Igor de Oliveira Knop

Abordagem para Guiar a Reprodução de Experimentos Computacionais: Aplicações em Biologia Computacional

Juiz de Fora

2016

Igor de Oliveira Knop

Abordagem para Guiar a Reprodução de Experimentos Computacionais: Aplicações em Biologia Computacional

Tese apresentada ao Programa de Pós-Graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Doutor em Modelagem Computacional.

Universidade Federal de Juiz de Fora – UFJF

Doutorado em Modelagem Computacional

Programa de Pós-Graduação em Modelagem Computacional

Orientador: Prof. D.Sc. Ciro de Barros Barbosa

Coorientador: Prof. D.Sc. Rodrigo Weber dos Santos

Juiz de Fora

2016

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Knop, Igor de Oliveira.
Abordagem para guiar a reprodução de experimentos computacionais : Aplicações em biologia computacional / Igor de Oliveira Knop. -- 2016.
127 f. : il.

Orientador: Ciro de Barros Barbosa
Coorientador: Rodrigo Weber dos Santos
Tese (doutorado) - Universidade Federal de Juiz de Fora, ICE/Engenharia. Programa de Pós-Graduação em Modelagem Computacional, 2016.

1. Workflows Científicos. 2. Metamodelagem. 3. Modelos Biológicos. I. Barbosa, Ciro de Barros, orient. II. Santos, Rodrigo Weber dos, coorient. III. Título.

Igor de Oliveira Knop

Abordagem para Guiar a Reprodução de Experimentos Computacionais: Aplicações em Biologia Computacional

Tese apresentada ao Programa de Pós-Graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Doutor em Modelagem Computacional.

Aprovada em Juiz de Fora, 31 de Março de 2016:

BANCA EXAMINADORA

Prof. D.Sc. Ciro de Barros Barbosa
PGMC/UFJF - Orientador

Prof. D.Sc. Rodrigo Weber dos Santos
PGMC/UFJF - Co-Orientador

Prof. D.Sc. Alcione de Paiva Oliveira
DCC/UFV

Prof. D.Sc. Márcio de Oliveira Barros
PPGI/UNIRIO

Prof. D.Sc. Heder Soares Bernardino
PGMC/UFJF

Prof. D.Sc. Marcelo Lobosco
PGMC/UFJF

Este trabalho é dedicado à Francisco Knop (in memorian). Viveu seus dias como exemplo de simpatia e simplicidade. Hoje é saudosa e carinhosamente lembrado por todos amigos e familiares.

Agradecimentos

Os meus agradecimentos iniciais são direcionados aos meus orientadores Ciro e Rodrigo. Muito obrigado pela paciência, ideias, conversas e confiança compartilhadas ao longo desses anos. Até mesmo quando o caminho era mais incerto, sempre foram presentes e atenciosos ao oferecer conselhos e firmes na hora de corrigir condutas e direcionar os trabalhos.

Obrigado aos membros da banca de qualificação pelo aceite e pela disponibilidade de tempo nesta época tão atribulada.

Agradeço aos colegas do PGMC por dividir as lutas, alegrias e frustrações desses últimos anos. Acabarei sendo injusto com alguns, mas não posso deixar de destacar os colegas Xandão Pigozzo, Bárbara Quintela e Micheli Farage por todo apoio ao trabalho e à Dani Schmitz, Grasiele Duarte e Karla Perine pelo convívio sempre animado.

Agradeço à toda esquipe técnica do programa. Em especial à Natália e ao Reginaldo pelo apoio nos momentos críticos.

Agradeço à CAPES, FAPEMIG e UFJF pelo suporte para o desenvolvimento da pesquisa.

Resumo

A biologia sistêmica é uma das áreas emergentes mais poderosas no terceiro milênio por combinar de forma interdisciplinar conhecimentos e ferramentas da biologia, ciência da computação, medicina, química e engenharia. Entretanto, o contínuo desenvolvimento de experimentos computacionais é acompanhado por problemas como a integração manual de ferramentas para simulação e análise; a perda de modelos pela obsolescência de *software*; e a dificuldade para reprodução dos experimentos devido à falta de detalhes do ambiente de execução utilizado. A maioria dos modelos quantitativos publicados em biologia são perdidos porque eles, ou não estão mais disponíveis, ou porque são insuficientemente caracterizados para permitir sua reprodução. Este trabalho propõe uma abordagem para guiar o registro de experimentos *in silico* com foco na sua reprodução. A abordagem prevê a criação de uma série de anotações durante um trabalho em modelagem computacional, amparado por um ambiente de *software*, onde o pesquisador realiza as etapas de integração de ferramentas, descrição de processos e execução de experimentos. O objetivo é capturar o processo de modelagem de forma não invasiva para aumentar a troca de conhecimento, permitir repetição e validação dos resultados e diminuir o retrabalho em grupos de pesquisa interdisciplinares. Um ambiente computacional protótipo foi construído e dois fluxos de trabalho de ferramentas diferentes foram integradas como estudos de caso. O primeiro usa modelos da eletrofisiologia cardíaca para se construir novas aplicações sobre o ambiente. O segundo apresenta um novo uso para os metamodelos de dinâmica de sistemas para simular a resposta do sistema imune inato em uma seção planar de tecido. Foi observada a completa captura dos *workflow* de simulação e tratamento dos dados de saída nos dois experimentos de controle. O ambiente permitiu a reprodução e adaptação dos experimentos em três níveis diferentes: a criação de novos experimentos utilizando a mesma estrutura do original; a definição de novos aplicativos que utilizam variações da estrutura do experimento original; reaproveitamento do fluxo de trabalho para alterações nos modelos e condições originais.

Palavras-chaves: *Workflows* Científicos. Reprodutibilidade. Metamodelagem. Modelos Biológicos.

Abstract

Systems Biology is one of the most powerful emerging areas in the third millennium that combines, in an interdisciplinary way, knowledge and tools of Biology, Computer Science, Medicine, Chemistry and Engineering. However, the continued development of computational experiments is accompanied by problems such as manual integration of tools to simulation and analysis; loss of models due software obsolescence; and the difficulty to reproduce the experiments due to lack of details of the execution environment used. Most quantitative models published in Biology are lost because they, or are no longer available or are insufficiently characterized for reproduction. This work proposes an approach to guide the registration of *in silico* experiments focused on its reproduction. The approach involves the creation of a series of annotations during computational modeling, supported by a software environment where the researcher conducts the tool integration steps, process description and execution of experiments. The goal is to noninvasively capture the modeling process to increase the exchange of knowledge, allow repetition and validation of the results and reduce rework in interdisciplinary research groups. A prototype was built and two different workflows have been integrated as case studies. The first uses models and tools of cardiac electrophysiology to build new applications on the environment. The second presents a new use for the system dynamics metamodeling to simulate the response of the innate immune system in a planar section of tissue. The complete capture of workflow, consisting of simulation and processing of output data, in two control experiments, was observed. The environment allowed the reproduction and adaptation of experiments at three different levels: the creation of new experiments using the same structure as the original one; the definition of new applications that use variations of the structure of the original experiment; the reuse of workflow to change models and original condition.

