a data e hora da do início e fim de cada atividade do workflow.

O resultado da modelagem da proveniência do workflow é ilustrado na Figura 5.1, que representa o modelo lógico da base de dados. Este modelo não é exclusivo para um workflow de Análise de Fadiga de *Risers*. Ele foi desenvolvido para suportar a inclusão de novos *workflows* de outros experimentos de engenharia *offshore*.

As entidades com cor de fundo azul armazenam informações mais específicas aos sistemas *offshore*. Elas armazenam os engenheiros que representam os usuários do sistema; os experimentos científicos, e os *workflows* que correspondem aos processos que estão sendo automatizados, que possuem um conjunto de atividades associadas.

As entidades com fundo branco armazenam informações referentes às execuções dos *workflows*. Elas representam as instâncias dos workflow e as instâncias de suas respectivas atividades. Também são registrados todos os dados envolvidos no processo. A descrição detalhada de cada atributo de cada tabela do modelo é apresentada no Apêndice A deste documento.

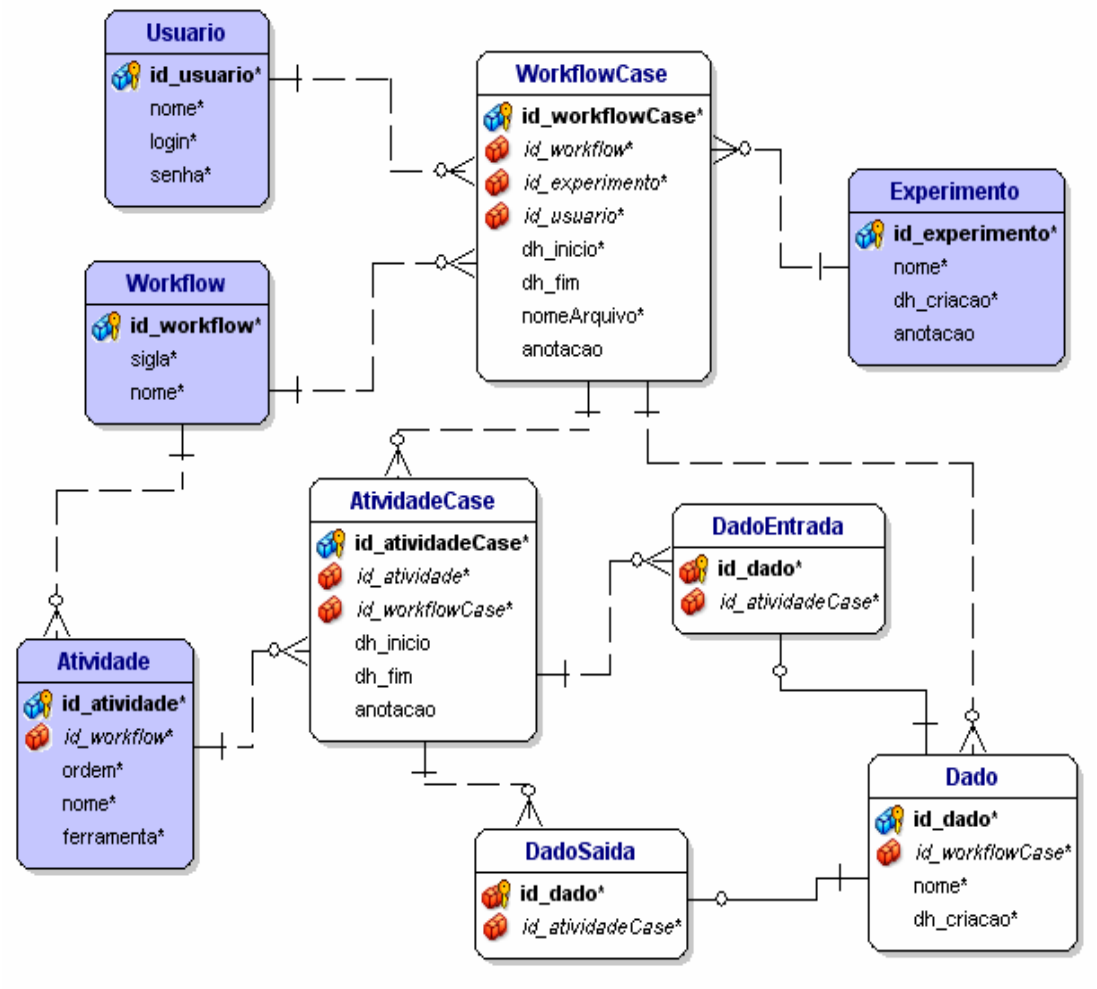


Figura 5.1: Modelo da proveniência em banco de dados.

Uma vez que o modelo de dados da proveniência já tinha sido definido, era necessário prover os meios para inserir as informações no banco e definir em qual sistema gerenciador de banco de dados (SGBD) ele seria implementado.

O Kepler possui componentes que comunicam com bancos de dados. O primeiro componente a ser utilizado é responsável por abrir uma conexão com o banco; um dos parâmetros desse componente é justamente informar o SGBD ao qual ele se conectará. Este componente oferece suporte aos principais SGBDs: SQL Server, MySQL, PostgreSQL e Oracle.

Foram feitos testes com o MySQL e com o PostgreSQL que são ferramentas gratuitas e muito utilizadas no meio acadêmico. Foi observado que o Kepler se comportou melhor com o PostgreSQL [96]. Então, este foi o SGBD escolhido para implementar o banco de dados.

Então, na implementação do workflow no Kepler, foram adicionados componentes que eram responsáveis para registrar no banco de dados de proveniência as informações sobre as execuções dos *workflows*. Desta maneira podemos representá-lo utilizando uma arquitetura em camadas, conforme ilustrado na Figura 5.2. A camada superior corresponde ao workflow do sistema *offshore* que executa as atividades do experimento, e a camada inferior representa o registro da proveniência [18].

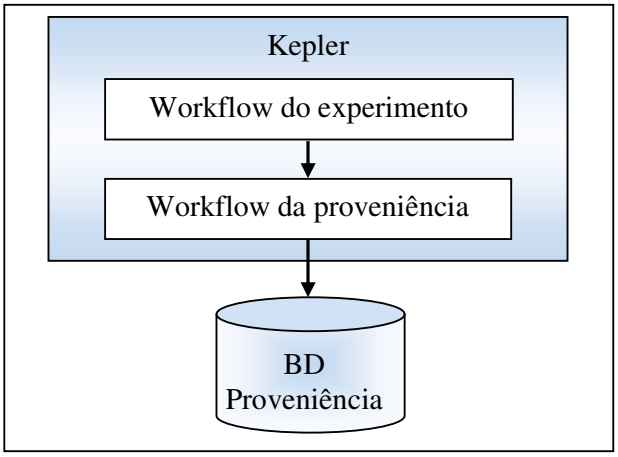


Figura 5.2: Arquitetura do workflow no Kepler

A Figura 5.3 apresenta uma visão da arquitetura do sistema que foi gerado para gerenciar os experimentos científicos de sistemas *offshore*. O workflow implementado no Kepler executa as atividades e gerencia a proveniência, cujos dados estão armazenados em um banco de dados PostgreSQL [97, 98]. Entretanto, ainda é preciso prover meios para que estas informações sejam consultadas pelos engenheiros. Para isso, foi criado um sistema web desenvolvido utilizando a linguagem PHP [99]. Os usuários acessam o sistema que lhes permite acompanhar a execução do workflow, consultar os dados de execuções anteriores e escrever comentários, anotações sobre as execuções.

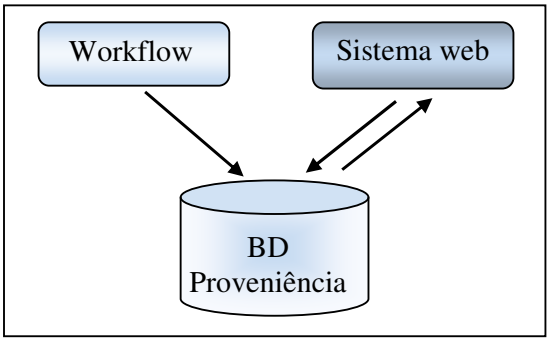


Figura 5.3: Arquitetura da solução da proveniência.

Um sistema web contribui para o compartilhamento das informações dos experimentos científicos porque as atualizações dos dados são refletidas para todos os usuários. Os ambientes heterogêneos dos experimentos não representam problemas, uma vez que ele pode ser acessado de qualquer computador conectado à rede; não necessita ser instalado no computador do cliente. Ele é acessado através de um navegador; portanto, ele é executado independentemente do sistema operacional do cliente. Os desenvolvedores têm mais controle para gerenciá-lo porque a instalação e as atualizações são feitas apenas nos servidores.

5.3 Implementação do workflow no Kepler

Nesta seção será apresentado o workflow que foi desenvolvido para automatizar as atividades computacionais realizadas pelos engenheiros de sistemas *offshore* que foram descritas na seção 4.2, assim como registra a proveniência dos dados e das atividades envolvidas no processo. A Figura 5.4 ilustra uma visão geral do workflow implementado no Kepler. Ela apresenta o diretor que define a semântica da interação entre os atores; os parâmetros de execução; e os atores, que conectados entre si, constituem o workflow científico.

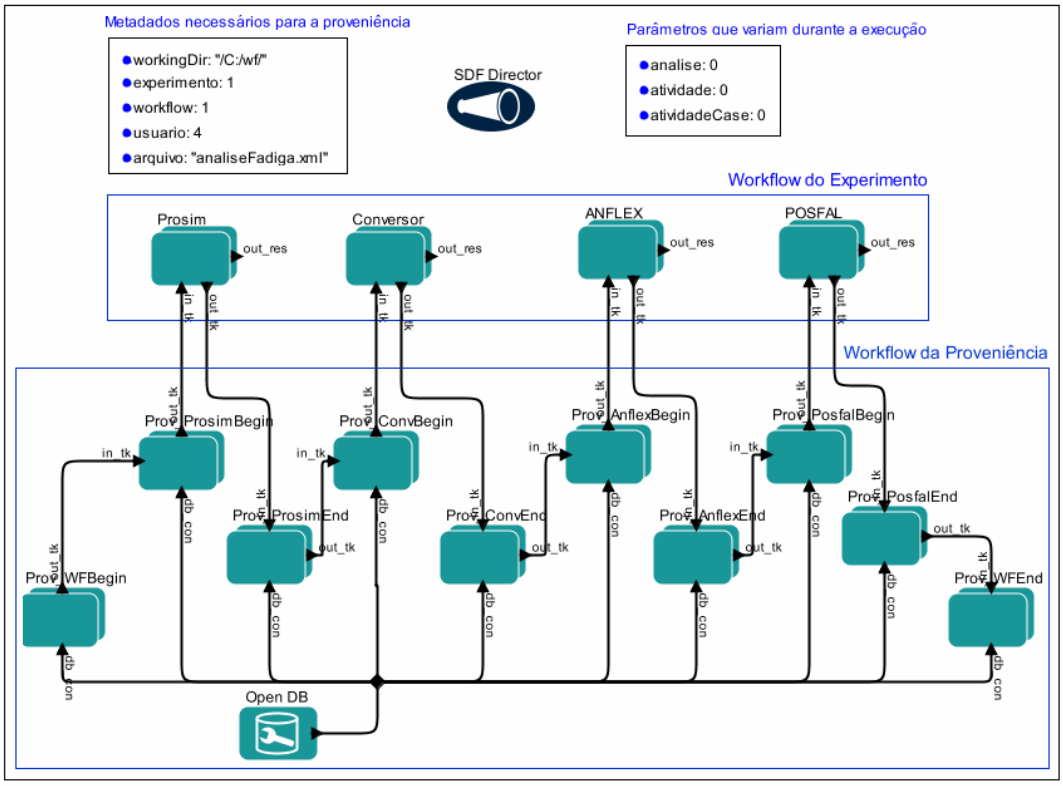


Figura 5.4: Visão geral do workflow implementado no Kepler.

Conforme foi ilustrado na Figura 5.2, a arquitetura do workflow do Kepler da Figura 5.4 possui um fluxo que representa a execução do experimento científico, neste caso da análise de fadiga de *risers*; e também possui um fluxo responsável pelo registro da proveniência. Essa divisão foi feita não só para melhor organizar o workflow visualmente, mas também para empregar a Metodologia do Reuso, a fim de permitir que novos *workflows* sejam desenvolvidos a partir desse. O workflow que controla a proveniência representa um modelo fechado, deste modo, o engenheiro pode aproveitá-

lo, e pode ficar mais concentrado em desenvolver o workflow que constitui o experimento científico.

O diretor SDF foi utilizado porque ele é responsável por garantir que o workflow seja executado de modo síncrono através da transferência de tokens entre os atores. Os parâmetros de execução do workflow são destacados na Figura 5.5. Eles estão agrupados dentro de dois retângulos para separar os parâmetros que devem ser definidos pelos engenheiros (Figura 5.5(a)) dos parâmetros que são atualizados automaticamente pelo próprio workflow (Figura 5.5 (b)).

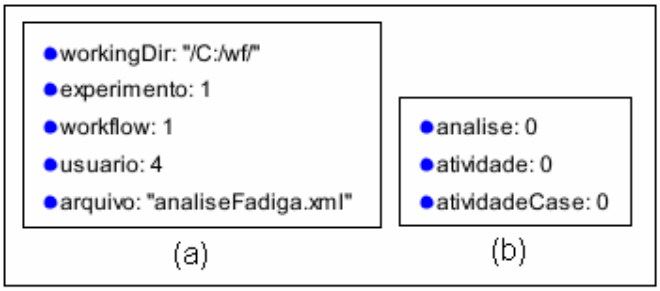


Figura 5.5: Parâmetros de execução do workflow no Kepler

O parâmetro workingDir indica o diretório raiz a partir do qual os programas do workflow do experimento científico são executados. Ele foi criado porque foram informados os endereços absolutos dos programas. Quando eram informados os endereços relativos ao diretório onde o arquivo do Kepler fora salvo, o Kepler gerava um erro de execução uma vez que não encontrava os programas.

Os parâmetros experimento, workflow, usuário e arquivo devem ser obrigatoriamente informados para registrar a proveniência do workflow no banco de dados. Experimento se refere ao identificador numérico do cadastro de um experimento no sistema web de proveniência, que corresponde a um projeto de sistema offshore. Workflow se refere ao identificador numérico do workflow cadastrado no sistema web de proveniência, neste caso indica que esta sendo executada uma Análise de Fadiga de *Risers*. Usuário se refere ao identificador numérico do usuário também cadastrado no sistema web de proveniência e indica o usuário que executou o workflow no Kepler. E Arquivo é o nome do arquivo no Kepler que foi configurado com determinados parâmetros de execução. É necessário criar esses parâmetros uma vez que são informações importantes para a proveniência e não há um outro modo de defini-las no

Kepler de modo claro para o engenheiro. Não é possível criar um formulário no Kepler onde essas informações poderiam ser preenchidas.

Os parâmetros **análise**, **atividade** e **atividadeCase** são definidos automaticamente pelo workflow, sendo que os dois últimos têm seus valores alterados dinamicamente durante a execução do workflow. **Análise** corresponde ao identificador numérico da instância da execução do workflow criada no banco de dados de proveniência no início da sua execução no Kepler. **Atividade** corresponde ao identificador numérico da atividade cadastrada pelo sistema web de proveniência que está sendo processada em um determinado momento da execução do workflow.

Assim como a **análise** representa a instância do workflow, o parâmetro **atividadeCase** representa uma instância da atividade do workflow. Então o seu valor corresponde ao identificador numérico gerado no banco de dados de proveniência no início da execução da atividade.

5.3.1 Workflow de Proveniência

O diretor e os parâmetros do workflow da Figura 5.4 já foram explicados. Então, nos concentraremos na descrição dos atores presentes nele. Podemos começar pelo ator OpenDB que é o responsável por criar uma conexão com o banco de dados da proveniência. Ele possui os seguintes parâmetros que devem ser obrigatoriamente definidos: *database format*, que indica o SGBD ao qual ele irá se conectar, neste caso estamos utilizando o PostgreSQL; *databaseURL*, que indica o nome do esquema no banco de dados que será utilizado; *username*, que representa o usuário que irá se conectar ao banco; e *password* que corresponde a senha do respectivo usuário. A saída deste ator é a referência para a conexão criada com o SGBD. Ela é distribuída para todos os componentes que irão se conectar ao banco de dados para executar um comando de inserção, alteração ou consulta de dados.

Os outros atores que constituem o workflow da proveniência são instâncias do CompositeActor que foi definido na seção 3.4.4, então cada um deles representa um sub-workflow e serão descritos a seguir.

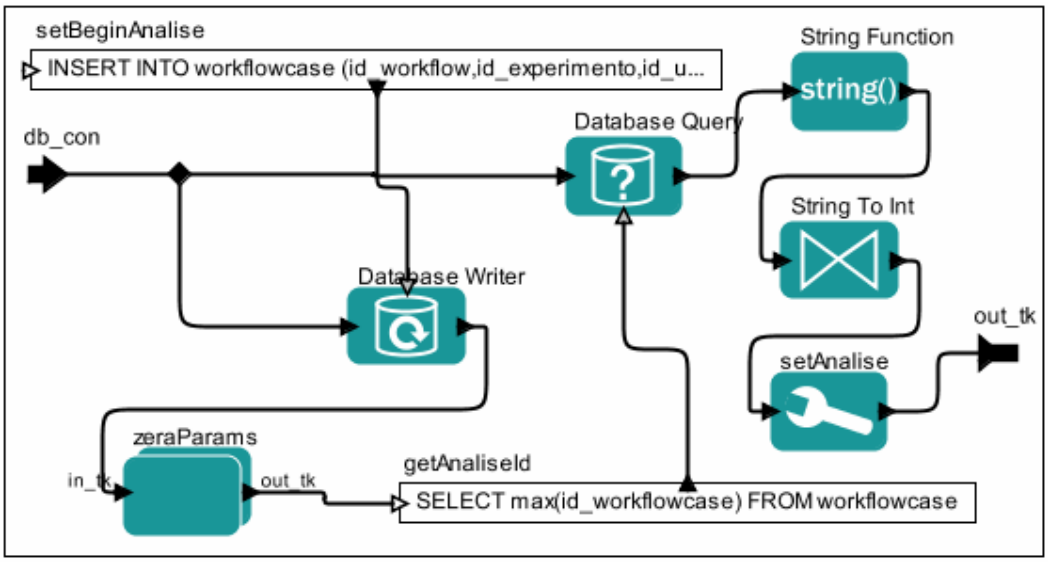


Figura 5.6: Sub-workflow que registra o início da execução do workflow

O ator Prov_WFBegin representa o sub-workflow ilustrado na Figura 5.6. Sua porta de entrada é a referência para a conexão do banco de dados, denominada db_con que é transmitida aos atores Database Writer e Database Query. O primeiro executa o comando definido na constante setBeginAnalise para registrar no banco de dados de proveniência o início da execução do workflow.

Em seguida, ele dispara um outro sub-workflow denominado zeraParams, ilustrado na Figura 5.7. Este sub-workflow é responsável por atribuir um valor nulo nos parâmetros analise, atividade e atividadeCase para garantir que eles não possuam valores referentes às execuções anteriores, de modo que possam comprometer a integridade e confiabilidade da proveniência.

A saída do zeraParams ativa a o ator Database Query que executa o comando definido na constante getAnaliseId para retornar o identificador da análise. Este valor é tratado no ator StringFunction que retira eventuais espaços em branco. Então este valor é convertido de string para inteiro no ator StringToInt e é finalmente atribuído ao parâmetro Analise do workflow. A porta de saída out_tk irá transmitir um token para o próximo ator do workflow de proveniência.

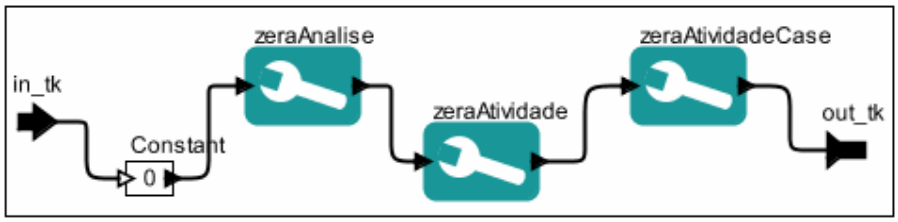


Figura 5.7: Sub-workflow que zera parâmetros de execução

Uma vez que o workflow de proveniência já registrou o início da execução da análise, agora ele irá trabalhar em um nível mais baixo de granularidade, já que acompanhará a execução de cada atividade do worflow. Antes da execução de cada atividade há um respectivo sub-workflow que registra o seu início, assim como, após a mesma execução, há um respectivo sub-workflow que registra a sua conclusão.

A primeira atividade do workflow de Análise de Fadiga é o Prosim. O ator Prov_ProsimBegin é o sub-workflow responsável por registrar o início da sua execução, e está ilustrado na Figura 5.8. Possui duas portas de entrada: db_con e in_tk. Este último ativa o comando definido na constante Prov_getProsimId que será executado pelo dbQuery1 que retorna o identificador numérico da atividade Prosim. O valor do parâmetro nomeAtiv1 está contido no valor dessa constante. Ele representa o nome da atividade (Prosim) que está na expressão lógica do comando de consulta ao banco de dados.

O identificador é tratado nos atores StringFunction, StringToInt e é atribuído ao parâmetro atividade pelo ator setAtividade. Em seguida, a constante setProsimBegin é ativada. Ela representa o comando executado pelo ator dbWriter que registra o início da execução da atividade Prosim. O dbWriter retorna o número de registros afetados pelo comando. Neste caso, este resultado apenas representa o token que ativa a constante getAtividadeCaseId. Ela representa o comando executado pelo ator dbQuery2 que retorna o identificador numérico da instância da atividade que fora criada pelo dbWriter. Este identificador é tratado nos atores StringFunction, StringToInt e é atribuído ao parâmetro atividadeCase pelo ator setAtividadeCase.

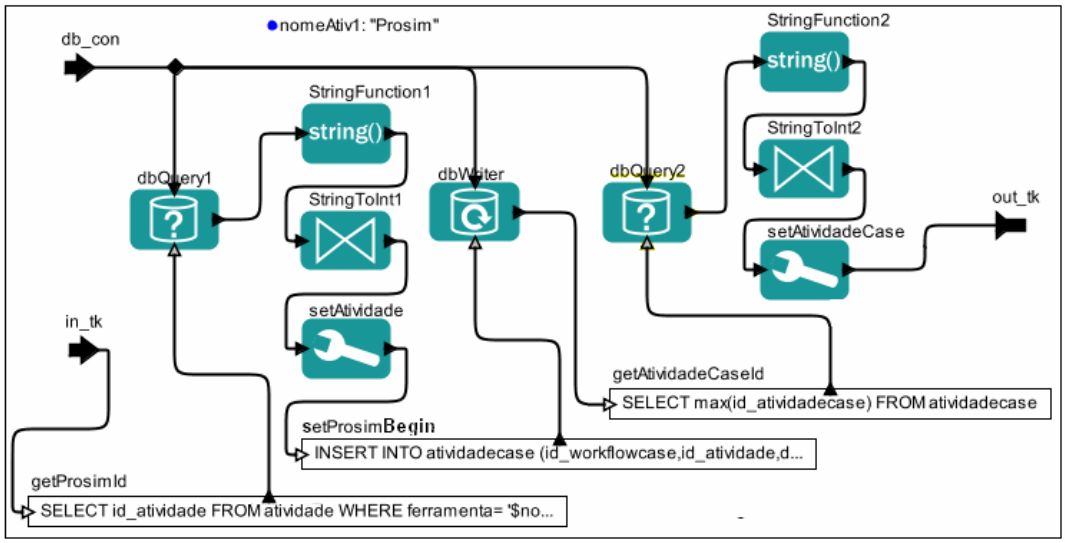


Figura 5.8: Sub-workflow que registra o início da execução do Prosim

A porta de saída out_tk do ator Prov_ProsimBegin contém o token que é enviado para o Workflow do Experimento. O CompositeActor Prosim representa o sub-workflow que executará efetivamente o Prosim. Quando a execução do Prosim for concluída, um token é enviado ao WorkfFlow de Proveniência, mais especificamente para o ator Prov_ProsimEnd ilustrado na Figura 5.9. Possui duas portas de entrada: db_con e in_tk. Este último ativa o comando definido na constante setProsimEnd que será executado pelo dbWriter que registra o fim da execução da instância da atividade do Prosim na proveniência.

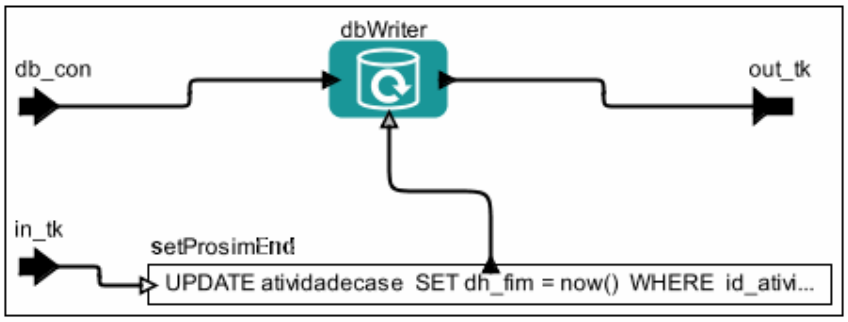


Figura 5.9: Sub-workflow que registra o fim da execução do Prosim

Como foi dito anteriormente, o workflow de proveniência possui, para cada atividade, um sub-workflow que registra o seu início e um outro que registra o se fim. Deste modo, os pares de atores Prov_ConversorBegin e Prov_ConversorEnd, Prov_AnflexBegin e Prov_AnflexEnd, Prov_PosfalBegin e Prov_PosfalEnd são responsáveis por registrar a proveniência das respectivas atividades: Conversor,

ANFLEX e POSFAL. Os funcionamentos desses sub-*workflows* são análogos aos sub-workflow que registram a proveniência do Prosim.

Concluindo o workflow de proveniência, o ator Prov_WFEnd representa o sub-workflow ilustrado na Figura 5.10. Possui duas portas de entrada: db_con e in_tk. Este último ativa o comando definido na constante setEndAnalise que será executado pelo Database Writer que registra o fim da execução do workflow na proveniência. Em seguida, a conexão com banco de dados é fechada e é exibido na tela um resultado que indica que a execução do workflow foi concluída.

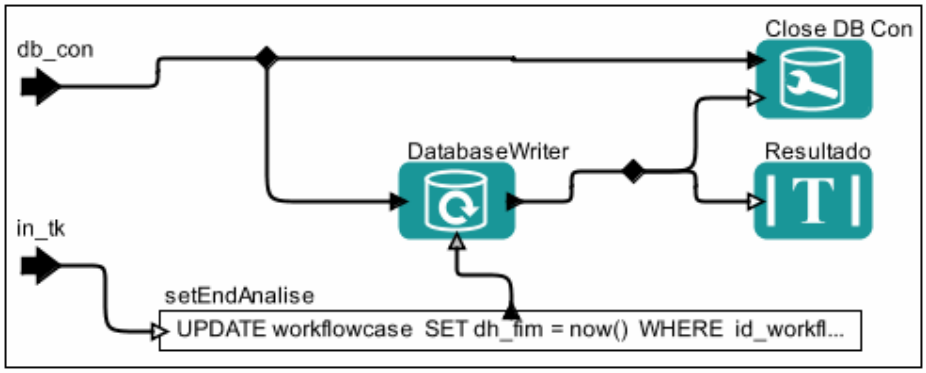


Figura 5.10: Sub-workflow que registra o fim da execução do workflow

5.3.2 Workflow do Experimento

O workflow da proveniência foi explicado acima. Então, agora apresentaremos o workflow do experimento. Os sub-*workflows* apresentados a seguir são responsáveis pela execução das atividades do experimento científico.

A primeira atividade é desempenhada pelo Prosim que realizará a análise dos movimentos do sistema offshore acoplado. A Figura 5.11 ilustra a execução do Prosim. O parâmetro workingDir corresponde ao diretório onde os executáveis deste processo estão localizados. Este sub-workflow possui a porta de entrada in_tk. Ela transmite o token que ativa a constante const_ProsimBat que contém o comando executar o ator setProsimBat. Este ator faz a chamada ao executável que cria o arquivo Prosim.bat. Este arquivo.bat contém a instrução que executará o Prosim.exe.

A saída do setProsimBat transmite o token para o ator ProsimExe, cujo comando é a constante const_Prosim que faz a chamada ao Prosim.bat. A saída do ProsimExe é distribuída para as portas de saída out_tk e out-res. A primeira transmitirá este token para o próximo ator no workflow. A segunda não está sendo efetivamente utilizada mas ela permite que o resultado seja armazenado em um arquivo ou pode exibi-lo na tela.

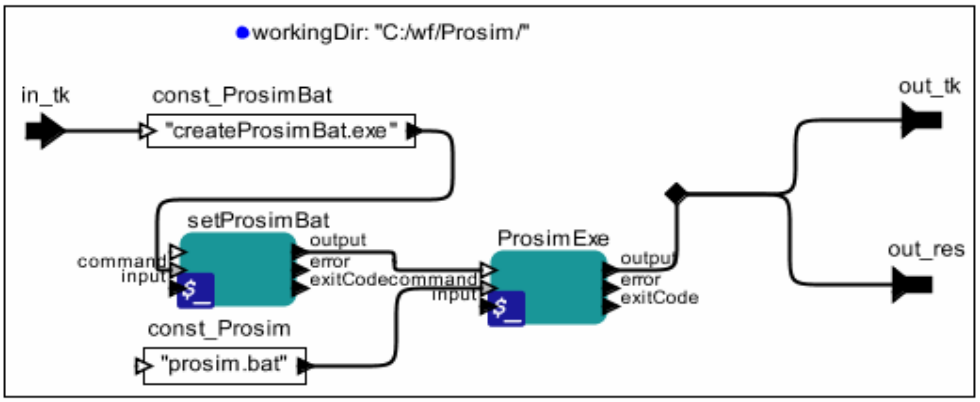


Figura 5.11: Sub-workflow de execução do Prosim

A segunda atividade é desempenhada pelo Conversor que irá converter a série de movimentos em seis arquivos, cada um representando uma direção de movimento. A Figura 5.12 ilustra a execução do Conversor. O parâmetro workingDir corresponde ao diretório onde os executáveis deste processo estão localizados. Este sub-workflow possui a porta de entrada in_tk. Ela transmite o token que ativa a constante const_conversorDat que contém o comando executar o ator setConversorDat. Este ator faz a chamada ao executável que cria o arquivo LE_ANF.dat. Este arquivo.dat contém os parâmetros de configuração que definem a execução do Conversor.

A saída do setConversorDat transmite o token para o ator ConversorExe, cujo comando é a constante const_conversor que faz a chamada ao PLD_TO_ANF.bat que encapsula a execução do PLD_TO_ANF.exe. A saída do ConversorEx é distribuída para as portas de saída out_tk e out-res. A primeira transmitirá este token para o próximo ator no workflow. A segunda não está sendo efetivamente utilizada mas ela permite que o resultado seja armazenado em um arquivo ou pode exibi-lo na tela.

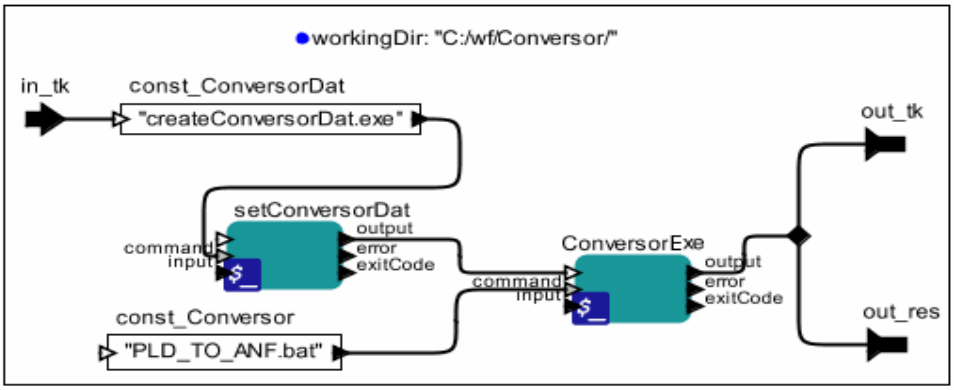


Figura 5.12: Sub-workflow de execução do Convesor

A terceira atividade é desempenhada pelo ANFLEX que efetuará uma análise estrutural nos *risers*. A Figura 5.13 ilustra a execução do ANFLEX. O parâmetro workingDir corresponde ao diretório onde os executáveis deste processo estão localizados. Este sub-workflow possui a porta de entrada in_tk. Ela transmite o token que ativa a constante const_AnflexBat que contém o comando executar o ator setAnflexBat. Este ator faz a chamada ao executável que cria o arquivo anflex.bat. Este arquivo.bat contém a sequência de comandos que executarão os programas anf_I_704.exe, anf_S_704.exe e anf_D.exe.

A saída do setAnflexBat transmite o token para o ator AnflexExe, cujo comando é a constante const_Anflex que faz a chamada ao anflex.bat. A saída do AnflexExe é distribuída para as portas de saída out_tk e out-res. A primeira transmitirá este token para o próximo ator no workflow. A segunda não está sendo efetivamente utilizada mas ela permite que o resultado seja armazenado em um arquivo ou pode exibi-lo na tela.