Key-words: Scientific Workflows. Reproducibility. Metamodeling. Biological Models.

Lista de ilustrações

Lista de tabelas

Glossário

AGOS *API Generator for ODE Solution*. 9, 15, 20, 21, 24, 81–85, 89, 115

AGOSWeb *AGOS Web*. 9, 86, 88–90, 115

AWK Linguagem de *script* AWK. 108

CELL-ML *Cell Markup Language*. 17, 21, 22, 81–83, 85–87, 89, 113, 115

EDO Equação Diferencial Ordinária. 34, 35

EDP Equação Diferencial Parcial. 35, 91

FISIOENV *Fisiocomp Environment*. 20, 23, 71–75, 83, 84, 86, 88–90, 106, 108, 115, 117

GCC *GNU C Compiler*. 71, 82, 89

GNU/Linux *GNU/Linux*. 71

GNUPLOT *GNUPLOT*. 94, 108

HSQLDB *HyperSQL DataBase*. 71

HTTP *HyperText Transfer Protocol*. 71

JSON *JavaScript Object Notation*. 54

JynaCore API *Java Dynamics Core API*. 10, 21, 24, 81, 89, 94, 104, 106, 107, 110, 115

JynaSim *JynaCore Simulator*. 94

LPS Lipopolissacarídeo. 32, 90–93, 103

MathML *Mathematics Markup Language*. 115

MIASE *Minimal Information About a Simulation Experiment*. 22, 44, 45, 112

NeuroML *Neuronal Markup Language*. 17

REST *Representational State Transfer*. 72, 83

SBML *Systems Biology Markup Language*. 17, 22, 112, 113

SBSI *Systems Biology Software Infrastructure*. 22, 112

SED-ML *Simulation Experiment Description Markup Language*. 22, 112

SEE *Simulation Experiment Environment*. 8, 70, 71, 75

SEEM *Simulation Experiment Environment Method*. 8, 67, 68, 70

SGBD Sistema Gerenciador de Banco de Dados. 68

SGWf Sistema Gerenciador de *Workflow*. 59, 71

SIH *Sistema Imune Humano*. 30–32, 40, 98, 110

SOAP *Simple Object Access Protocol*. 72

SWfMS *Scientific Workflow Management System*. 43

UML *Unified Modeling Language*. 42, 52

URI *Uniform Resource Identifier*. 66, 75, 76

UUID *Universally Unique Identifier*. 66, 75, 76, 83

XML *Extensible Markup Language*. 21, 22, 54, 112

Sumário

1 Introdução

A biologia sistêmica é uma das disciplinas emergentes mais poderosas no terceiro milênio, utilizando de forma interdisciplinar os princípios, conhecimentos e ferramentas advindas da biologia, ciência da computação, medicina, química e engenharia para preencher as lacunas existentes entre todas essas áreas na busca pelo entendimento do comportamento dos sistemas biológicos (MEDINA, 2013).

Cada vez mais os métodos tradicionais para conduzir ciência a partir de observações de processos naturais *in vivo* ou por ambientes controlados *in vitro* estão sendo complementados ou substituídos por experimentos *in silico* (LUDÄSCHER et al., 2006). Assim hipóteses são propostas, confirmadas ou refutadas através de simulações sobre modelos computacionais (LASZEWSKI; HATEGAN; KODEBOYINA, 2007). Com os avanços em tecnologia da informação, novas ferramentas estão mudando a forma como se faz ciência, automatizando tarefas repetitivas e com grande consumo de tempo e liberando a atenção dos cientistas para outras funções (FOSTER, 2005).

O desenvolvimento de experimentos *in silico* é uma atividade que apresenta complicações além do domínio biológico. O desenvolvimento de ferramentas computacionais para estudo das dinâmicas de diversos campos da biologia é acompanhado por uma série de novos problemas que podem atrasar o processo de produção científica. Podemos citar: a integração manual de diversas ferramentas de simulação e de análise; a perda de modelos pela obsolescência de software e a dificuldade para reprodução dos experimentos devido à falta de detalhes da configuração do ambiente (HUCKA et al., 2003). Assim, a maioria dos modelos quantitativos publicados em biologia são perdidos ou porque eles não estão mais disponíveis ou porque são insuficientemente caracterizados para permitir uma reutilização (NOVERE et al., 2005).

Tornou-se necessário definir padrões abertos para intercâmbio de modelos de forma que possam ser lidos e interpretados por diversas ferramentas de simulação. Padrões como o Cell Markup Language (CELL-ML) para células (LLOYD; HALSTEAD; NIELSEN, 2004), o Systems Biology Markup Language (SBML) para sistemas biológicos dinâmicos (HUCKA et al., 2003) e o Neuronal Markup Language (NeuroML) para redes de neurônios (GLEESON et al., 2010) já são bem estabelecidos e em estado maduro de desenvolvimento.

Os modelos desenvolvidos precisam ser disponibilizados publicamente para que grupos de pesquisa diferentes possam realizar a troca de modelos entre si. Há um grande esforço na criação de repositórios públicos de modelos e vários já são reconhecidos e aceitos pela comunidade científica (NOVERE et al., 2006; KESELER et al., 2005; CAMPAGNE et al., 2004). Esses repositórios permitem que modelos sejam: organizados dentro de seus

domínios de aplicação; recebam um processo de curadoria; sejam pesquisados e recuperados para posterior adaptação para a construção de novos modelos.

Além do compartilhamento de modelos, há a preocupação com a descrição e compartilhamento de todo o processo de execução dos experimentos *in silico*. Na grande parte dos experimentos, fornecer o modelo sozinho não é suficiente para se obter os mesmos resultados. Um experimento *in silico* pode exigir condições iniciais diferentes, uso de ferramentas auxiliares para tratamento de dados intermediários ou de saída, fontes de dados de entrada podem ser alimentadas manualmente ou em uma ordem muito específica pelo próprio pesquisador (WALTEMATH et al., 2011a; WALTEMATH et al., 2011b). Ao encadeamento de programas de forma que os dados produzidos por um sejam consumidos por outro é dado o nome de *workflow*. Um experimento *in silico* também pode ser visto como um *workflow* científico (LAWRENCE, 1997; MATTOSO et al., 2010).

1.1 Questão

O cenário levantado acima leva a formular, em um âmbito mais geral, a questão: **"Como guiar a reprodução de experimentos *in silico*?"**. Ou seja, como e onde atuar de forma que a descrição dos experimentos *in silico* permita auxiliar pesquisadores a realizar uma posterior repetição, reprodução ou adaptação. Estando o processo de modelagem computacional de sistemas biológicos fortemente vinculado a um contexto de produção de *software*, espera-se que esta descrição e posterior reuso sejam amparados por um ferramental que automatize total ou em parte os diversos passos do ciclo de vida de um experimento.