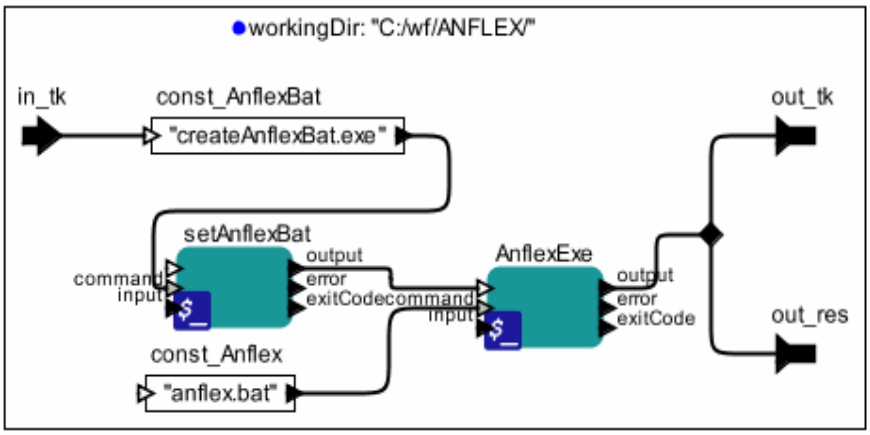


Figura 5.13: Sub-workflow de execução do ANFLEX

A última atividade é desempenhada pelo POSFAL que efetivamente realiza a análise de fadiga dos risers. A Figura 5.14 ilustra a execução do POSFAL. O parâmetro workingDir corresponde ao diretório onde os executáveis deste processo estão localizados. Este sub-workflow possui a porta de entrada in_tk. Ela transmite o token que ativa a constante const_PosfalPsf que contém o comando executar o ator setPosfalRunfile. Este ator faz a chamada ao executável que cria o arquivo Runfile.psf. Este arquivo.dat contém a lista dos arquivos 'fah' que serão processados na execução do POSFAL. A saída do setPosfalRunfile transmite o token para o ator PosfalExe, cujo comando é a constante const_Posfal que faz a chamada ao posfal.bat que encapsula a execução do posfal15.exe.

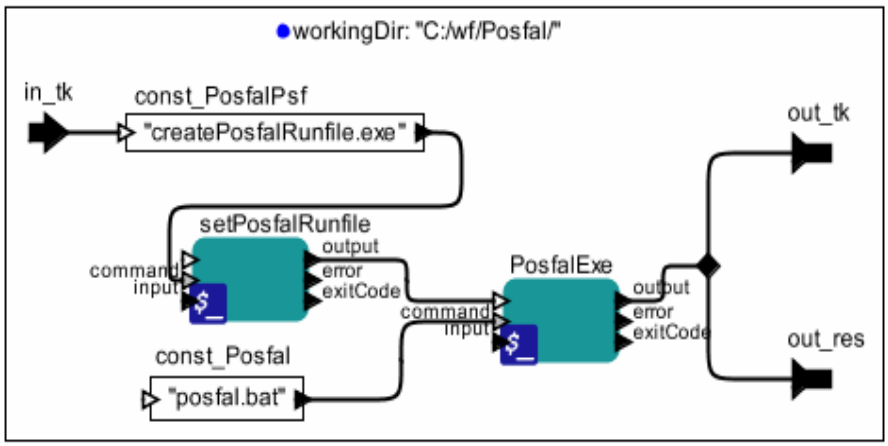


Figura 5.14: Sub-workflow de execução do POSFAL

A saída do PosfalExe é distribuída para as portas de saída out_tk e out-res, sendo que esta última não está sendo efetivamente utilizada mas ela permite que o resultado seja armazenado em um arquivo ou pode exibi-lo na tela. A saída out_tk é transmitida para o workflow de proveniência que executará o sub-workflow que registra o fim da atividade do POSFAL e, em seguida, o outro que registra o fim da execução do próprio workflow.

5.4 Configuração do Ambiente do *Workflow*

Conforme foi apresentado na seção 5.3, o *workflow* de análise de fadiga possui dois fluxos de execução, que correspondem a um *workflow* de proveniência associado a um *workflow* que executa o próprio experimento. É necessário que o ambiente do *workflow* seja devidamente configurado antes dos engenheiros serem capazes de executá-lo no Kepler. O texto a seguir está montado como um tutorial para a configuração desse ambiente e para a execução do *workflow*, apresentando a sequência de ações que os usuários devem executar.

É importante destacar que *workflow* armazena a proveniência em um bancos de dados PostGreSQL e o sistema que consulta a proveniência é um sistema web, portanto, deve ser executado em um navegador. Diante disso, para este projeto, é necessário que o sistema gerenciador de banco de dados PostgreSQL esteja instalado na máquina onde os *workflow* serão executados e que um servidor web Apache esteja devidamente configurado para interpretar PHP, uma vez que foi a linguagem em que o sistema foi desenvolvido. As atividades de instalação do servidor e do banco de dados não serão descritas aqui, uma vez que não estão contidas no escopo desta dissertação.

Os parâmetros de execução do *workflow* para registrar a proveniência foram apresentados na Figura 5.5 (a). Os parâmetros **usuário**, **experimento** e ***workflow*** correspondem, respectivamente, aos identificadores numéricos dos usuários, experimentos e *workflows* que foram cadastrados primeiramente no sistema web. Estes parâmetros claramente informam quem executou o experimento e qual o *workflow* (processo) que ele desempenhou.

Antes de informar como esses cadastros são realizados, será apresentada uma visão geral do mesmo.

5.4.1 Acesso ao Sistema

O sistema possui um perfil de administrador para poder realiza o primeiro acesso ao sistema quando não há usuários cadastrados. A Figura 5.15 apresenta a tela de login do sistema. O administrador e posteriormente os novos usuários devem informar seus respectivos logins e senhas para terem acesso ao sistema.

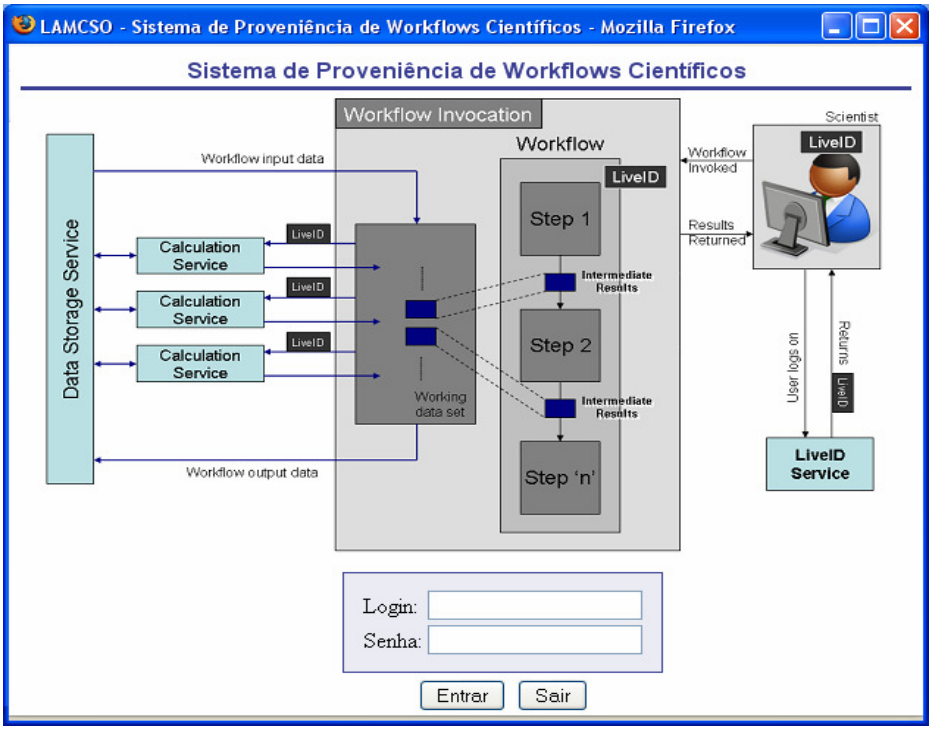


Figura 5.15: Tela inicial de login do sistema web

A sua interface é bem leve e simples. Ele possui um menu superior, representado na Figura 5.16, através do qual acessa as funcionalidades de consultas do sistema e possui os seguintes links: Experimentos, que exibe os experimentos que correspondem a um conjunto de análises; Análises, que apresenta as instâncias de execução dos *workflows* associadas a um experimento; *Workflows*, que exibe os processos que foram automatizados; Usuários, que mostra os usuários cadastros no sistema; e Sair, onde o usuário se desconecta do sistema.

Figura 5.16: Menu superior do sistema web

5.4.2 Cadastro de Usuários

Ao clicar na opção Usuários do menu superior, o sistema apresenta uma tabela que contém todos os usuários cadastrados no sistema que correspondem aos engenheiros que executam os *workflows*. Esta tela é ilustrada na Figura 5.17. As colunas da tabela apresentam o identificador, o nome e o login de cada usuário. O sistema permite cadastrar novos usuários, assim como consultar, alterar, excluir e pesquisar usuários já cadastrados.

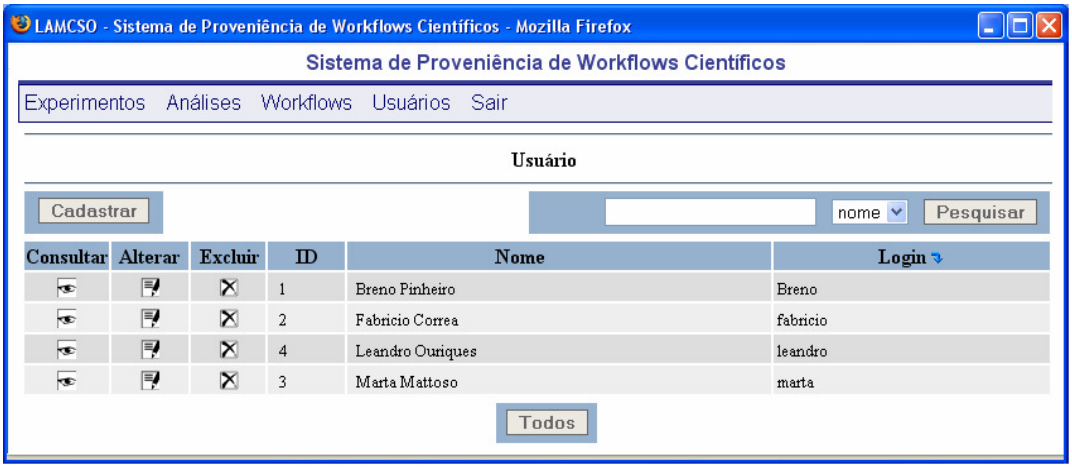


Figura 5.17:Tela que exibe os usuários do sistema

A Figura 5.18 apresenta o formulário que define as informações referentes aos usuários. Ele é exibido quando as funcionalidades de cadastrar, alterar e excluir são acionadas. Para que um usuário seja cadastrado é necessário informar seu nome, login e a senha de acesso e confirmar a senha.

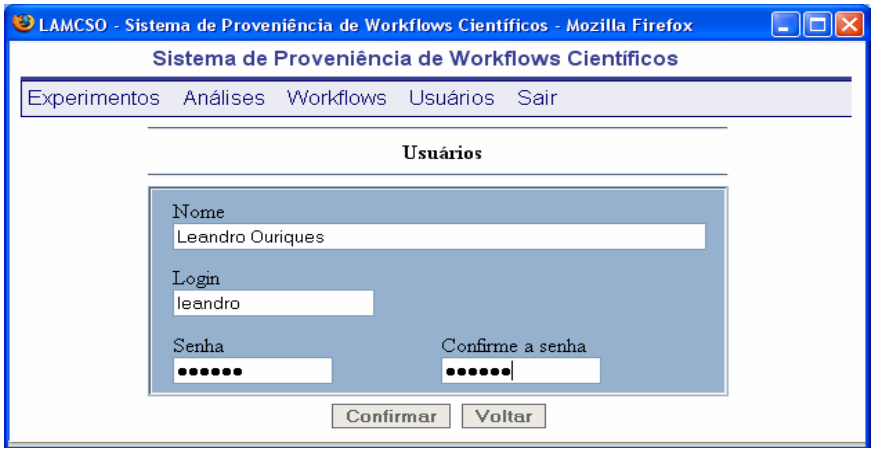


Figura 5.18: Tela de cadastro dos usuários do sistema

5.4.3 Cadastro de *Workflows*

Apesar de que esta dissertação só esteja analisando um wokflow de análise de fadiga de *risers*, o sistema de proveniência foi modelado para suportar quaisquer *workflows* que representem um experiment científico. Então, o sistema oferece uma funcionalidade para cadastrar outros *workflows*.

Ao clicar na opção *Workflows* do menu superior, o sistema apresenta uma tabela que contém todos os *workflows* cadastrados no sistema que correspondem aos experimentos científicos de engenharia *offhore* que podem ser automatizados. Esta tela é ilustrada na Figura 5.19. As colunas da tabela apresentam o identificador, a sigla e o nome de cada *workflow*. O sistema permite cadastrar novos *workflows*, assim como consultar, alterar, excluir e pesquisar *workflows* já cadastrados.

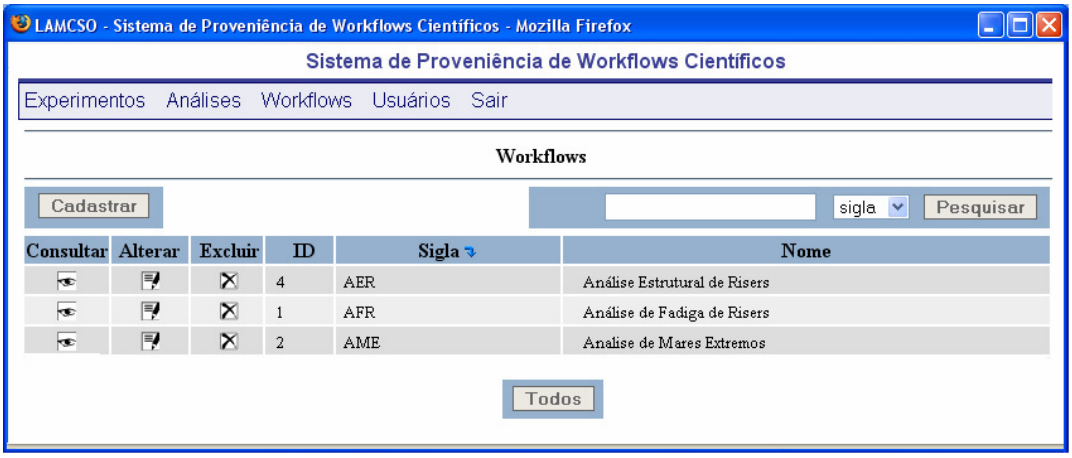


Figura 5.19: Tela que exibe os *workflows* do sistema

A Figura 5.20 apresenta o formulário que define as informações referentes aos *workflows*. Ele é exibido quando as funcionalidades de cadastrar, alterar e excluir são acionadas. Para que um *workflow* seja cadastrado é necessário informar uma sigla e seu nome. Um *workflow* é formado por uma seqüência de atividades, portanto, neste cadastrado do *workflow*, também é necessário informá-las. O sistema apresenta na mesma interface deste formulário, uma tabela que contém todas as atividades executadas no *workflow*. As colunas da tabela apresentam a ordem da sua execução no *workflow*, o nome de cada atividade e o nome da ferramenta computacional que implementa a atividade. Então, o sistema permite cadastrar novas atividades, assim como alterar e excluir as atividades já cadastradas.

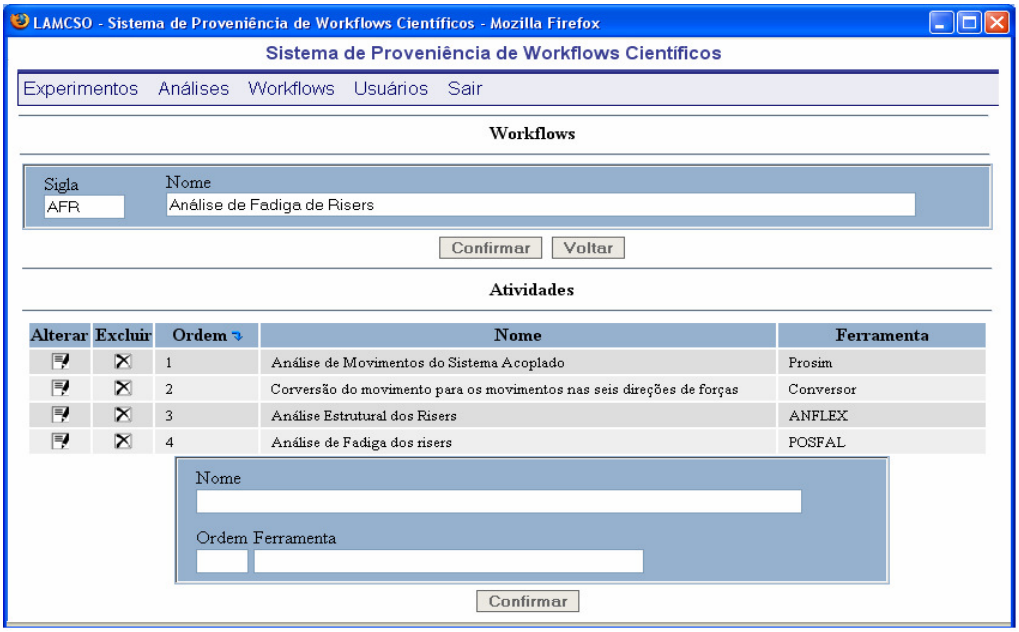


Figura 5.20: Tela de cadastro do *workflow* e das suas respectivas atividades

5.4.4 Cadastro de Experimentos

Ao clicar na opção Experimentos do menu superior, o sistema apresenta uma tabela que contém todos os experimentos científicos cadastrados no sistema. Um experimento é constituído de várias execuções de um *workflow*, porém estas execuções de cada experimento são apresentadas na opção Análises do menu superior. Esta tela é ilustrada na Figura 5.21. As colunas da tabela apresentam o identificador, o nome atribuído ao experimento, a sua data de criação e anotações cadastradas pelos engenheiros. O sistema permite cadastrar novos experimentos, assim como consultar, alterar, excluir e pesquisar experimentos já cadastrados.

Figura 5.21: Tela que exibe os experimentos do sistema

A Figura 5.22 apresenta o formulário que define as informações referentes aos experimentos. Ele é exibido quando as funcionalidades de cadastrar, alterar e excluir são acionadas. Para que um experimento seja cadastrado é necessário informar seu nome e data e hora de criação. Os usuários também podem adicionar anotações sobre os experimentos.

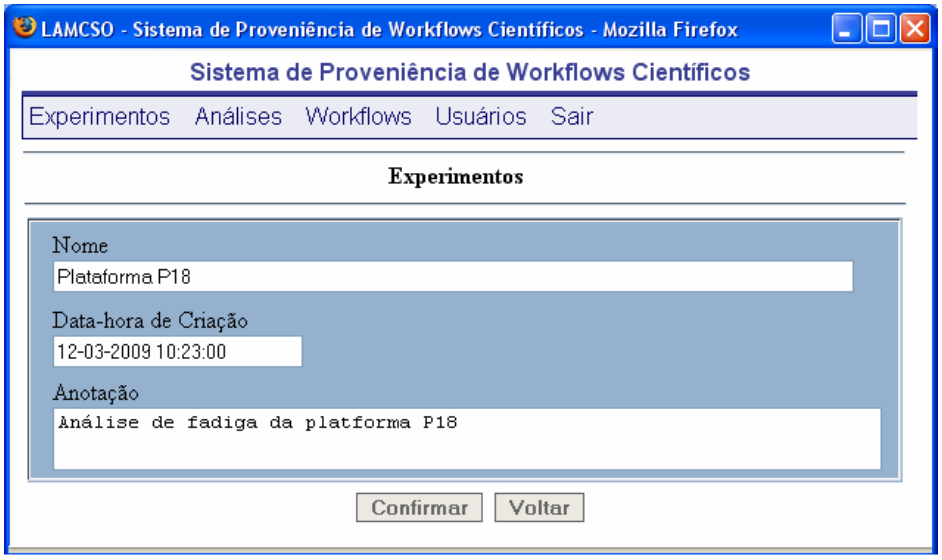


Figura 5.22: Tela que cadastra os experimento no sistema

Cada vez que um novo experimento é criado no sistema web de proveniência, ele monta uma estrutura de diretórios para o projeto do experimento a partir de um diretório raiz padrão, como será ilustrado no item 6.2.2 com o estudo de caso.

5.5 Execução do *Workflow*

Uma vez que o ambiente do *workflow* já foi configurado, o usuário deve obter os parâmetros de execução no sistema web, informá-los no *workflow* do Kepler e, então, executá-lo.

5.5.1 Configuração dos Parâmetros de Execução

Os parâmetros de execução do *workflow* apresentados na Figura 5.23 são os seguintes: **workingDir**, **usuário**, **experimento**, ***workflow*** e **arquivo**. O parâmetro **workingDir** indica o diretório raiz a partir do qual os programas do *workflow* do experimento científico são executados. O parâmetro **experimento** é o identificador do experimento que será realizado pelo *workflow* e é obtido consultando a tabela apresentada na Figura 5.19

O parâmetro ***workflow*** é o identificador do *workflow* (processo) que será realizado e é obtido consultando a tabela apresentada na Figura 5.21. No estudo de caso desta dissertação, ele corresponde ao identificador do *workflow* de Análise de Fadiga dos *Risers*. O parâmetro **usuário** é o identificador do usuário que irá executar o *workflow* e é obtido consultando a tabela apresentada da Figura 5.17. O parâmetro **arquivo** corresponde ao nome do próprio arquivo do Kepler em que o *workflow* foi configurado com estes determinados parâmetros de execução.

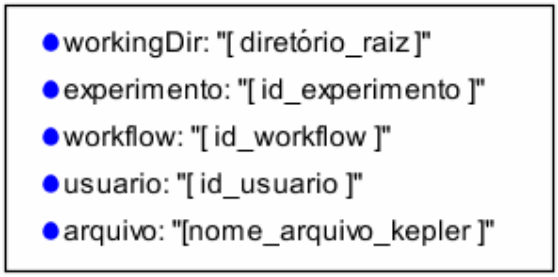


Figura 5.23: Parâmetros de excução do *workflow*

5.5.2 Execução do *Workflow*

Quando os parâmetros estiverem definidos, o usuário já pode executar o workflow. Então, ele deve clicar no botão 'Play' da barra de ferramentas. Enquanto o workflow estiver em execução, o botão ficará marcado com um fundo laranja. Assim como durante a execução de cada atividade, o ator correspondenter ficará marcado com uma borda amarela. A Figura 5.24 ilustra o *worflow* em execução. No instante em que a imagem foi capturada, o ANFLEX estava sendo executado.

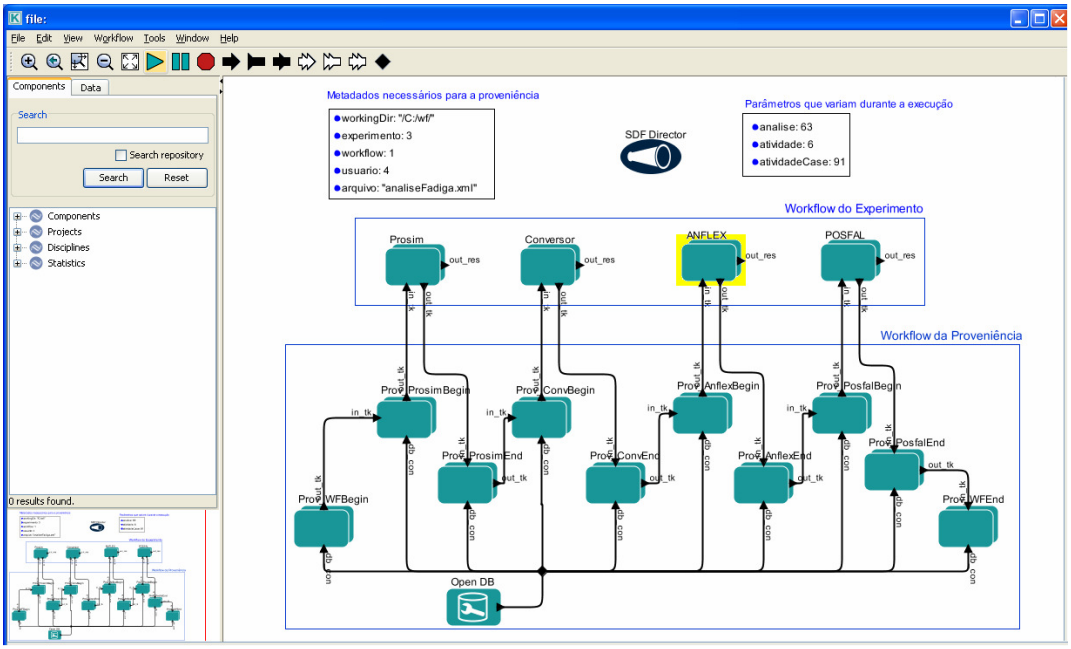


Figura 6.24: Workflow em execução

5.6 Consulta dos Resultados de Proveniência

Os resultados das análises podem ser acompanhados e consultados no sistema web de proveniência. As figuras a seguir ilustram as interfaces deste sistema, constituindo um tutorial para utilizá-lo.

5.6.1 Acompanhamento dos Experimentos

Ao clicar na opção Análises do menu superior, o sistema apresenta uma tabela que contém todas as execuções dos *workflows* cadastrados no sistema. Uma execução está obrigatoriamente associada a um experimento. Esta tela é ilustrada na Figura 5.25. As colunas da tabela apresentam o nome do experimento, a data do início da execução do *workflow*, o nome do engenheiro responsável pela análise e o nome do arquivo gerado no Kepler onde foram definidos os parâmetros para a análise. O sistema permite consultar, alterar, excluir e pesquisar análises já cadastradas. O cadastro das análises é realizado pelos componentes do Kepler que se conectam ao banco e executam os comandos SQL (Structured Query Language).

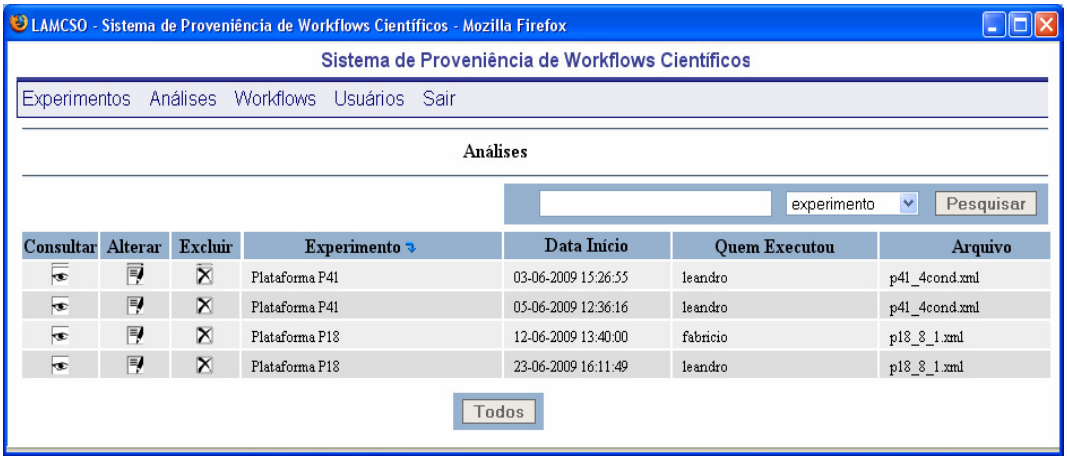


Figura 5.25: Tela que exibe as instâncias do *workflow* no sistema

A Figura 5.26 apresenta o formulário que define as informações referentes às análises. Ele é exibido quando as funcionalidades de alterar e excluir são acionadas. Na verdade esse formulário corresponde a uma consulta da proveniência gerada a partir da execução do *workflow* no Kepler. Os usuários apenas podem adicionar anotações à análise.

Figura 5.26: Tela que exibe os detalhes das instâncias do *workflow* no sistema

O formulário apresenta o nome do experimento associado á análise, a sigla do *workflow* que foi executado, o login do usuário que executou o *workflow*, a data e hora do início e do fim da execução e o nome do arquivo gerado no Kepler onde foram definidos os parâmetros para a análise. O sistema também apresenta na mesma interface deste formulário, os detalhes da execução de cada atividade, incluindo os seus dados de entrada e de saída.

As colunas da tabela que contém as atividades executadas no *workflow* apresentam o nome da ferramenta computacional que implementa a atividade, a data e hora do início e do fim de cada atividade e anotações específicas da atividade que são cadastradas pelos engenheiros.

Em relação aos dados envolvidos no *workflow*, são apresentas duas tabelas, uma contendo os dados de entradas de cada atividade (Figura 5.27), e outra contendo os dados de saída de cada atividade (Figura 5.28). Em cada tabela, é apresentado o nome da atividade associada ao dado, o nome do arquivo e a data da sua criação.

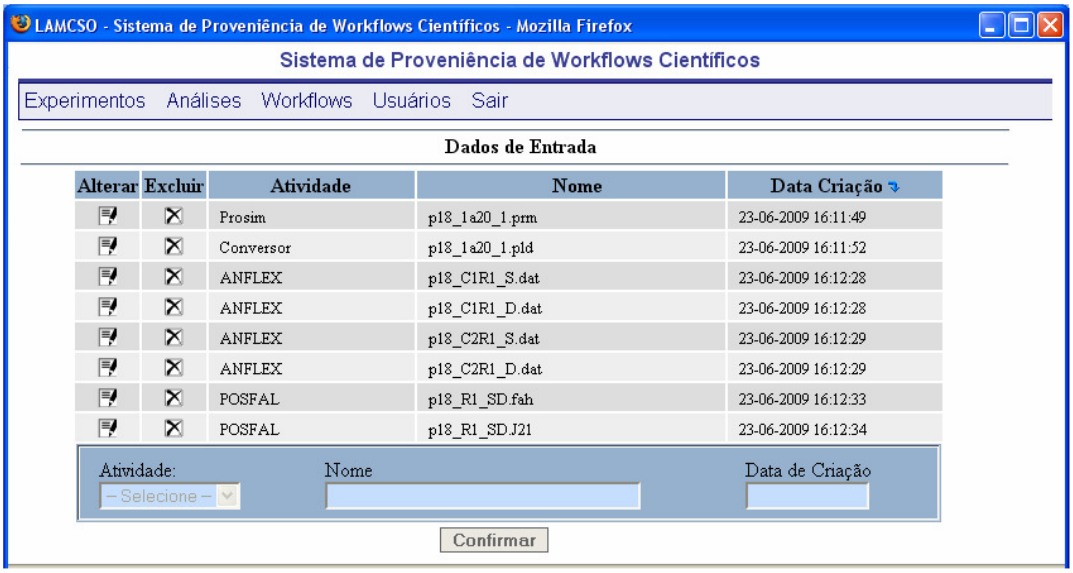


Figura 5.27: Tela que exibe os arquivos de entrada de cada atividade do *workflow*

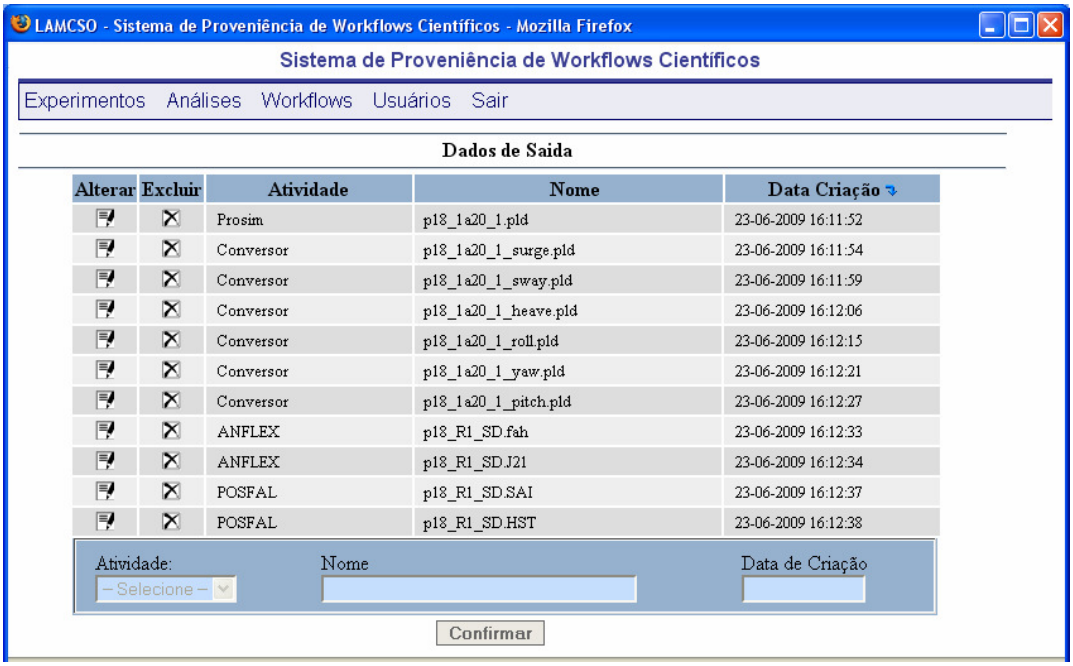


Figura 5.28: Tela que exibe os arquivos de saída de cada atividade do *workflow*

É importante ressaltar que esse sistema é só um protótipo para auxiliar a proveniência dentro do escopo do trabalho. Ele pode ser aperfeiçoado para exibir outros níveis de informações de proveniência e desenvolver relatórios que permitam gerar estatísticas envolvendo o tempo de execução dos *workflow* e de suas respectivas atividades, para que possam contribuir para otimizar os experimentos da engenharia *offshore*.

5.6.2 Resultado Final

A seção anterior apresentou os arquivos que são criados durante a execução do workflow. Entretanto, o objetivo principal da execução do workflow da análise de fadiga dos risers é obter a estimativa de vida útil à fadiga. O resultado final é obtido após a execução do POSFAL e está contido no arquivo com extensão **sai**.