1.2 Hipótese

Para responder essa questão, este trabalho propõe um novo método de construção de experimentos científicos, chamado de *Simulation Experiment Environment Method* (SEEM), que irá identificar as informações relevantes de todo o ciclo de vida do experimento *in silico*. O experimento será descrito desde sua conceitualização até a implementação computacional, e serão identificados os artefatos e processos utilizados para a obtenção dos resultados. Para tanto, o método prevê o uso de um ambiente computacional no qual o pesquisador irá publicar seu experimento. O método identificará as possíveis automações no processo de extração de informações e guiará a decisão do pesquisador para os pontos nos quais a documentação não puder ser automatizada. O método deve estimular o pesquisador a fornecer o máximo de informações durante o processo de criação e manutenção dos experimentos na forma de anotações. A partir dessas anotações acrescentadas ao experimento, o método irá guiar uma posterior reprodução. Também através de anotações, os pontos que impedem a reprodução do experimento serão destacados, e o

método irá guiar o pesquisador a concentrar seu trabalho para prosseguir com o processo científico.

1.3 Justificativa

Apesar das diversas iniciativas resolverem partes isoladas do problema, ainda não há um método sistemático que permita capturar ou anotar em um nível mais abstrato as informações necessárias para a livre troca de dados, modelos e processos entre ambientes diferentes. Os esforços atuais se dividem entre soluções para o processo de criação (com foco em linguagens de modelagem); métodos numéricos de simulação ou na composição e execução do experimento (sistemas de gerenciamento de *workflow*). Não há um método que forneça ao usuário vários níveis de abstração, indo do mais concreto a nível de orquestração de aplicativos ao mais abstrato, a nível de modelos e dados de saída. Portanto, faz-se necessário criar um novo método que descreva o experimento como um todo com as informações mínimas necessárias para sua reprodução, mas sem sobrecarregar o processo de anotação.

1.4 Abordagem

Na apresentação do método SEEM, exploramos sua relação com as diversas fases do ciclo de vida de um experimento científico *in silico*. Descrevemos os principais conceitos e as suas inter-relações de uma forma abstrata. As informações fundamentais para a reprodução são capturadas na forma de anotações e fornecemos uma sugestão de arquitetura para que os passos do método possam ser implementados em algum sistema computacional.

Para exemplificar a aplicação ao método proposto, foi desenvolvido um ambiente de *software* chamado *Simulation Experiment Environment* (SEE), para experimentação *in silico* de acordo com as recomendações propostas. O SEE apresenta uma interface *web* para o usuário, na qual é possível criar, documentar, simular e compartilhar modelos de sistemas biológicos. Os pesquisadores podem integrar novas ferramentas ao SEE e construir novos modelos. Esses modelos são utilizados na definição de experimentos *in silico* e o sistema automatiza ao máximo a sua documentação. As anotações são exportadas, utilizando protocolos bem definidos pela comunidade, para reprodução posterior. Inicialmente, esse novo ambiente dá suporte a duas linguagens de modelagem, o padrão CellML (GARNY et al., 2008) e uma linguagem de metamodelos de Dinâmica de Sistemas (BARROS, 2001). Dessa forma, pretendemos demonstrar a abrangência do método para diferentes experimentos, com processos de simulação bem diferentes, que podem ser descritos, armazenados, simulados e exportados por meio de padrões abertos.

Buscamos com o SEEM permitir que grupos de pesquisa na área de modelagem computacional tenham seu trabalho facilitado por um método que foca na reprodução de experimentos. O método permite que um pesquisador publique seu resultado em um ambiente de simulação e este pode ser reproduzido por outros pesquisadores com automação completa ou parcial.

Ao permitir que dados sejam colhidos automaticamente, disponibilizamos a base para que possam ser construídas aplicações que gerenciam métricas para avaliação das ferramentas e técnicas utilizadas. Embora o método seja aplicado na construção do experimento, auxiliando nesse processo, técnicas visando a melhoria do processo de modelagem, como reuso de modelos, conversão de modelos e orquestração, ou visando a eficiência da execução do experimento, como por exemplo, técnicas de paralelização e uso de computação em larga escala, estão fora do escopo deste trabalho.

Este trabalho também apresenta conceitos de dois campos mais específicos da Biologia Sistêmica nos quais os estudos de caso foram realizados. A modelagem da Eletrofisiologia Cardíaca se propõe a resolver problemas associados às doenças cardiovasculares, que são responsáveis pelo maior número de mortes no mundo (WORLD HEALTH ORGANIZATION, 2014). O interesse está nos processos que geram e propagam o potencial de ação das células e como as doenças cardiovasculares afetam o seu comportamento padrão. Mais especificamente, são utilizados modelos relacionados aos efeitos da Doença de Chagas nos miócitos cardíacos de camundongos durante a fase aguda da doença. Como o potencial de ação é alterado durante a invasão pelo protozoário da Doença de Chagas, outro campo que foi utilizado como aplicação envolve a investigação da dinâmica básica de modelos do Sistema Imune Inato. O sistema imune apresenta um conjunto de células e tecidos que interagem para impedir ou minimizar os efeitos causados por vírus, bactérias e microrganismos no organismo humano.

1.5 Histórico do Trabalho

Este trabalho teve início com a idealização do que iria se tornar o Fisiocomp Environment (FISIOENV), criado como parte de um projeto no Laboratório de Fisiologia Computacional da UFJF. As atividades durante o projeto consistiam em desenvolver um experimento para eletrofisiologia cardíaca via interface *web* baseada no AGOS. O AGOS é um gerador de código C++ para resolução de problemas modelados por equações diferenciais ordinárias com condições iniciais. Durante o desenvolvimento dos trabalhos, um ponto crítico observado foi a constante necessidade de se contactar outros pesquisadores em busca do material para reproduzir os experimentos antes realizados *ad hoc* por cada um deles. Ou seja, experimentos criados caso a caso, sem interesse em generalização ou posterior adaptação. Era mantido o contato com o autor da ferramenta e um pesquisador

dos modelos situados em dois países diferentes. Toda a troca de informações e arquivos era feita por *e-mails* ou com a ajuda de colegas envolvidos anteriormente em projetos relacionados.

Toda essa necessidade de se obter mais detalhes não expostos nas publicações foi observado tanto na configuração do ambiente para a execução da ferramenta quanto no entendimento e simulação do modelo de eletrofisiologia cardíaca. O *software* original é um programa em linha de comando que utiliza um grande conjunto de bibliotecas e vários arquivos de configuração além do modelo. O modelo é descrito em CELL-ML (GARNY et al., 2008), uma linguagem de intercâmbio de modelos de célula baseado em Extensible Markup Language (XML) (BRAY et al., 2006), mas as relações de causalidade deveriam ser obtidas pela análise da descrição matemática na publicação ou inspeção do código gerado. Com isso, foi alterado o escopo do projeto ao invés de desenvolver puramente uma interface para o aplicativo existente. Optou-se por criar uma nova infraestrutura para integração de ferramentas de modelagem e simulação e, como caso de uso, o AGOS foi integrado como uma nova aplicação *web* para exploração e modelos dinâmicos (KNOP et al., 2012a).