Capítulo 6 - Estudo de Caso

6.1 Descrição do Sistema Offshore

A plataforma P18 é uma semi-submersível posicionada no Campo de Marlim, ancorada por 8 linhas de ancoragem e incorporando 72 *riser*s flexíveis e um SCR.. O casco tem quatro colunas e dois *pontoons*, como ilustrado na Figura 6.1. Foi projetado para operar em uma lâmina d`água de 1000 metros.

A Figura 6.1 também apresenta o sistema de referência estrutural da plataforma, considerado para as análises. O eixo estrutural Z da plataforma é vertical, com sentido positivo para cima, e origem na base dos *pontoons*. Os eixos estruturais X e Y formam o plano horizontal, com origem no CG da plataforma; o eixo X tem sentido positivo da popa para a proa. Considerando um sistema de coordenadas globais, o eixo X está alinhado com a direção Leste, e o eixo Y com a direção Norte.

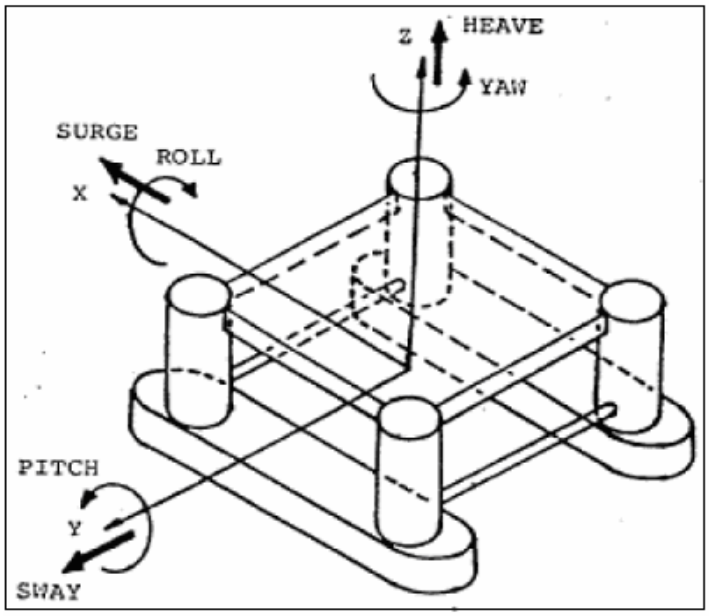


Figura 6.1: Plataforma P18

É necessário, para se proceder aos estudos propostos neste trabalho, descrever as características físicas e montar os modelos numéricos do casco da unidade flutuante, das linhas de ancoragem e dos risers (rígidos e flexíveis) e, ainda, conhecer os dados ambientais dependentes da locação da plataforma.

6.1.1 Dados do casco

A Tabela 6.1 apresenta algumas características do casco da plataforma P-18 e a Tabela 6.2 apresenta as coordenadas do centro de gravidade.

Tabela 6.1: Principais características do casco da P18

Característica	Valor
Comprimento total	101,00 m
Distância tranversa entre os centros das colunas	54,72 m
Boca total	88,10 m
Pontoons (2) (Comprimento x Boca x Pontal moldados)	89,68 x 16 x 9,10 m
Diâmetro moldado dos *bracings*	2,00m
Altura dos centros dos *bracing* em relação a quilha	13,10 m
Elevação do convés superior (Upper Deck)	43,90 m
Elevação do convés inferior (Lower Deck)	37,10 m
Calado de Operação	23, 10 m
Deslocamento (calado de operação)	36193,0 t
Calado de Trânsito	8,83 m
Deslocamento (calado de trânsito)	24530,0 t
Calado de Sobrevivência em Trânsito	18,40 t
Deslocamento (calado em trânsito)	32811,0 t
Calado de Sobrevivência em Operação	23,10 m
Deslocamento (calado de sobrevivência)	36,193 t
Peso da plataforma	333757,8 kN
Peso da plataforma em deslocamento	355053,3 kN

Os valores relacionados nestas tabelas não consideram as parcelas correspondentes aos risers e linhas de ancoragem. No modelo acoplado, onde todas as linhas são representadas por elementos finitos, a contribuição das mesmas na massa, amortecimento e rigidez do sistema é implicitamente considerada e transmitida à unidade flutuante através das forças de topo das linhas. Assim, a diferença no dado de peso da plataforma e deslocamento corresponde à componente vertical das cargas das linhas de ancoragem e risers.

Tabela 6.2: Coordenadas do CG da P18 para o calado de operação

	X (m)	Y (m)	Z(m)
CG	0,08	-0,35	19,30

6.1.2 Coeficientes de correnteza e vento

Os coeficientes de força de arraste de correnteza e vento da P18 foram fornecidos pela Petrobras, e serão apresentados nas tabelas e gráficos nesta seção. Estes coeficientes foram obtidos como resultado de ensaios em túnel de vento.

Tabela 6.3: Coeficientes de Correnteza no sistema de ref. do SITUA ($N.s^2/m^2$)

Incidência (grau)	Cd (Surge)	Cd (Sway)
0,00	3,2050E+05	0,0000E+00
22,50	4,7650E+05	1,9730E+05
45,00	4,8230E+05	4,8220E+05
67,50	3,1180E+05	7,5260E+05
90,00	0,0000E+00	7,0610E+05
112,50	-3,1180E+05	7,5260E+05
135,00	-4,8320E+05	4,8220E+05
157,50	-4,7650E+05	1,9730E+05
180,00	-3,2050E+05	0,0000E+00

Tabela 6.4: Coeficientes de Vento no sistema de ref. do SITUA (N.s2/m2)

Incidência (grau)	Cd (Surge)	Cd (Sway)
0,00	1,0150E+03	0,0000E+00
22,50	1,1470E+03	4,7500E+02
45,00	9,0600E+02	9,0600E+02
67,50	4,7000E+02	1,1350E+03
90,00	0,0000E+00	1,1120E+03
112,50	-4,5800E+02	1,1060E+03
135,00	-9,2000E+02	9,2000E+02
157,50	-1,1000E+03	4,5600E+02
180,00	-1,0610E+03	0,0000E+00

6.1.3 Linhas de Ancoragem

Como mencionado anteriormente, a plataforma P18 está ancorada por um total de 8 linhas de ancoragem, e incorpora 72 *riser*s flexíveis e um SCR. As figuras 6.2, 6.3 e 6.4 apresentam, respectivamente, as vistas frontal, lateral e superior das linhas de ancoragem da plataforma.

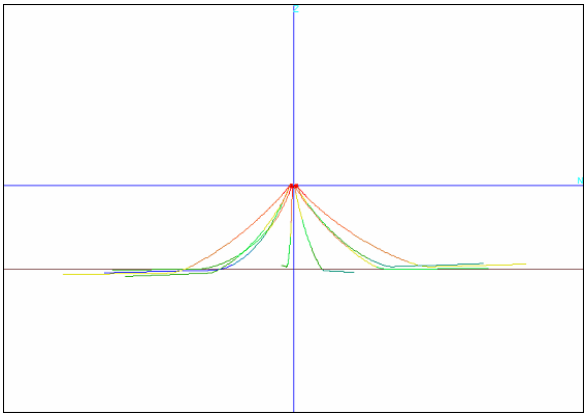


Figura 6.2: Vista frontal do sistema

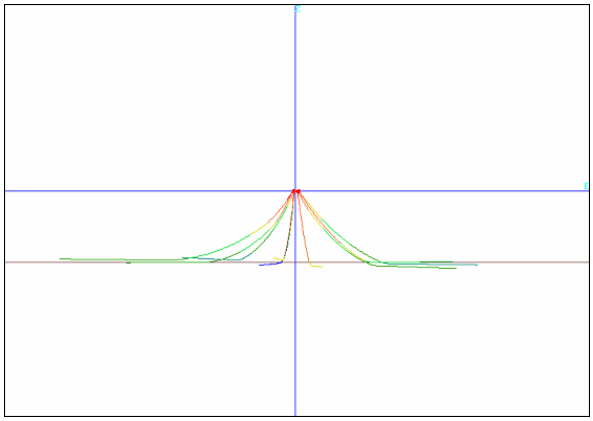


Figura 6.3: Vista lateral do sistema

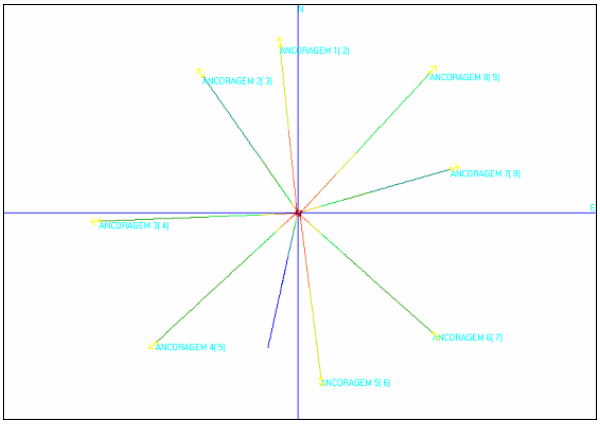


Figura 6.4: Vista superior do sistema

6.1.4 Riser Rígido – SCR

Nesta seção, são apresentadas as características do riser rígido (*SCR*) da P18, que possui 10” de diâmetro nominal. A Tabela 6.5 apresenta a disposição dos segmentos que compõem o riser rígido (basicamente, um segmento de *stress joint* e segmentos correspondentes ao corpo do *riser*). O corpo do riser incorpora uma camada de cobertura (*coating*) anticorrosão, com as características indicadas na Figura 6.5.

Tabela 6.5 – Disposição dos materiais no SCR

Segmento	Material	Comprimento
SJ	Stress Joint	1,103
Seg7	SCR EXP GAS 10"(S)	50
Seg6	SCR EXP GAS 10"(S)	100
Seg5	SCR EXP GAS 10"(S)	1000
Seg4	SCR EXP GAS 10"(S)	100
Seg3	SCR EXP GAS 10"(S)	300
Seg2	SCR EXP GAS 10"(S)	500
Anc	SCR EXP GAS 10"(S)	500

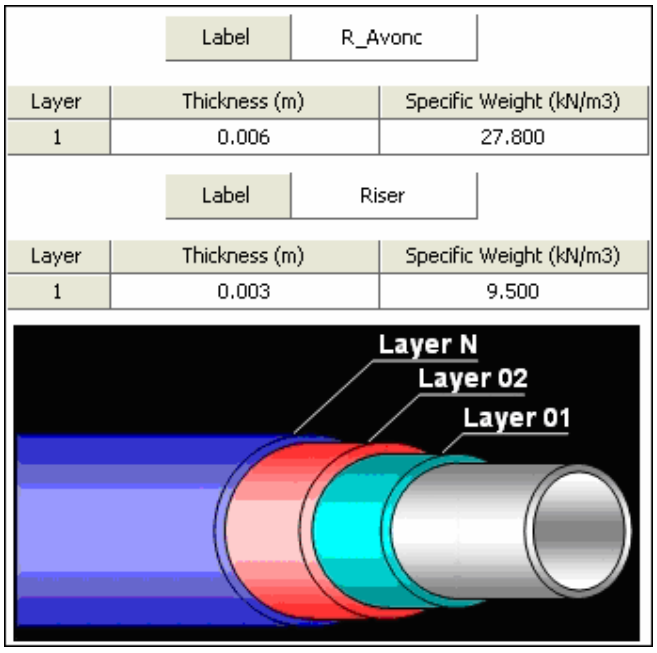


Figura 6.5: Camadas do *coating*

6.2 Execução do Workflow

6.2.1 Simplificações Adotadas

Como o objetivo deste capítulo é apenas ilustrar a aplicação do ambiente de *workflow* desenvolvido, e não executar um projeto real completo de análise de fadiga de onda de risers, algumas simplificações serão adotadas:

- O modelo elaborado para a P18 irá incorporar apenas o SCR descrito no item 6.1.4, desconsiderando a presença dos demais risers flexíveis;
- Para as análises dinâmicas, será tomado um tempo total de simulação de apenas 3600 segundos;
- Será considerado um número reduzido de somente duas condições ambientais que irão validar as estruturas de repetição (*loops*) da arquitetura do workflow.

6.2.2 Configuração do Ambiente do Workflow

Antes de executar o Workflow, devem ser executados os passos descritos no item 5.4 para configurar o ambiente. durante a configuração do ambiente do workflow, conforme apresentado na seção 5.4.4, é criada uma estrutura de diretórios para o experimento, para cada experimento criado no sistema web de proveniência. Para o caso em estudo foi definido o endereço **C:\WF** como o diretório raiz, e o experimento foi denominado **P18**. A árvore de diretórios gerada a partir do diretório P18 é apresentada na Figura 6.6.

Os dados que constituem os modelos representam os parâmetros de entrada da primeira atividade do workflow desenpenhada pelo prosim. A seguir são apresentados os arquivos que compõem o modelo do estudo de caso da plataforma P18.

- **P18op.wnf**, que contém os coeficientes hidrodinâmicos do casco gerados pelo Wamit;
- **batimetria2.fun**, que contém os dados de batimetria do fundo do mar;
- **p18_rev0_comb1.prm** e **p18_rev0_comb2.prm**, que contêm os dados do modelo do casco acoplado às linhas de ancoragem e ao riser rigído para análise com o Prosim de duas condições ambientais;

- **p18_rev0_comb1_Case01S.DAT** e **p18_rev0_comb1_Case01D.DAT**, que contêm os dados do SCR para as respectivas análises estática e dinâmica no ANFLEX.

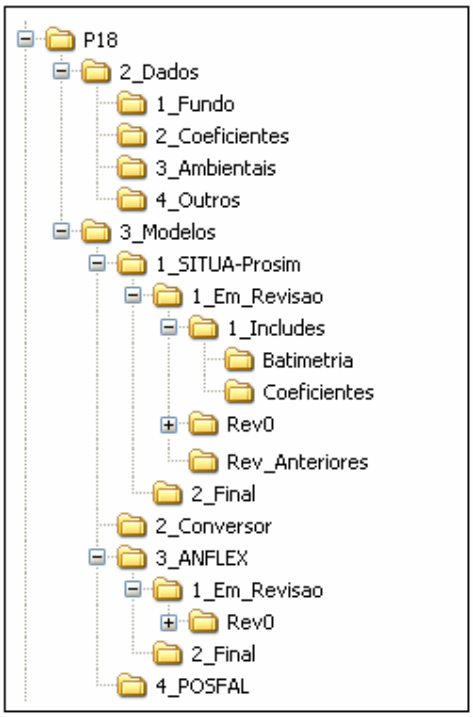


Figura 6.6: Árvore de diretórios do estudo de caso

6.2.3 Execução do Workflow e Resultados Obtidos

Para a execução do workflow propriamente dita, devem ser executados os passos descritos no item 5.5.

Neste estudo de caso, o *workflow* da análise de fadiga de onda da plataforma P18 foi executado em uma estação de trabalho com processador Dual Core de 1.8 GHz e 2 Gb de memória. Os engenheiros são capazes de acompanhar o andamento do experimento no sistema web de proveniência. Ao final da execução, ele tem a visão completa do tempo de execução de cada atividade do workflow. O registro da proveniência do experimento é apresentado na Figura 6.7.

O resultado final, em termos da estimativa de vida útil à fadiga do riser, está contido no arquivo com extensão **.sai** na pasta **\P18\3_Modelos\4_POSFAL**.

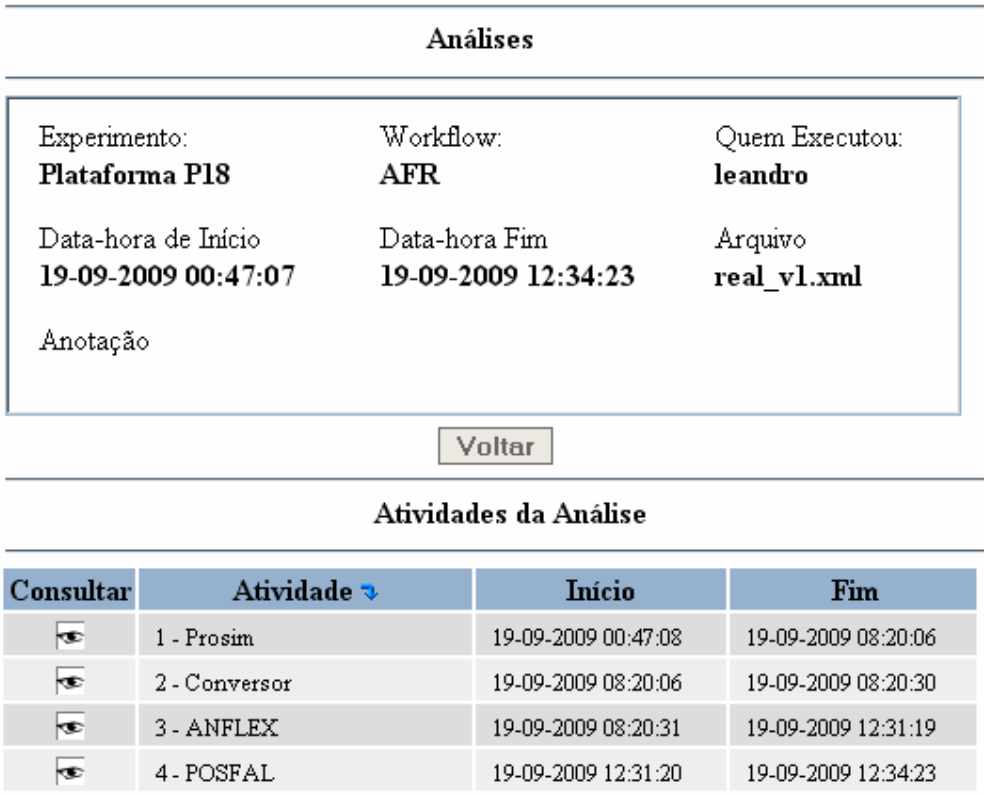


Figura 6.7: Registro da Proveniência do Experimento da Plataforma P18

Capítulo 7 - Considerações Finais

7.1 Conclusões

Diante da enorme quantidade de dados e resultados a serem gerenciados ao longo do processo de análise e projeto de fadiga de risers, grandes benefícios poderão ser obtidos com a aplicação de uma solução utilizando *workflow* científico, pois os engenheiros terão mais controle sobre as simulações realizadas e será mantida a proveniência dos dados ao longo da execução de todo o *workflow*.

A determinação da estimativa de vida útil de um *riser* envolve uma espiral de projeto que consiste em combinar diversos fatores críticos que contribuem para o desgaste das estruturas e, a partir da combinação desses fatores, os engenheiros podem especificar e aperfeiçoar os parâmetros de projeto de plataformas de extração de petróleo.

Em todos os casos é necessário repetir as análises por diversas vezes, submetendo o conjunto "unidade flutuante + linhas de ancoragem+ *risers*" a várias condições ambientais distintas. No contexto geral de *workflows* científicos, segundo Altintas [89] este processo repetitivo pode revelar muito a respeito do escopo que está sendo estudado e permitir que os cientistas refinem os parâmetros de configuração de forma que alcancem resultados mais eficientes e encontrem soluções baseadas na proveniência dos dados.

O gerenciamento da proveniência dos dados apresenta-se como um aspecto importante para a execução desses sistemas de *workflow* [89]. Neste contexto, cada execução do *workflow*, pode estar relacionada a atividades que visam encontrar a composição ideal de determinado tipo de *riser*, considerando, por exemplo, a relação entre o custo e o benefício esperado. Cada *workflow* executado pode representar uma configuração de *riser*. As variações dos respectivos parâmetros devem ser adequadamente registradas para proveniência.

Como já fora dito anteriormente, os *workflows* científicos têm sido aplicados em diversas áreas de pesquisa acadêmicas, como bioinfomática, química e astronomia, mas nesse trabalho estamos expandindo as aplicações desses conceitos para a Engenharia *Offshore*. Outro destaque que pode ser atribuído a esse trabalho é que ele mostrou a sua

natureza interdisciplinar uma vez que houve a integração do conhecimento de engenheiros de sistemas e engenheiros civis.

Os recentes avanços nos hardwares permitiram o aumento da velocidade de processamentos dos sistemas, o que resultou no aumento na quantidade de análises realizadas, que geram cada vez mais informação. Os principais benefícios que os *workflows* científicos agregam aos processos são os recursos que permitem organizar as informações produzidas, bem como consulta-las através do registro da proveniência.

A informação é o principal bem da sociedade atual, entretanto, é muito importante que se tenha o controle sobre ela. Uma grande quantidade de informação pode se transformar em um problema muito difícil de gerenciar. Informação desorganizada é informação inútil, uma vez que de nada adianta se ter informação se não há meios de acessá-la. A Google foi a pioneira a ter essa visão no mercado. O principal serviço que ela disponibiliza é uma ferramenta de busca na web que foi a tese de doutorado dos donos da empresa. O seu sistema de busca tem como objetivo permitir ao usuário acessar de modo mais eficiente o conhecimento desejado dentro do infinito repositório de informações que são disponibilizadas na internet a cada dia em todo o mundo.

A gerência da proveniência é uma área de pesquisa nova, mas está avançando rapidamente. Os pesquisadores estão avançando em várias direções nesta área, incluindo a capacidade de integrar a proveniência derivada de diferentes modelos, com os mecanismos de análise e visualização para exploração dos dados. As pesquisas também estão permitindo o crescimento de uma filosofia para colaboração entre os cientistas, que têm o potencial de mudar o modo como eles fazem ciência. Ao explorar a proveniência das informações em um ambiente colaborativo, os cientistas podem aprender, por exemplo, a acelerar os seus trabalhos e, potencialmente, reduzir o seu tempo de introspecção [44]. A "consciência coletiva", no contexto da exploração científica, pode evitar a duplicação de esforços e incentivar o progresso científico de modo contínuo, documentado e reproduzível.

A pesquisa dos sistemas de gerenciamento de *workflows* científicos existentes foi determinante para definirmos que o Kepler seria aquele utilizado nesse trabalho, apesar dele não possuir um recurso de proveniência. Chegamos à conclusão que atualmente não há uma ferramenta de *workflow* completa que possua os seguintes recursos: estruturas de controle e de repetição para implementar o *workflow*, recurso

próprio de proveniência, acesso a web services e ambientes de Grid. Estes recursos estão presentes em pelo menos uma das ferramentas observadas: Kepler, Vistrails, Triana e Taverna, porém nenhuma destas possuem todos eles. Cabe ressaltar que, apesar de o desenvolvimento ter sido realizado sobre o Kepler, as contribuições realizadas, tanto no *workflow* quanto no sistema de proveniência, não dependem de um SGWfC específico e podem ser acopladas a outros sistemas, gerando as mesmas vantagens.

7.2 Trabalhos futuros

7.2.1 Uso de *Workflows* em outras atividades de projeto de engenharia offshore

Este trabalho implementou um *workflow* de análise de fadiga de onda de *risers*. Este é somente um de muitos experimentos científicos realizados em engenharia *offshore*. Os conceitos de *workflows* podem ser aplicados para automatizar outros processos. Nesse sentido, está sendo modelado um meta-wokflow para a Rede Galileu da Petrobras. O meta-*workflow* será constituído de outros *workflows* além deste apresentado neste trabalho, como por exemplo, os processos relacionados ao projeto de sistemas de ancoragem, e os relacionados à análise de risers sob condições ambientais extremas.

7.2.2 Implementação em Ambientes de Processamento Paralelo

A implantação da cultura para utilização de *workflows* científicas abre novas idéias e possibilidades para a automação dos processos. Podem-se estudar meios para distribuir o processamento das atividades do *workflow*. No *workflow* de análise de fadiga de *risers* mesmo, há muitas possibilidade de paralelizar a execução dos aplicativos. O SITUA pode gerar vários arquivos de modelagem de extensão **prm**, cada um contendo uma determinada quantidade de condições ambientais. Então, podem ser executadas várias instâncias do Prosim, para gerar os movimentos do sistema acoplado que sofreu ação das respectivas condições ambientais. Alguns trabalhos junto ao Kepler e VisTrails já vem sendo realizados nessa direção no PESC-COPPE [20,100,101] e que podem agora ser explorados no SITUA.

Da mesma maneira, podem ser executadas várias instâncias do Conversor, para converter esses arquivos de movimentos de extensão **pld** nos seis arquivos de movimentos no formato **thf**. Também podem ser criadas várias instâncias do POSFAL para realizar as análises de fadiga dos *risers*.

7.2.3 Associação com a Definição do Padrão GXML para a Rede Galileu

No estudo do *workflow* de engenharia *offshore*, os arquivos de entrada e de saída dos aplicativos não seguem um padrão definido. Eles contêm uma série de valores que não possuem uma informação semântica associada. Essas ausências de informações podem dificultar a implementação dos *workflows*. Portanto, nesse contexto de *workflows*, aconselha-se adotar uma linguagem ou um padrão de intercâmbio de dados para representar os arquivos de entradas e saída. Os arquivos no formato INI ou CSV foram largamente utilizados para este fim, entretanto, recentemente eles têm sido substituídos pelos arquivos no formato XML. Neste sentido, atualmente, no contexto da Rede Galileu, estão sendo executadas atividades no sentido de empregar um padrão único de arquivos de entrada e saída, no formato denominado GXML (Galileu eXtended Markup Language.

7.2.4 Uso de Sistemas que demandam Interfaces Gráficas

Programas que demandam uma interação humano-interface e dependem de recursos gráficos próprios representam hoje uma barrreira para ser enfrentada pelos SGWfC existentes. Avanços devem ser realizados nessa área ou a geração de soluções paliativas devem ser desenvolvidas. Por exemplo, além dos programas descritos anteriormente, ainda há o POSSINAL, outro programa que em seu estágio atual só pode ser utilizado pela interface. Ele é capaz de gerar estatísticas e espectros de resposta dos movimentos, da tração ou do esforço que são produzidos, por exemplo, pelo SITUA. Também pode funcionar como um filtro do arquivo das séries de movimentos que seria processada no CONVERSOR e, posteriormente, utilizada pelo ANFLEX. Alguns softwares utilizados nos experimentos científicos já estão sendo utilizados há muito tempo. Eles foram desenvolvidos numa época em que os analistas e programadores não tinham uma grande preocupação com a documentação e padronização dos sistemas. Muitas vezes o conhecimento é perdido no tempo uma vez que ele não é repassado entre as pessoas que os utilizam durante um determinado tempo e seus substitutos. A utilização das boas práticas de programação e de padrões de desenvolvimento são essenciais para aumentar a vida útil de um software.

Referencias Bibliográficas

[1] Espaço Conhecer – Petrobras. Disponível em <http://www2.petrobras.com.br/EspacoConhecer/HistoriaPetroleo/apresentacao.asp> Acesso em 13 de junho de 2009.

[2] Petrobras. Disponível em <http://www2.petrobras.com.br/> Acesso em: 13 de junho de 2009.

[3] Poder Naval. Disponível em: http://www.naval.com.br/blog/?p=4409. Acesso em: 13 de junho de 2009.

[4] Outra Política: Disponível em: http://outrapolitica.wordpress.com/2008/07/15/petroleo-royalties-e-fundo-solidario-para-o-brasil-de-amanha/. Acesso em: 13 de junho de 2009.

[5] A grande produção offshore de petróleo no Brasil. Disponível em: http://ciencia.hsw.uol.com.br/exploracao-petroleo-mar9.htm. Acesso em: 13 de junho de 2009.

[6] LEMOS, M., 2004, Workflow para bioinformática. Tese de D.Sc., PUC-RJ - Departamento de Informática, Rio de Janeiro.

[7] SERRA, S. M. C., 2005, *Gerência de workflows em bioinformática*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[8] SENRA, S. F., 2004, *Metodologias de Análise e Projeto Integrado de Sistemas Flutuantes para Explotação de Petróleo Offshore*. Tese de D.Sc., COPPE/UFRJ – Programa de Engenharia Civil, Rio de Janeiro.

[9] CORREA, F.N., 2003, *Aplicação de Metodologias Híbridas em Estudos Paramétricos sobre o Comportamento de Sistemas Offshore*. Tese de M.Sc., COPPE/UFRJ, Programa de Engenharia Civil, Rio de Janeiro.

[10] LIMA, A. L., 2006, *Avaliação de Metodologias de Análise de Unidades Estacionárias de Produção de Petróleo Offshore*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia Civil, Rio de Janeiro.

[11] BAHIENSE, R. A., 2007, *Implementação e Avaliação de uma Metodologia Fortemente Acoplada para Análise de Sistemas Flutuantes Offshore*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia Civil, Rio de Janeiro.

[12] CORREA, F.N., 2008, *Ferramentas Computacionais para Análise Acoplada de Sistemas Offshore*. Tese de D.Sc., COPPE/UFRJ, Programa de Engenharia Civil, Rio de Janeiro.

[13] GIRÓN, A. R. C., 2009, *Aplicação de Metodologias de Projeto Integrado de Sistemas de Ancoragem e Risers na Explotação de Petróleo Offshore*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia Civil, Rio de Janeiro.

[14] CAVALCANTI, M. C. R., 2003, *Gerência de Recursos Científicos: Apoiando a Realização de Experimentos In Silico*. Tese de D.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[15] DOS SANTOS, R. T., 2006, *O Ambiente 10+C para Definição e Execução de Workflows In-Silico através de Serviços Web*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[16] DIGIAMPIETRI, L. A., 2007, *Gerenciamento de workflows científicos em bioinformática*. Tese de D.Sc., UNICAMP – Instituto de Computação, Campinas.

[17] VON HELD, V. S., 2007, *Um ambiente para gerenciamento e execução de experimentos de expressão gênica com microarranjos*. Tese de M.Sc., COPPE/ UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[18] OLIVEIRA, D. C. M., 2008, *MiningFlow: um sistema de Workflow para apoiar o Processo de Descoberta do Conhecimento em Textos*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro

[19] BATISTA, A. M. S., 2008, *Gerência de Dados Genômicos em Sistemas de Workflows de Bioinformática*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[20] BARROS, P. M., 2009, *Gerência de Distribuição na Execução de Workflows Cientificos em Bioinformática.*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[21] COOPETRÓLEO – Cooperativa de Profissionais da Indústria do Petróleo Ltda. Disponível em: http://www.coopetroleo.com.br/pfpp.htm. Acesso em: 13 de junho de 2009.

[22] Fórum de Mergulho. Disponível em: http://www.forum-mergulho.com/print.html &client=printer&f=24&t=240. Acesso em: 13 de junho de 2009.

[23] Governo do Estado do Espírito Santo Disponível em: http://www.es.gov.br/site/ noticias/show.aspx?noticiaId=99657527. Acesso em: 13 de junho de 2009.

[24] Fotothing – Photo Sharing. Disponível em: http://www.fotothing.com/. Acesso em: 13 de junho de 2009.

[25] WAMIT, 1995, *WAMIT - A Radiation-Difraction Panel Program for Wave-Body Interactions*. Version 5.3, User Manual, Department of Ocean Engineering Massachusettes Institute of Technology.

[26] JACOB, B.P., 2006, *Programa Prosim: Simulação Numérica do Comportamento de Unidades Flutuantes Ancoradas.* Versão 3.0, Manual Teórico, COPPE/UFRJ, Programa de Engenharia Civil, Rio de Janeiro.

[27] MOURELLE, M.M., GONZALEZ, E.C., JACOB, B.P., Setembro/1995, *ANFLEX: Computational System for Flexible and Rigid Risers Analysis*. Proceedings of the 9th International Symposium on *Offshore* Engineering, Brazil *Offshore* 95, Rio de Janeiro.

[28] MOURELLE, M.M., GONZALEZ, E.C., SIQUEIRA, M.Q.- *Manual Teórico do Programa ANFLEX*, Relatório RL064-2001, CENPES/PDP/MC, 2001.

[29] ANFLEX, 2002, *ANFLEX - Análise Não Linear de Riser e Linha de Ancoragem*. Versão 5.1.8, Manual de Entrada de Dados, Petrobras/Cenpes.

[30] SOUZA, L.F.A., 1998, *Análise de Riser Rígido em Catenária com Ênfase na Verificação à Fadiga*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia Naval e Oceânica, Rio de Janeiro.

[31] POSFAL – Pós-Processador de Fadiga Aleatória – Manual de entrada de dados. Doc. 2.0, CENPES/DIPREX/SEDEM, Dezembro/1996.

[32] TORRES, A.L.F.L., SAGRILO, L.V.S., SIQUEIRA, M.Q. et al., 1995. *A Procedure for Random Fatigue Analysis of Offshore Structures*. Proceedings of the 9th International Symposium on *Offshore* Engineering, Brazil *Offshore* 95, Rio de Janeiro.

[33] WAINER, J., WESKE, M., VOSSEN, G., MEDEIROS, C.B., 1996. *Scientific Workflow Systems*.In Proceedings of the NSF Workshop on Workflow and Process Automation Information Systems.

[34] TELECKEN, T., 2003, *Definição de Processos de Workflow*.

[35] HOLLINGSWORTH D., 1995, *Workflow Management Coalition The Workflow Reference Model*, Workflow Management Coalition, TC00-1003 Issue 1.1.

[36] BONIFÁCIO, A., BARBOSA, A., BARBOSA, F., 2008, *Análise do uso de um sistema de workflow científico para modelagem de sistemas dinâmicos deformáveis*. UFJF - Mestrado Multidisciplinar em Modelagem Computacionais, 36036-300, Juiz de Fora.

[37] YAMAMOTO, W. T., 2007, *Fluxo de Dados em Workflows Científicos: O Sistema Kepler*. Monografia de B.Sc., USP – Instituto de Matemática e Estatística, Departamento de Ciência da Computação, São Paulo.