Em outro trabalho prévio, foi desenvolvido o JynaCore API (KNOP, 2011), que é ao mesmo tempo uma interface de aplicativos e um ambiente de modelagem e simulação de sistemas dinâmicos baseado em Dinâmica de Sistemas e em seus metamodelos. Essa ferramenta foi idealizada e utilizada no domínio de aplicação de modelagem de processos de *software*, no qual os metamodelos eram utilizados para modelar indivíduos com estados próprios ao invés de valores médios globais. Essa abordagem poderia ser utilizada no domínio de sistemas biológicos em que há a uma grande interligação de sub-sistemas e modelos com conformações espaciais. Aplicou-se a técnica de modelagem e suas ferramentas para a construção dos modelos do sistema imune humano inato (KNOP et al., 2012b), nos quais, a partir de modelos com dinâmicas mais simples, foi possível progressivamente acrescentar elementos e explorar simulações espaço-temporais (KNOP et al., 2012c).

Os dois trabalhos prévios, bem como seus modelos, apresentaram características bem particulares, com um fluxo de trabalho complexo e de difícil manutenção e reprodução. Optou-se por realizar a junção do estudos tanto na linha de *workflows* científicos quanto na metamodelagem de Dinâmica de Sistemas para o desenvolvimento de um método que amparado por *software* permita capturar quais são os pontos que precisam de atenção por parte do pesquisador interessado em reproduzir o experimento.

1.6 Abordagens para Reprodução de Experimentos

A reprodução de experimentos é um requisito básico para a ciência. Mas com o crescimento do número e da complexidade das ferramentas de *software* na biologia

computacional, uma série de problemas não diretamente relacionados com a pesquisa podem emergir. A reprodução dos experimentos foi tema de uma série de trabalhos ao longo dos anos e estão relacionados conceitualmente (ao método SEEM) ou tecnicamente (à implementação SEE).

A diretriz de relatório para experimentos *in silico* denominada Minimal Information About a Simulation Experiment (MIASE) (WALTEMATH et al., 2011a) estabelece um conjunto mínimo de informações que deve ser fornecido para tornar a descrição de um experimento completamente disponível para outros pesquisadores. Um experimento compatível com a MIASE descreve os modelos, os procedimentos para simulação e em qual ordem, o processamento dos dados numéricos resultantes e a descrição da saída final.

A troca dos experimentos pode envolver a definição de formatos de intercâmbio de *workflows*. Ainda dentro dos guias do MIASE o Simulation Experiment Description Markup Language (SED-ML) codifica a descrição de um experimento em XML mas reutiliza outros padrões já estabelecidos para a descrição de modelos (WALTEMATH et al., 2011b). Um grande conjunto de trabalhos se moveu para a adoção do SED-ML como linguagem de intercâmbio, como o Systems Biology Software Infrastructure (SBSI) (ADAMS et al., 2013) que é um conjunto de aplicações e bibliotecas na área de biologia sistêmica e sintética que objetiva prover acesso a algoritmos paralelizados para tarefas custosas, tais como estimação de parâmetros de modelos descritos por SBML (HUCKA et al., 2003). A atual implementação lê fluxos de trabalhos de outras ferramentas utilizando a proposta do `jlibsedml`, uma biblioteca em Java para leitura, edição e validação de modelos SED-ML (BERGMANN, 2014b).

Dentre as abordagens baseadas em repositório de descrição de experimentos com interface via *web*, pode-se destacar o projeto Repose (GURAVAGE; MERKS, 2011). Baseando-se no SED-ML, o projeto se propõe a permitir que pesquisadores possam criar, editar e compartilhar experimentos. Foi idealizado para servir como um "caderno de bancada" virtual e documentar um experimento da idealização até a publicação, associando aos experimentos uma máquina de estados. Entretanto, ele não se propõe a realizar a execução e reprodução de experimentos.

Na mesma linha de repositórios, o *Workflow-based spatial Decision Support System* (WOODSS) (MEDEIROS et al., 2005) é uma infraestrutura criada inicialmente para planejamento ambiental que foi estendida para descrever e armazenar *workflows* científicos. Através de anotações semânticas, ele descreve os fluxos de trabalho e os serializa em WS-BPEL (OASIS, 2007) para ser utilizado em alguma ferramenta compatível de orquestração de serviços *web*, pois ele não realiza a sua execução.

O *SED-ML Web Tools* (BERGMANN, 2011), desenvolvido em conjunto com a *libSedML* (BERGMANN, 2014a), é uma biblioteca .NET para leitura de arquivos SED-ML que permite aos pesquisadores criar e simular experimentos através de uma interface

web. Atualmente, é possível criar um experimento a partir do zero ou a partir de um modelo SBML (HUCKA et al., 2003) ou CELL-ML (NICKERSON; YOU, 2011).

O projeto Reprozip (CHIRIGATI; SHASHA; FREIRE, 2013) realiza a repetição de experimentos por capturar a estrutura de *workflows* descritos no *software* VisTrails (CALLAHAN et al., 2006) em um arquivo compactado que contêm todos os executáveis utilizados. Isso é feito pelo monitoramento das chamadas ao sistema operacional durante a execução do *workflow*. Uma árvore de dependências a programas é montada e permite que o experimento seja repetido em um outro computador no qual que o VisTrails estiver instalado. É necessário que o sistema operacional de destino e todos os programas sejam iguais aos do experimento original.

Mais recentemente, foi proposto o ResearchCompedia.org (STODDEN; MIGUEZ; SEILER, 2015), uma infraestrutura que permite que os autores publiquem seus artigos, dados e *software* necessários para reproduzir os resultados. A equipe conduz uma execução do código para certificar o experimento junto ao artigo e fornece o código para que qualquer um o utilize em suas próprias máquinas. O projeto se propõe a oferecer os chamados "artigos executáveis" através de execução na nuvem por virtualização se eles atenderem a requisitos de tempo e processamento. Entretanto, não realiza a captura e exposição formal do *workflow* do trabalho.

1.7 Objetivos

Este trabalho consiste na criação de uma nova abordagem para construção de experimentos científicos reprodutíveis, chamado de Simulation Experiment Environment Method (SEEM). É definido um modelo de dados mínimo para realizar o mapeamento de todo o *workflow* de um experimento para capturar os dados necessários para sua posterior reprodução. Uma arquitetura conceitual é proposta para a implementação de um ambiente de publicação de experimentos que consiga realizar cada um dos passos necessários para gerar os resultados obtidos previamente. Adicionalmente, metadados referentes ao estado da caracterização do experimento são capturados e um conjunto de anotações é gerado para descrever cada uma das condições que impeçam sua reprodução.

Para evidenciar a aplicabilidade do método, um protótipo de ambiente computacional, chamado de Simulation Experiment Environment (SEE), é desenvolvido para investigar a criação, execução e crítica da reprodutibilidade de experimentos computacionais. Ou seja, o ambiente permite acompanhar o desenvolvimento do experimento desde a sua concepção até a sua implementação e publicação. Este trabalho se restringiu a experimentos de modelagem computacional de sistemas biológicos, mas isso não é uma limitação da abordagem proposta e tipos de experimentos computacionais podem utilizar o método, como, por exemplo, os baseados em mineração de dados, processamento de

imagens e modelos estocásticos.

Como base para a construção do SEE, foi utilizado o FISIOENV (KNOP et al., 2012a), que é uma infraestrutura capaz de descrever *workflows* de simulações e implantação dos programas responsáveis pelo processamento dos dados. Essa infraestrutura foi idealizada para prover um mínimo de funcionalidades na qual seria possível se criar outras aplicações mais complexas utilizando os construtores básicos, como processos e artefatos. Todas as funcionalidades estão disponíveis através de uma interface *web* e expõem uma interface de aplicativos básica na forma de serviços *web*.

Por fim, como estudo de caso, é realizada a implantação de duas ferramentas no SEE: o AGOS e a JynaCore API. A primeira apresenta um modelo de *workflow* mais complexo, dependente de um grande número de arquivos em disco e diversas ferramentas diferentes. A segunda, a JynaCore API, trabalha com a interpretação de modelos descridos em Dinâmica de Sistemas ou metamodelos de Dinâmica de Sistemas e é desenvolvida sobre a plataforma Java. Ambos os experimentos utilizam algumas ferramentas em comum para realizar o pós-processamento dos dados e apresentar os resultados graficamente.

Com as duas ferramentas integradas ao SEE, experimentos na área de biologia computacional são descritos, simulados, e compartilhados entre instâncias do ambiente, destacando possíveis pontos que inviabilizam sua reprodução. Os dois estudos de caso evidenciam a independência do domínio de aplicação, pois o primeiro experimento utiliza modelos da área de eletrofisiologia cardíaca, enquanto o segundo apresenta modelos espaço-temporais do sistema imune inato.

1.8 Organização do Trabalho

Este documento está organizado em sete capítulos. Este capítulo apresentou uma introdução aos temas relacionados na elaboração do presente trabalho, explicando a nossa questão, motivação, objetivos, hipótese e justificativa.

O Capítulo 2 faz uma revisão dos principais conceitos utilizados neste trabalho e é dividido em três grandes seções: revisa os conceitos relacionados a *workflows* científicos, seus padrões de intercâmbio de modelos e experimentos para acelerar a produção científica; técnicas para modelagem computacional de sistemas biológicos por simulação, em especial, uma revisão de Dinâmica de Sistemas e a metamodelagem de Dinâmica de Sistemas, utilizados em nossos estudos de caso; uma breve revisão dos domínios biológicos explorados neste trabalho, como a eletrofisiologia cardíaca e os componentes do sistema imune inato.

O Capítulo 3 descreve a nossa contribuição conceitual com o método Simulation Experiment Environment Method (SEEM), seu modelo de dados e os passos a serem seguidos para a captura da estrutura dos experimentos. Adicionalmente, propõe uma arqui-

tetura para uma futura implantação de um ambiente computacional chamado Simulation Experiment Environment (SEE).

O Capítulo 4 apresenta a implementação e uso do Simulation Experiment Environment Method, uma realização da arquitetura proposta pelo SEEM. Como os modelos de dados, anotações e críticas são utilizadas para o acompanhamento da reprodutibilidade de experimentos.

O Capítulo 5 apresenta dois estudos de caso em que dois experimentos *in silico* diferentes são capturados e anotados para posterior reprodução. O primeiro estudo utiliza o SEEM no domínio da eletrofisiologia cardíaca no qual um experimento para análise de comportamento de modelo é composto utilizando um conjunto de ferramentas de linha de comando. O segundo descreve a construção de um modelo para o sistema imune inato e o uso de metamodelagem para explorar a distribuição espacial.

O Capítulo 6 que traz uma discussão dos resultados alcançados e como o SEEM se relaciona com outros estudos na área.

Por fim, o Capítulo 7 apresenta as considerações finais desta tese, um resumo dos resultados alcançados e indica caminhos para trabalhos futuros.

2 Revisão Conceitual

Este capítulo apresenta uma revisão dos principais conceitos utilizados neste trabalho e está organizado em cinco grandes seções. A Seção 2.1 formaliza os conceitos biológicos para dois campos de estudo diferentes: o da Eletrofisiologia Cardíaca e do Sistema Imune Inato. Esses conceitos serão utilizados, posteriormente, nos dois estudos de caso presentes no desenvolvimento deste trabalho. Já a Seção 2.2 apresenta os conceitos de Biologia Sistêmica e interação entre as áreas de computação e matemática. A Seção 2.3 apresenta métodos de modelagem e simulação e uma revisão sobre sobre métodos de modelagem computacional utilizados neste trabalho, como a Metamodelagem de Dinâmica de Sistemas. A Seção 2.4 formaliza os conceitos de *workflows* científicos no contexto da biologia computacional e as técnicas para execução, registro e armazenagem de experimentos *in silico*. Por fim, os conceitos de repetição e reprodução de experimentos são vistos na Seção 2.5.

2.1 Conceitos Biológicos

Este trabalho apresenta aplicações computacionais em dois domínios da Biologia: a Eletrofisiologia Cardíaca e o Sistema Imune Inato. Esta Seção realiza uma breve revisão dos conceitos dessas duas áreas.

2.1.1 Eletrofisiologia Cardíaca

O coração é um órgão responsável por impulsionar o sangue rico em oxigênio para todas as células do organismo via sistema arterial e recolher o sangue pobre em oxigênio via sistema venoso. O fluxo sanguíneo é gerado pelo funcionamento de duas bombas separadas pelo septo vertical. Cada bomba é dividida em duas câmaras: a superior, chamada de átrio, e a inferior, chamada de ventrículo.

Para realizar a função de bombeamento do sangue, o coração realiza a contração (sístole) e o relaxamento (diástole). Esse funcionamento é gerido por um sistema de sincronismo gerado por um conjunto de células chamado de sino-atrial ou nódulo de Keith e Flack (NOBLE, 1984). Um mal funcionamento no estímulo ou na sua propagação pelas células do coração pode ocasionar uma série de sintomas como palpitações, fadiga, desmaios ou mesmo a morte do indivíduo por parada cardíaca (HAGIWARA; IRISAWA; KAMEYAMA, 1988).

No nível celular, o tecido do coração é composto pelas células musculares cardíacas conhecidas como miócitos cardíacos ou cardiomiócitos (FAWCETT; MCNUTT, 1969).

Essas células se organizam em sarcômeros, que são agrupamentos de células separadas por membranas celulares, mas com junções comunicantes conhecidas como *gap junctions*. Isso dá a característica de alta permeabilidade de íons e, consequentemente, rápida propagação do potencial elétrico na direção da fibra. A Figura 1 apresenta a estrutura de um sarcômero que forma a fibra cardíaca e é responsável pela contração do coração.

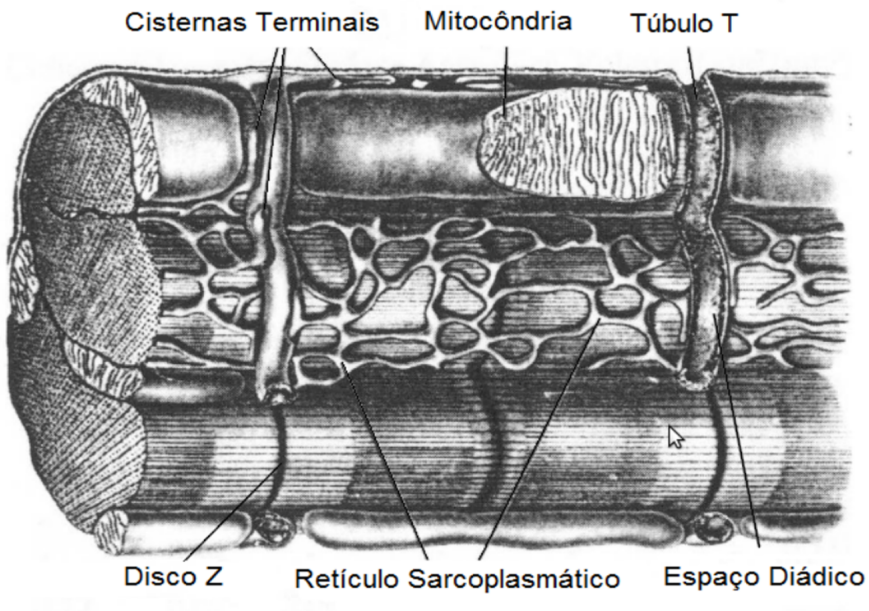


Figura 1 – Células cardíacas organizadas em um sarcômero. Adaptado de Fawcett e McNutt (1969).

A membrana celular separa o interior da célula do meio exterior e controla o fluxo de substâncias entre os dois meios (ALBERTS et al., 2013). Ela é constituída por uma série de fosfolipídios que possuem uma de suas extremidades hidrofílicas e outra hidrofóbica. Essas moléculas se rearranjam para formar duas camadas, uma com sua extremidade hidrofílica virada para o meio extracelular e outra para o meio intracelular. Essa bicamada impede a passagem dos íons, mas permite que proteínas maiores se anexem a ela. Essas proteínas cumprem a maior parte das funções da membrana, como ancoragem às macromoléculas que atuam como receptores de enzimas e realizam o transporte de íons, nutrientes e metabólitos por entre a membrana. A Figura 2 apresenta um esquema da bicamada lipídica e um canal iônico, responsável por aumentar ou diminuir a permeabilidade a íons através da membrana.

A passagem de íons pela membrana ocorre de forma passiva por difusão e sua condutividade é regulada pelas características da célula, em especial ao potencial transmembrânico, ou seja, a diferença de potencial entre o interior e exterior da célula. Uma pequena variação no potencial da membrana vai alterar o funcionamento dos canais iônicos, que por sua vez vão alterar ainda mais o potencial da membrana. A súbita elevação

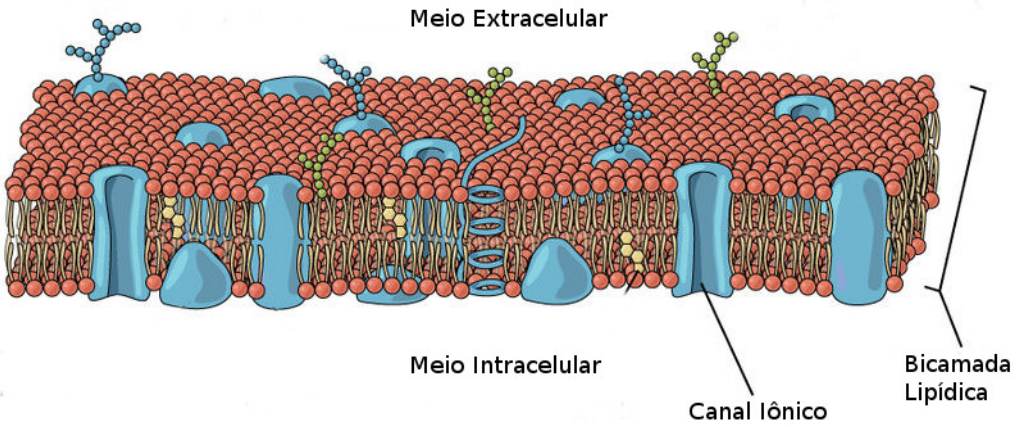


Figura 2 – Bicamada lipídica com proteínas maiores anexadas. Adaptado de OPENSTAX CNX (2013).

do potencial da membrana pelos em função de uma estímulo externo é conhecido como potencial de ação. A Figura 3 apresenta um gráfico do potencial de ação sobre a membrana de um axônio gigante de lula em função de um estímulo aplicado por volta de 10ms.

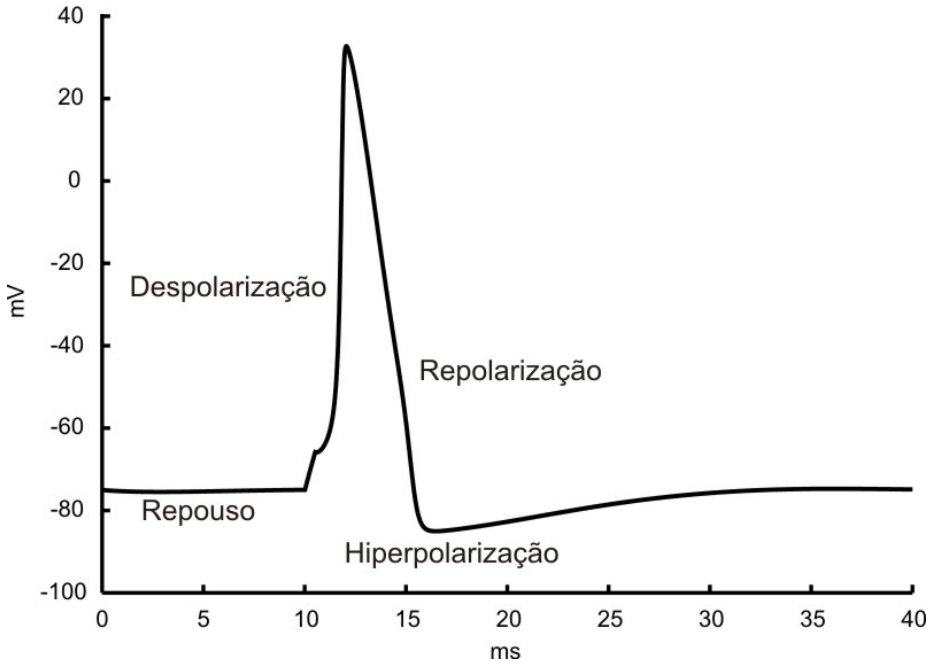


Figura 3 – Potencial de ação no axônio gigante da lula. O eixo vertical apresenta o potencial transmembrânico e o horizontal o tempo. Reproduzido de Campos (2008).

O potencial de ação é gerido pelas correntes de íons pela membrana e sua forma varia de espécie e função celular. Entretanto, sua forma pode ser dividida em cinco fases diferentes. Na fase de repouso o potencial da membrana é de aproximadamente $-80mV$ devido às diferenças de concentrações de íons intra e extracelular. Por exemplo, para o axônio da lula, o íon de sódio Na^+ apresenta concentração de $50000\mu M$ no meio intra-

celular e $497000\mu M$ no meio extracelular; já o íon de potássio K^+, $397000\mu M$ no meio intracelular e $20000\mu M$ no meio extracelular. No início da chamada fase de despolarização, os canais iônicos de Na^+ se abrem, aumentando a permeabilidade da membrana aos íons e gerando uma corrente de entrada para o interior da célula. Isso faz com que o potencial da membrana se torne mais positivo em relação ao de repouso. A fase seguinte, chamada de repolarização inicial é constituída por uma abertura dos canais de K^+ por um curto espaço de tempo. Neste momento os íons saem da célula e geram uma súbita repolariação. A fase de platô é quando os íons de cálcio entram na célula, realizando uma despolarização e estimulando a liberação de mais íons Ca^{2+} armazenados no retículo sarcoplasmático.

O aumento da concentração de cálcio no interior da célula causa a contração do miócito. A fase de repolarização é o retorno da célula para o repouso homeostático. Ela se dá através da abertura dos canais de potássio e efeitos como o funcionamento da bomba de sódio-potássio Na^+K^+ e recaptura de íons de Ca^{2+} pelo retículo sarcoplasmático. A Figura 4 apresenta as fases do potencial de ação e dá destaque para as correntes iônicas em cada uma delas.

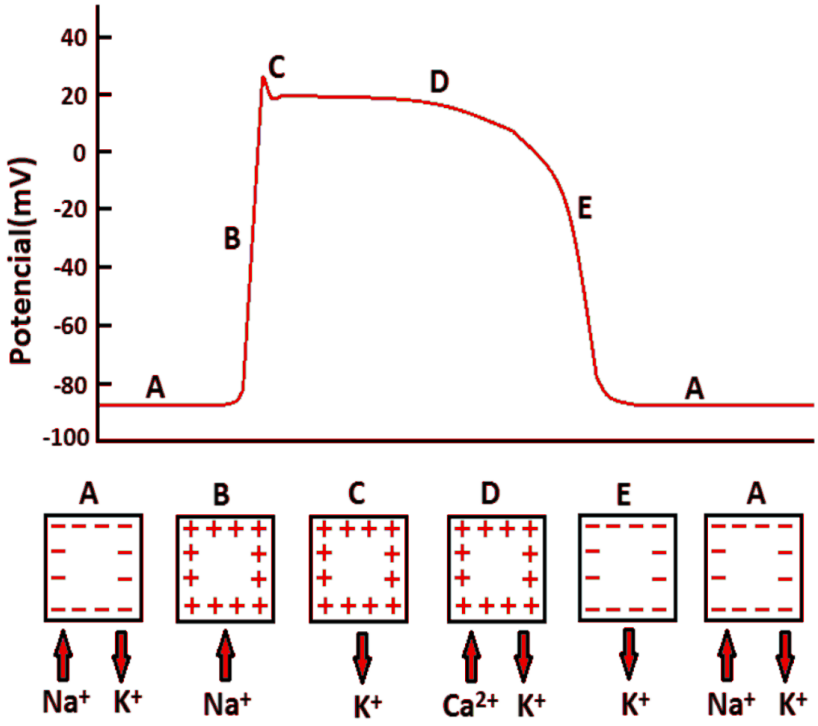


Figura 4 – Fases do Potencial de ação e respectivas correntes de íons do ventrículo cardíaco: (A) Repouso; (B) Despolarização, (C) Repolarização Inicial; (D) Platô; (E) Repolarização. Reproduzido de Barros (2013).

Em 1952, Hodgkin e Huxley (1952) realizaram a primeira modelagem matemática do potencial de ação de um axônio gigante de uma lula que serviu de base para um

grande número de estudos de sistemas excitáveis. Apesar dos modelos realçarem apenas as características principais de um sistema real (NIEDERER; SMITH, 2012), os modelos de células cardíacas tiveram ampla aceitação a partir do trabalho pioneiro de DiFrancesco e Noble (1985) que estudou as fibras de Purkinge. Posteriormente, Bondarenko et al. (2004) realizaram um estudo para a construção de modelos para os miócitos de camundongos. Os camundongos são largamente utilizados como cobaias para a produção de novas drogas. Ter um modelo matemático mais preciso para seus miócitos cardíacos é uma grande ferramenta para estudos da propagação dos potenciais de ação e anomalias cardíacas que acometem os humanos. Este modelo continua sendo estudado e estendido para acrescentar mais características (CAMPOS, 2008; SANTO, 2014). Mais recentemente, o modelo foi estendido para levar em conta efeitos bioquímicos associados aos eletrofisiológicos (BONDARENKO, 2014). Entre as aplicações desses modelos, está o estudo dos efeitos da Doença de Chagas sobre o comportamento dos canais iônicos.

A Doença de Chagas é uma infecção endêmica em várias áreas das Américas do Sul e Central. Causa uma distinta e frequentemente fatal inflamação do músculo cardíaco, a chamada miocardite. A cada ano, mais de 20000 novos casos são registrados e pelo menos 8000 recém nascidos contraem a doença durante a gestação (PAN AMERICAN ORGANIZATION; WORLD HEALTH ORGANIZATION, 2014). Durante estágio avançado da doença, extra-sistoles ventriculares e taquicardia têm sido observadas em casos de miocardite aguda. Experimentos com miócitos cardíacos na fase aguda da doença de Chagas sugerem a diminuição corrente transiente de saída de potássio $I_{K_{to}}$, independente corrente de cálcio Ca^{2+} em corações caninos (PACIORETTY et al., 1995).

A Figura 5 apresenta as formas de onda do potencial de ação dos miócitos para controle e ratos chagásicos no estágio agudo da doença. Durante os experimentos, a temperatura foi controlada e mantida entre 25°C e 28°C. Os potenciais de ação foram elicitados via corrente de passagem de transmembrana com amplitudes de 1nA, 5ms de duração a uma frequência de 1Hz. A taxa de amostragem utilizada para registro foi de 10KHz (CAMPOS, 2011).

Os conceito aqui apresentados serão utilizados na Seção 5.1, onde um *workflow* é construído para investigar o efeito da corrente de $I_{K_{to}}$ no potencial de ação do modelo proposto por Bondarenko et al. (2004).

2.1.2 Sistema Imune Humano

O corpo humano é protegido contra invasões por vírus, bactérias e parasitas por um sistema complexo de células, tecidos e órgãos que formam o Sistema Imune Humano (SIH). Entender como o SIH funciona é essencial para obter novas descobertas sobre sua natureza e lidar efetivamente contra doenças. As superfícies do corpo humano são protegidas pelo tecido epitelial, que constitui uma barreira física entre os ambientes internos e externos

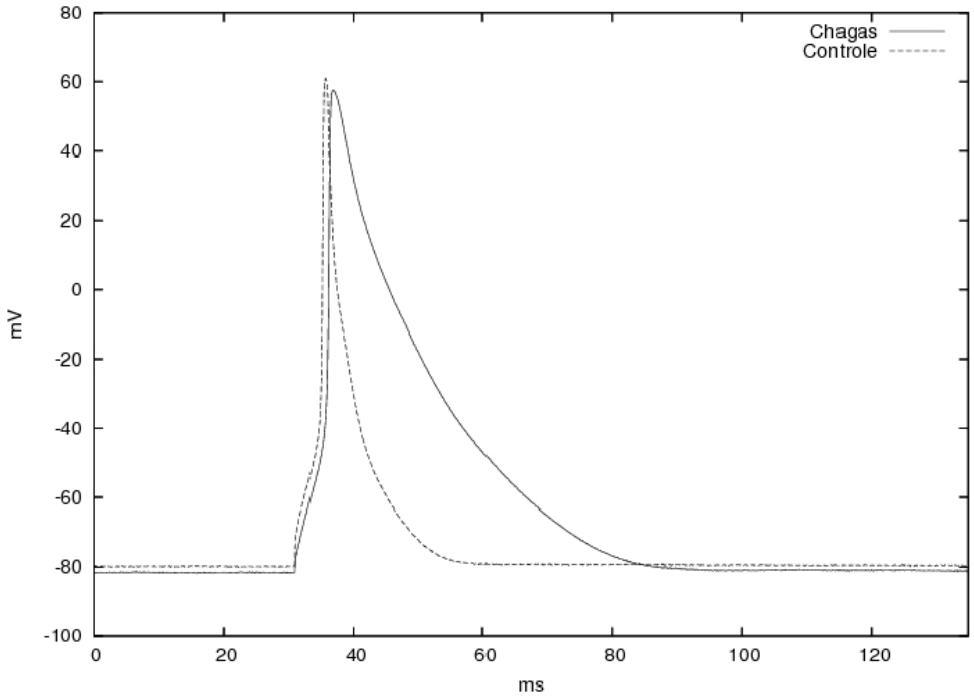


Figura 5 – Experimentos *in vitro*: formas de onda de controle e miócitos de ratos chagásicos. Adaptado de Campos (2011).

do corpo. O tecido epitelial forma um bloqueio eficaz contra o meio ambiente externo. Entretanto, eventualmente, ele pode ser atravessado ou invadido por patógenos, causando infecções. Depois de atravessar o epitélio, os patógenos encontram as células e moléculas do SIH e dá-se início a uma resposta inflamatória.

Uma primeira linha de defesa consiste de barreiras físicas, como a pele e membranas mucosas. Quando as barreiras são atravessadas, o SIH combate os invasores com um conjunto de células especializadas, tentando impedir que estes se instalem e se multipliquem no organismo. O SIH é composto por dois sistemas que interagem em si, o sistema imune inato e o sistema adaptativo (PAUL, 2008). O sistema imune inato é responsável pelas defesas não específicas, mais imediatas e fortes, que previnem ou limitam as infecções pela maioria dos microrganismos patogênicos. As células conhecidas como os leucócitos, reconhecem partes específicas dos patógenos, aqui designados como antígenos. O sistema imune adaptativo está presente apenas nos vertebrados e depende principalmente do reconhecimento executado por linfócitos, que possuem a capacidade de distinguir um patógeno e direcionar a ele uma resposta imune mais específica e mais forte.

A resposta inicial do corpo a um estresse biológico agudo, tal como uma infecção bacteriana, é uma resposta inflamatória aguda (JANEWAY et al., 2010). A estratégia do SIH é manter residente nos tecidos um conjunto de células de guarda, os macrófagos, para procurar por qualquer sinal de infecção. Quando encontram esse sinal, os macrófagos começam a produzir uma série de substâncias, como as citocinas, que recrutam outras

células, tais como os neutrófilos e moléculas do sistema imune inato a partir dos vasos sanguíneos para o local de infecção.

A endotoxina *Lipopolissacarídeo* (LPS) é um imunoestimulante potente que pode induzir uma resposta inflamatória aguda comparável à de uma infecção bacteriana. Após a lise das bactérias pela ação das células do SIH, o LPS pode ser libertado no hospedeiro, intensificando a resposta inflamatória e a ativação algumas células do sistema inato. O LPS pode desencadear uma resposta inflamatória através da sua interação com receptores da superfície de algumas células. Quando há a ligação com receptores de macrófagos e outras células brancas, isso induz estas células a realizar a fagocitose. A fagocitose degrada as bactérias internamente nas células e estas secretam proteínas conhecidas como citocinas ou quimiocinas, bem como outras moléculas.

O processo inflamatório tem muitos benefícios no controle da infecção. Além do recrutamento de células e de moléculas da imunidade inata a partir dos vasos sanguíneos para o local do tecido infectado, ele aumenta o fluxo de linfa, contendo microrganismos e células que transportam partes do agente patogênico para os vasos linfáticos vizinhos, em que estas células apresentam essas partes para os linfócitos para iniciar a resposta adaptativa. Uma vez que a resposta do sistema adaptativo é ativada, a inflamação também recruta as células efetoras do sistema imune adaptativo para o local da infecção.

Na Seção 5.2 outro estudo de caso cria um *workflow* para modelar parte da dinâmica básica do sistema inato utilizando o JynaCore API como uma ferramenta de descrição e motor de simulação para modelos de Dinâmica de Sistemas (KNOP et al., 2012b) de forma adimensional. Adicionalmente, outro *workflow* propõe o uso de Metamodelos de Dinâmica de Sistemas para uma configuração planar de seções de tecido (KNOP et al., 2012c). Será também apresentado o uso cenários de metamodelos de Dinâmica de Sistemas para a criação de novos experimentos a partir de um mesmo modelo básico.

2.2 Biologia Sistêmica

Sistemas biológicos são conhecidos como sistemas complexos nos quais um grande número de elementos simples interagem entre si para produzir comportamentos complexos. Entretanto, na realidade, existem conjuntos de elementos bem distintos, que interagem seletivamente para exibir um comportamento coerente (KITANO, 2002).

Kitano (2002) ainda destaca as possíveis interfaces para troca de conhecimento entre os dois ciclos de descobertas, onde experimentos *in silico* podem ser aplicados na produção de fármacos e terapias junto da pesquisa tradicional. Como uma peça chave da indústria farmacêutica, experimentos computacionais podem implicar na redução de custos e aumento no sucesso do desenvolvimento de novos produtos e serviços médicos. A Figura 6 ilustra os pontos onde há a troca de informações entre os dois ciclos de

experimentação.