[38] CAVALCANTI, M.C.R., TARGINO, R., BAIÃO, F., RÖSSLE, S.C., BISCH, P. M., PIRES, P. F., CAMPOS, M. L. M., MATTOSO, M. L. Q., 2005, *Managing structural genomic workflows using Web services*. Data & Knowledge Engineering, 53(1):45–74

[39] PASTORELLO Jr, G. Z., MEDEIROS C. B., SANTANCHÈ A., 2007, *Applying Scientific Workflows to Manage Sensor Data*. In 2007 e-Science Workshop.

[40] FOLLI, A., YOSHIMURA, B., NERY, G., *Adequabilidade do Kepler para pacotes de bioinformatica do R*. Relatório de Pesquisa, USP - Instituto de Matemática e Estatística, São Paulo.

[41] SERRA, S., SEABRA, F., DAHIS, R., CAMPOS M. L., MATTOSO, M. L. Q., 2008, *Controles de Fluxo Explícitos em Workflows Científicos*. In II e-Science Workshop.

[42] GODERIS, A., BROOKS, C., ALTINTAS, I., LEE, E. A., GOBLE, C., 2007, *Composing Different Models of Computation in Kepler and Ptolemy II*. Em Proceedings 2nd International Workshop on Workflow systems in e-Science in conjunction with ICCS 2007.

[43] MOREAU, L., FREIRE, J., FUTRELLE, J., McGRATH, R., MYERS, J., PAULSON, P., 2007, *The Open Provenance Model*. Technical Report UNSPECIFIED, ECS, University of Southampton.

[44] FREIRE, J., KOOP, D., SANTOS, E., SILVA, C., 2008, *Provenance for Computational Tasks: A Survey*. In Computing in Science & Engineering. ISSN: 1521-9615 DOI: 10.1109/MCSE.2008.79.

[45] CALLAHAN, S., FREIRE, J., SANTOS, E., SCHEIDEGGER, C., SILVA, C., VO, H., 2006, *Using Provenance to Streamline Data Exploration through Visualization.* Technical Report, SCI Institute, University of Utah.

[46] OLIVEIRA, F., MURTA, L., WERNER, C., MATTOSO, M. L. Q., 2008, *Using Provenance to Improve Workflow Design*. Second International Provenance and Annotation Workshop - IPAW.

[47] SIMMHAN, Y., PLALE, B., GANNON, D., 2005, *A Survey of Data Provenance in e-Science* In SIGMOD Rec., Vol. 34, No. 3. (September 2005), pp. 31-36.

[48] COHEN, S., COHEN-BOULAKIA, S., DAVIDSON, S. B., 2006, *Towards a Model of Provenance and User Views in Scientific Workflows*. Em Lecture Notes in Computer Science, Volume 4076, Julho de 2006, pages 264-279.

[49] SERRA, S. M. C., CAMPOS, M. L. M., MATTOSO, M. L. Q., 2009, *Towards a Taxonomy of Provenance in Scientific Workflow Management Systems*. Em: IEEE 2009 Third International Workshop on Scientific Workflows (SWF 2009), Los Angeles. 7th IEEE International Conference on Web Services (ICWS 2009).

[50] DAVIDSON, S., FREIRE, J., 2008, *Provenance and Scientific Workflows: Challenges and Opportunities*. Em Proceeding of ACM SIGMOD International Conference on Management of Data.

[51] CALLAHAN S., FREIRE, J., SCHEIDEGGER, C., SILVA, C., 2008, *Towards Provenance-Enabling ParaView*. Em Proceedings of IPAW, pp. 120-127.

[52] Taverna Project Website. Disponível em: http://taverna.sourceforge.net Acesso em: 26 de maio de 2009.

[53] Vistrails Wiki. Disponível em: http://www.vistrails.org/index.php. Acesso em: 20 de maio de 2009.

[54] Provenance Challenge Wiki. Disponível em. http://twiki.ipaw.info/bin/view/Challenge/. Acesso em: 12 de junho de 2009.

[55] DIGIAMPIETRI, L.A., BARGA, R.S., 2007, *Automatic capture and efficient storage of e-science experiment provenance*. Em Concurrency and Computation: Practice and Experience,

[56] DIGIAMPIETRI, L.A., MEDEIROS, C.B., SETÚBAL, J.C., BARGA, R.S., 2007, *Traceability Mechanisms for Bioinformatics Scientific Workflows*. Em Proceedings of the AAAI2007's Workshop on Semantic e-Science (SeS07), pages 26–33, Vancouver, Canada.

[57] DA SILVA, F. N., 2006, *In services: Um sistema para gerenciamento de dados intermediários em workflows científicos na bioinformática*. M.Sc., IME – Engenharia de Sistemas e Computação, Rio de Janeiro.

[58] BOWERS, S., McPHILLIPS, T., LUDAESCHER, B, COHEN, S., DAVIDSON, S., 2006, *A Model for User-Oriented Data Provenance in Pipelined Scientific Workflows*. Em Lecture Notes in Computer Science, Volume 4145, Provenance and Annotation of Data, 2006, pages 133-147.

[59] DA SILVA, F. N., CAVALCANTI, M. C., DÁVILA, A. M. R., 2006, *In Services: Data Management for In Silico Workflows*. Em IEEE Computer Society.

[60] VAN DER AALST, W. M. P., HOFSTEDE, A. H. M., 2005, *YAWL: Yet another workflow language*. Em Information Systems, 30(4):245–275.

[61] MUNISWAMY-REDDY, K., HOLLAND D., BRAUN U., SELTZER, M., 2006, "Provenance-Aware Storage Systems". Em Annual Tech '06: USENIX Annual Technical Conference.

[62] MATTOSO, M. L. Q.; WERNER, C. M. L., TRAVASSOS, G. H.; MURTA, L., BRAGANHOLO, V., OGASAWARA, E., OLIVEIRA, F., MARTINHO, W., 2009, "Desafios no apoio à composição de experimentos científicos em larga escala". Em Seminário Integrado de Software e Harware, 2009, Bento Gonçalves. SEMISH, Congresso da SBC, 2009.

[63] FREIRE, J., SILVA C., 2008, *Simplifying the Design of Workflows for Large-Scale Data Exploration and Visualization*. Em Microsoft eScience Workshop.

[64] LINS, L., KOOP, D., ANDERSON, E., CALLAHAN, S., SANTOS, E., SCHEIDEGGER, C., FREIRE, J., SILVA, C., 2008, *Examining Statistics of Workflow Evolution Provenance: A First Study*. Em Proceedings of International Conference on Scientific and Statistical Database Management (SSDBM).

[65] DAVIDSON, S., FREIRE, J., EYAL, A., LUDAESCHER, B., McPHILLIPS, T., BOWERS, S., ANAND, M., 2007, *Provenance in ScientificWorkflow Systems*. Em IEEE Data Engineering Bulletin, 32(4), pp. 44-50, 2007.

[66] ELLKVIST, T., KOOP, D., ANDERSON, E., FREIRE, J., SILVA, C., 2008, *Using Provenance to Support Real-Time Collaborative Design of Workflows*, Em Proceedings of IPAW, pp. 266-279.

[67] MARINHO, A., MURTA, L. ; WERNER, C. M. L., BRAGANHOLO, V. P., CRUZ, S.M.S., MATTOSO, M. L. Q., 2009, *A Strategy for Provenance Gathering in Distributed Scientific Workflows*. Em: IEEE 2009 Third International Workshop on Scientific Workflows (SWF 2009), Los Angeles. 7th IEEE International Conference on Web Services (ICWS 2009).

[68] MATTOSO, M. L. Q., SERRA, S. M. C., MATTOS, A., SILVA, F. C., RUBERG, N., 2008, *Gerência de Workflows Científicos: Uma Análise Crítica no Contexto da Bioinformática*. Trabalho Técnico, COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[69] VAN DER AALST, W.M.P., 1998, *The application of petri nets to workflow management*. Em The Journal of Circuits, Systems and Computers, 8, 21–66.

[70] CARDOSO, L. F., 2003, *Bill of Experiments: Um Sistema Colaborativo para Explicitação, Reuso e Planejamento de Workflows Científicos*. Tese de M.Sc., COPPE/UFRJ – Programa de Engenharia de Sistemas e Computação, Rio de Janeiro.

[71] OGASAWARA, E., MURTA L., WERNER C., MATTOSO, M., 2008, *Linhas de Experimento: Reutilização e Gerência de Configuração em Workflows Científicos*. Em XXIII Simpósio Brasileiro de Banco de Dados (SBBD), XXII Simpósio Brasileiro de Engenharia de Software (SBES), Campinas, Brasil.

[72] MEDEIROS, C. B., PEREZ-ALCAZAR, J., DIGIAMPIETRI, L., PASTORELLO Jr, G. Z., SANTANCHÈ, A., TORRES, R. S., MADEIRA, E., BACARIN, E., 2005, *WOODSS and the Web: Annotating and Reusing Scientific Workflows*. Em SIGMOD Record, 34(3):18–23.

[73] CALLAHAN, S., FREIRE, J., SILVA, C., 2007, *Provenance for visualizations: Reproducibility and Beyond*. Em IEEE Computing in Science & Engineering, 9(5), pp. 82-90.

[74] The MayaVi Data Visualizer. Disponível em: http://mayavi.sourceforge.net. Acesso em: 23 de maio de 2009.

[75] Paraview Open Source Scientific Visualization. Disponível em: http://www.paraview.org. Acesso em: 23 de maio de 2009.

[76] CALLAHAN, S., FREIRE, J., SANTOS, E., SCHEIDEGGER, C., SILVA, C., VO, H., 2006, *VisTrails: visualization meets data management*. Em: ACM SIGMOD, p. 745–747.

[77] CALLAHAN, S., FREIRE, J., SANTOS, E., SCHEIDEGGER, C., SILVA, C., 2005, *Managing the Evolution of Dataflows with VisTrails*. Em Proceedings of IEEE Workshop on Workflow and Data Flow for Scientific Applications, 2006.

[78] CALLAHAN, S., BAVOIL, L., CROSSNO, P., FREIRE, J., SANTOS, E., SCHEIDEGGER, C., SILVA, C., VO, H., 2005, *VisTrails: Enabling Interactive Multiple-View Visualizations*. Em Proceedings of IEEE Visualization, 2005.

[79] MyGrid. Disponível em: http://www.mygrid.org.uk/. Acesso em: 26 de maio de 2009.

[80] MISSIER, P., 2008, *Granular workflow provenance in Taverna*. In Symposium on Provenance in Scientific Workflows, Salt Lake City, Oct. 2008

[81] OOIN, T., ADDIS, M., FERRIS, J., MARVIN, D., SENGER, M., GREEENWOOD, M., et al., 2004. *Taverna: a tool for the composition and enactment of bioinformatics workflows*. Em Bioinformatics vol. 20 issue 17, Oxford Univ Press.

[82] Resource Description Framework (RDF). Disponível em: http://www.w3.org/ TR/rdf-concepts/. Acesso em 27 de maio de 2009.

[83] Triana – Open Source Problem Solving Software. Disponível em: http://www.trianacode.org. Acesso em 28 de maio de 2009.

[84] SHIELDS, M., TAYLOR, I., 2004, *Programming Scientific and Distributed Workflow with Triana Services*. Em Concurrency and Computation: Practice & Experience Volume 18 , Issue 10 (Agosto de 2006) Workflow in Grid Systems pp: 1021-1037 ISSN:1532-0626

[85] SHANKAR, S., KINI, A., DEWITT, D.J., NAUGHTON, J., 2006, *Integrating databases and workflow systems*. Em ACM SIGMOD Record Volume 34 ,Issue 3 (Setembro de 2005) Special section on scientific workflows pp.5 -11 ISSN:0163-5808

[86] The Kepler Project. Disponível em: http://kepler-project.org. Acesso em: 29 de maio de 2009.

[87] WOOLLARD, D., 2008, *Supporting the Engineering Aspects of e-Science Through Workflow Services*. Qualifying Examination Report, University of Southern California.

[88] LUDAESCHER, B., ALTINTAS, I., BERKLEY, C., HIGGINS, D., JAEGER, E., JONES, M., LEE, E., TAO, J., ZHAO, Y., 2006, *Scientific Workflow Management and the Kepler Systems*. Em Concurrency and Computation: Practice & Experience, 18(10), pp. 1039-1065, 2006.

[89] ALTINTAS, I., BIRNBAUM, A., BALDRIDGE, K. K., SUDHOLT, W., MILLER, M., AMOREIRA, C., POTIER, Y., LUDAESCHER, B., 2005, *A Framework for the Design and Reuse of Grid Workflows*. Em Scientific Applications of Grid Computing: First International Workshop, SAG 2004, Beijing, China, September 20-24, 2004

[90] ALTINTAS, I, BERKLEY, C., JAEGER, E., JONES, M., LUDAESCHER, B., MOCK, S., Junho de 2004, *Kepler: An Extensible System for Design and Execution of Scientific Workflows*, Em 16th Intl. Conference on Scientific and Statistical Database Management(SSDBM), Santorini Island, Greece.

[91] ALTINTAS, I., BARNEY, O., CHENG, Z., CRITCHLOW, T., LUDAESCHER, B., PARKER, S., SHOSHANI, A., VOUK, M., 2006, *Accelerating the scientific exploration process with scientific workflows*, Em Institute of Physics Publishing - Journal of Physics: Conference Series 46 (2006) 468–478

[92] McPHILLIPS, T., BOWERS, S., 2006, *An Approach for Pipelining Nested Collections in Scientific Workflows*. Em SIGMOD Record, Vol. 34, No. 3, Sept. 2005.

[93] ALTINTAS, I., BARNEY, O., JAEGER, E., 2006, *Provenance Collection Support in the Kepler Scientific Workflow System*. Em 1st International Workshop on Computational Chemistry and Its Application in e-Science in conjunction with ICCS 2006.

[94] VAN DER AALST, W., HOSFTEDE, A. H. M., KIEPUSZEWSKI, B., BARROS, A. P., 2003, *Workflow Patterns.Distributed and Parallel Databases*. Em: Distributed and Parallel Databases 14 (1): pp. 5--51. DOI:10.1023/ A:1022883727209.

[95] BOWERS, S., LUDAESCHER, B., NGU, A. H. H., CRITCHLOW, T., 2006, *Enabling Scientific Workflow Reuse through Structured Composition of Dataflow and Control-Flow*. Em ICDE´06, p. 70-80.

[96] PostgreSQL: Disponível em: http://www.postgresql.org/. Acesso em 12 de junho de 2009.

[97] PostgreSQL Reference Manual Volume 1: SQL Language Reference for version 8.2.4 June 2007. Editora: Network Theory LTD. ISBN 0-9546120-2-7

[98] PostgreSQL Reference Manual Volume 2: Programming Guide for version 8.2.4 June 2007. Editora: Network Theory LTD. ISBN 0-9546120-3-5

[99] PHP: Hypertext Preprocessor. Disponível em: http://www.php.net. Acesso em 12 de junho de 2009.

[100] SERRA, S.M.C.; BARROS, P.; BISCH, P.; CAMPOS, M. L. M.; MATTOSO, M. L. Q., 2008, *Provenance services for distributedworkflows*. Em 8th IEEE International Symposium on Cluster Computing and the Grid, 2008, Lyon. CCGrid´08, 2008. p. 526-533.

[101] SERRA, S. M. C.; SILVA, F. N.; GADELHA Jr., L. M .R.; CAVALCANTI, M. C.; CAMPOS, M. L. M.; MATTOSO, M. L. Q., 2008, *A Lightweight Middleware Monitor for Distributed Scientific Workflows*. Em International Workshop on Workflow Systems in e-Science, Lyon. 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2008), p. 693-698.

Apêndices

Apêndice A – Dicionário de Dados

A tabela abaixo apresenta uma descrição sucinta de cada entidade do banco de dados utilizado para controlar a proveniência. As outras tabelas a seguir contêm uma descrição de cada atributo das entidades do mesmo banco de dados.

Tabela A.1: Entidades

Nome	Descrição
Atividade	Representa as atividades dos workflows.
AtividadeCase	Representa as instâncias das atividades das execuções dos workflows.
Dado	Contém todos os arquivos envolvidos durante a execução de um workflow.
DadoEntrada	Contém os arquivos que são parâmetros de entrada para uma atividade na execução de um workflow.
DadoSaida	Contém os arquivos que foram gerados por uma atividade na execução de um workflow.
Experimento	Representa as análises científicas realizadas pelos engenheiros.
Usuario	Contém os engenheiros que executam os workflows científicos.
Workflow	Representa os workflows científicos executados pelos engenheiros.
WorkflowCase	Representa as execuções dos workflows.

Tabela A.2: Atividade

Chave	Nome	Tipo	Descrição
PK1	id_atividade	INTEGER UNSIGNED NOT NULL	Identificador da tabela Atividade.
FK	id_workflow	SMALLINT UNSIGNED NOT NULL	Identificador da tabela Workflow, representa o workflow no qual a atividade é executada.
	nome	VARCHAR(100) NOT NULL	Nome da atividade
	ordem	SMALLINT UNSIGNED NOT NULL	Ordem em que a atividade é executada dentro do workflow.
	ferramenta	VARCHAR(50) NOT NULL	Nome da ferramenta computacional que executa a atividade.

Tabela A.3: AtividadeCase

Chave	Nome	Tipo	Descrição
1	id_atividadeCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela AtividadeCase.
F	id_atividade	INTEGER UNSIGNED NOT NULL	Identificador da tabela Atividade, representa a atividade que executada no workflow
F	id_workflowCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela WorkflowCase, representa a execução do workflow na qual a atividade foi realizada.
	dh_inicio	TIMESTAMP NOT NULL	Data e hora em que a atividade teve início.
	dh_fim	TIMESTAMP	Data e hora em que a atividade foi concluída.
	anotacao	TEXT	Anotações ou observações feitas pelos pesquisadores sobre a atividade da execução do workflow.

Tabela A.4 : Dado

Chave	Nome	Tipo	Descrição
1	id_dado	INTEGER UNSIGNED NOT NULL	Identificador da tabela Dado.
F	id_workflowCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela WorkflowCase, representa a instância da execução do workflow que utilizou o arquivo.
	nome	VARCHAR(100) NOT NULL	Nome do arquivo envolvido na execução do workflow.
	dh_criacao	TIMESTAMP NOT NULL	Data e hora da criação do arquivo.

Tabela A.5 : DadoEntrada

Chave	Nome	Tipo	Descrição
1 F	id_dado	INTEGER UNSIGNED NOT NULL	Identificador da tabela Dado, representa o arquivo que serviu de parâmetro de entrada para uma atividade do workflow.
F	id_atividadeCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela AtividadeCase, representa a atividade do workflow que utilizou o arquivo.

Tabela A.6 : DadoSaida

Chave	Nome	Tipo	Descrição
1 F	id_dado	INTEGER UNSIGNED NOT NULL	Identificador da tabela Dado, representa o arquivo que foi produzido por uma atividade do workflow.
F	id_atividadeCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela AtividadeCase, representa a atividade do workflow que gerou o arquivo.

Tabela A.7 : Experimento

Chave	Nome	Tipo	Descrição
PK1	id_experimento	INTEGER UNSIGNED NOT NULL	Identificador da tabela Experimento.
	nome	VARCHAR(100) NOT NULL	Nome simbólico atribuido ao experimento.
	dh_criacao	TIMESTAMP NOT NULL	Data e hora da criação do experimento.
	anotacao	TEXT	Anotações ou observações feitas pelos engenheiros sobre o experimento.

Tabela A.8 : Usuario

Chave	Nome	Tipo	Descrição
PK1	id_usuario	INTEGER UNSIGNED NOT NULL	Identificador da tabela Usuario.
	Nome	VARCHAR(100) NOT NULL	Nome do usuário.
	Login	VARCHAR(10) NOT NULL	Login do usuário.
	Senha	VARCHAR(32) NOT NULL	Senha encriptada do usuário.

Tabela A.9 : Workflow

Chave	Nome	Tipo	Descrição
PK1	id_workFlow	SMALLINT UNSIGNED NOT NULL	Identificador da tabela Workflow.
	Sigla	VARCHAR(20) NOT NULL	Sigla do workflow.
	Nome	VARCHAR(150) NOT NULL	Nome do workflow.

Tabela A.10 : WorkflowCase

Chave	Nome	Tipo	Descrição
PK1	id_workflowCase	INTEGER UNSIGNED NOT NULL	Identificador da tabela WorkflowCase.
FK	id_workflow	SMALLINT UNSIGNED NOT NULL	Identificador da tabela Workflow, representa o workflow executado na análise.
FK	id_experimento	INTEGER UNSIGNED NOT NULL	Identificador da tabela Experimento, representa a análise científica que está sendo realizada.
FK	id_usuario	INTEGER UNSIGNED NOT NULL	Identificador da tabela Usuario, representa o usuário que executou o workflow,
	dh_inicio	TIMESTAMP NOT NULL	Data e hora em que a execução do workflow teve início.

	dh_fim	TIMESTAMP	Data e hora em que a execução do workflow foi concluída.
	nomearquivo	VARCHAR(100) NOT NULL	Nome do arquivo do sistema gerenciador de workflow que foi criado para realizar a análise científica.
	anotacao	TEXT	Anotações ou observações feitas pelos engenheiros sobre a execução do workflow.

Apêndice B – Script de Criação do Banco de Dados

```
-- create database
-------------------------------------------------------------------


CREATE DATABASE wf WITH OWNER = postgres ENCODING = 'WIN1252';


-- Table: experimento
-------------------------------------------------------------------

CREATE TABLE experimento (
  id_experimento serial NOT NULL,
  nome character varying(100) NOT NULL,
  dh_criacao timestamp without time zone NOT NULL,
  anotacao text,
  CONSTRAINT experimento_pkey PRIMARY KEY (id_experimento)
) WITH (OIDS=FALSE);
ALTER TABLE experimento OWNER TO postgres;
COMMENT ON TABLE experimento IS 'Representa as análises científicas
realizadas pelos engenheiros.';
COMMENT ON COLUMN experimento.id_experimento IS 'Identificador da
tabela Experimento.';
COMMENT ON COLUMN experimento.nome IS 'Nome simbólico atribuido ao
experimento.';
COMMENT ON COLUMN experimento.dh_criacao IS 'Data e hora da criação do
experimento.';
COMMENT ON COLUMN experimento.anotacao IS 'Anotações ou observações
feitas pelos engenheiros sobre o experimento.';

CREATE UNIQUE INDEX idx_experimento1 ON experimento USING btree
(id_experimento);


-- Table: usuario
-------------------------------------------------------------------

CREATE TABLE usuario (
  id_usuario serial NOT NULL,
  nome character varying(100) NOT NULL,
  "login" character varying(10) NOT NULL,
  senha character varying(32) NOT NULL,
  CONSTRAINT usuario_pkey PRIMARY KEY (id_usuario)
) WITH (OIDS=FALSE);
ALTER TABLE usuario OWNER TO postgres;
```

```
COMMENT ON TABLE usuario IS 'Contém os engenheiros que executam os
workflows científicos. ';
COMMENT ON COLUMN usuario.id_usuario IS 'Identificador da tabela
Usuario.';
COMMENT ON COLUMN usuario.nome IS 'Nome do usuário.';
COMMENT ON COLUMN usuario."login" IS 'Login do usuário.';
COMMENT ON COLUMN usuario.senha IS 'Senha encriptada do usuário.';

CREATE UNIQUE INDEX idx_usuario1 ON usuario USING btree (id_usuario);


-- Table: workflow
-----------------------------------------------------------------------

CREATE TABLE workflow(
  id_workflow serial NOT NULL,
  sigla character varying(20) NOT NULL,
  nome character varying(150) NOT NULL,
  CONSTRAINT workflow_pkey PRIMARY KEY (id_workflow)
) WITH (OIDS=FALSE);
ALTER TABLE workflow OWNER TO postgres;
COMMENT ON TABLE workflow IS 'Representa os workflows científicos
executados pelos engenheiros.';
COMMENT ON COLUMN workflow.id_workflow IS 'Identificador da tabela
Workflow.';
COMMENT ON COLUMN workflow.sigla IS 'Sigla do workflow.';
COMMENT ON COLUMN workflow.nome IS 'Nome do workflow.';

CREATE UNIQUE INDEX idx_workflow1 ON workflow USING btree
(id_workflow);


-- Table: atividade
-----------------------------------------------------------------------

CREATE TABLE atividade (
  id_atividade serial NOT NULL,
  id_workflow smallint NOT NULL,
  nome character varying(100) NOT NULL,
  ferramenta character varying(50) NOT NULL,
  ordem smallint NOT NULL,
  CONSTRAINT atividade_pkey PRIMARY KEY (id_atividade),
  CONSTRAINT atividade_id_workflow_fkey FOREIGN KEY (id_workflow)
      REFERENCES workflow (id_workflow) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE atividade OWNER TO postgres;
COMMENT ON TABLE atividade IS 'Representa as atividades dos
workflows.';
COMMENT ON COLUMN atividade.id_atividade IS 'Identificador da tabela
Atividade.';
COMMENT ON COLUMN atividade.id_workflow IS 'Identificador da tabela
Workflow, representa o workflow no qual a atividade é executada.';
COMMENT ON COLUMN atividade.nome IS 'Nome da atividade';
COMMENT ON COLUMN atividade.ordem IS 'Ordem em que a atividade é
executada dentro do workflow.';
CREATE INDEX idx_atividade1 ON atividade USING btree (id_workflow);
CREATE UNIQUE INDEX idx_atividade2 ON atividade USING btree
(id_atividade);
```

```
-- Table: workflowcase
-----------------------------------------------------------------------

CREATE TABLE workflowcase (
  id_workflowcase serial NOT NULL,
  id_workflow smallint NOT NULL,
  id_experimento smallint NOT NULL,
  id_usuario smallint NOT NULL,
  dh_inicio timestamp without time zone NOT NULL,
  dh_fim timestamp without time zone,
  nomearquivo character varying(100) NOT NULL,
  anotacao text,
  CONSTRAINT workflowcase_pkey PRIMARY KEY (id_workflowcase),
  CONSTRAINT     workflowcase_id_experimento_fkey     FOREIGN     KEY
(id_experimento)
      REFERENCES experimento (id_experimento) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT workflowcase_id_usuario_fkey FOREIGN KEY (id_usuario)
      REFERENCES usuario (id_usuario) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT workflowcase_id_workflow_fkey FOREIGN KEY (id_workflow)
      REFERENCES workflow (id_workflow) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE workflowcase OWNER TO postgres;

COMMENT ON TABLE workflowcase IS 'Representa as execuções dos
workflows.';
COMMENT ON COLUMN workflowcase.id_workflowcase IS 'Identificador da
tabela WorkflowCase.';
COMMENT ON COLUMN workflowcase.id_workflow IS 'Identificador da tabela
Workflow, representa o workflow executado na análise.';
COMMENT ON COLUMN workflowcase.id_experimento IS 'Identificador da
tabela Experimento, representa a análise científica que está sendo
realizada.';
COMMENT ON COLUMN workflowcase.id_usuario IS 'Identificador da tabela
Usuario, representa o usuário que executou o workflow,';
COMMENT ON COLUMN workflowcase.dh_inicio IS 'Data e hora em que a
execução do workflow teve início.';
COMMENT ON COLUMN workflowcase.dh_fim IS 'Data e hora em que a
execução do workflow foi concluída.';
COMMENT ON COLUMN workflowcase.nomearquivo IS 'Nome do arquivo do
sistema gerenciador de workflow que foi criado para realizar a análise
científica.';
COMMENT ON COLUMN workflowcase.anotacao IS 'Anotações ou observações
feitas pelos engenheiros sobre a execução do workflow.';

CREATE INDEX idx_workflowcase1 ON workflowcase USING btree
(id_experimento);
CREATE UNIQUE INDEX idx_workflowcase2 ON workflowcase USING btree
(id_workflowcase);
CREATE INDEX idx_workflowcase3 ON workflowcase USING btree
(id_workflow);
CREATE INDEX idx_workflowcase4 ON workflowcase USING btree
(id_usuario);
```

```
-- Table: atividadecase
-----------------------------------------------------------------------

CREATE TABLE atividadecase (
  id_atividadecase serial NOT NULL,
  id_atividade smallint NOT NULL,
  id_workflowcase smallint NOT NULL,
  dh_inicio timestamp without time zone NOT NULL,
  dh_fim timestamp without time zone,
  anotacao text,
  CONSTRAINT atividadecase_pkey PRIMARY KEY (id_atividadecase),
  CONSTRAINT      atividadecase_id_atividade_fkey      FOREIGN      KEY
(id_atividade)
      REFERENCES atividade (id_atividade) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT     atividadecase_id_workflowcase_fkey     FOREIGN     KEY
(id_workflowcase)
      REFERENCES workflowcase (id_workflowcase) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE atividadecase OWNER TO postgres;

COMMENT ON TABLE atividadecase IS 'Representa as instâncias   das
atividades das execuções dos workflows.';
COMMENT ON COLUMN atividadecase.id_atividadecase IS 'Identificador da
tabela AtividadeCase.';
COMMENT ON COLUMN atividadecase.id_atividade IS 'Identificador da
tabela Atividade, representa a atividade que executada no workflow';
COMMENT ON COLUMN atividadecase.id_workflowcase IS 'Identificador da
tabela WorkflowCase, representa a execução do workflow na qual a
atividade foi realizada.';
COMMENT ON COLUMN atividadecase.dh_inicio IS 'Data e hora em que a
atividade teve início.';
COMMENT ON COLUMN atividadecase.dh_fim IS 'Data e hora em que a
atividade foi concluída.';
COMMENT ON COLUMN atividadecase.anotacao IS 'Anotações ou observações
feitas pelos pesquisadores sobre a atividade da execução do
workflow.';

CREATE INDEX idx_atividadecase1 ON atividadecase USING btree
(id_atividade);
CREATE INDEX idx_atividadecase2 ON atividadecase USING btree
(id_workflowcase);
CREATE UNIQUE INDEX idx_atividadecase3 ON atividadecase USING btree
(id_atividadecase);


-- Table: dado
-----------------------------------------------------------------------
CREATE TABLE dado (
  id_dado serial NOT NULL,
  id_workflowcase smallint NOT NULL,
  nome character varying(100) NOT NULL,
  dh_criacao timestamp without time zone NOT NULL,
  CONSTRAINT dado_pkey PRIMARY KEY (id_dado),
  CONSTRAINT dado_id_workflowcase_fkey FOREIGN KEY (id_workflowcase)
      REFERENCES workflowcase (id_workflowcase) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE dado OWNER TO postgres;
```

```
COMMENT ON TABLE dado IS 'Contém todos os arquivos envolvidos durante
a execução de um workflow.';
COMMENT ON COLUMN dado.id_dado IS 'Identificador da tabela Dado.';
COMMENT ON COLUMN dado.id_workflowcase IS 'Identificador da tabela
WorkflowCase, representa a instância da execução do workflow que
utilizou o arquivo.';
COMMENT ON COLUMN dado.nome IS 'Nome do arquivo envolvido na execução
do workflow.';
COMMENT ON COLUMN dado.dh_criacao IS 'Data e hora da criação do
arquivo.';

CREATE UNIQUE INDEX idx_dado1 ON dado USING btree (id_dado);
CREATE INDEX idx_dado2 ON dado USING btree (id_workflowcase);


-- Table: dadoentrada
-----------------------------------------------------------------------

CREATE TABLE dadoentrada (
  id_dado smallint NOT NULL,
  id_atividadecase smallint NOT NULL,
  CONSTRAINT dadoentrada_pkey PRIMARY KEY (id_dado),
  CONSTRAINT     dadoentrada_id_atividadecase_fkey     FOREIGN     KEY
(id_atividadecase)
      REFERENCES atividadecase (id_atividadecase) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT dadoentrada_id_dado_fkey FOREIGN KEY (id_dado)
      REFERENCES dado (id_dado) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE dadoentrada OWNER TO postgres;
COMMENT ON TABLE dadoentrada IS 'Contém os arquivos que são parâmetros
de entrada para uma atividade na execução de um workflow.';
COMMENT ON COLUMN dadoentrada.id_dado IS 'Identificador da tabela
Dado, representa o arquivo que serviu de parâmetro de entrada para uma
atividade do workflow.';
COMMENT ON COLUMN dadoentrada.id_atividadecase IS 'Identificador da
tabela AtividadeCase, representa a atividade do workflow que utilizou
o arquivo.';

CREATE INDEX idx_dadoentrada1 ON dadoentrada USING btree(id_dado);
CREATE     INDEX     idx_dadoentrada2     ON     dadoentrada     USING
btree(id_atividadecase);


-- Table: dadosaida
-----------------------------------------------------------------------

CREATE TABLE dadosaida(
  id_dado smallint NOT NULL,
  id_atividadecase smallint NOT NULL,
  CONSTRAINT dadosaida_pkey PRIMARY KEY (id_dado),
  CONSTRAINT     dadosaida_id_atividadecase_fkey     FOREIGN     KEY
(id_atividadecase)
      REFERENCES atividadecase (id_atividadecase) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT dadosaida_id_dado_fkey FOREIGN KEY (id_dado)
      REFERENCES dado (id_dado) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
) WITH (OIDS=FALSE);
ALTER TABLE dadosaida OWNER TO postgres;
```

```
COMMENT ON TABLE dadosaida IS 'Contém os arquivos que foram gerados
por uma atividade na execução de um workflow.';
COMMENT ON COLUMN dadosaida.id_dado IS 'Identificador da tabela Dado,
representa o arquivo que foi produzido por uma atividade do
workflow.';
COMMENT ON COLUMN dadosaida.id_atividadecase IS 'Identificador da
tabela AtividadeCase, representa a atividade do workflow que gerou o
arquivo.';

CREATE INDEX idx_dadosaida1 ON dadosaida USING btree (id_dado);
CREATE INDEX idx_dadosaida2 ON dadosaida USING btree
(id_atividadecase);
```

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo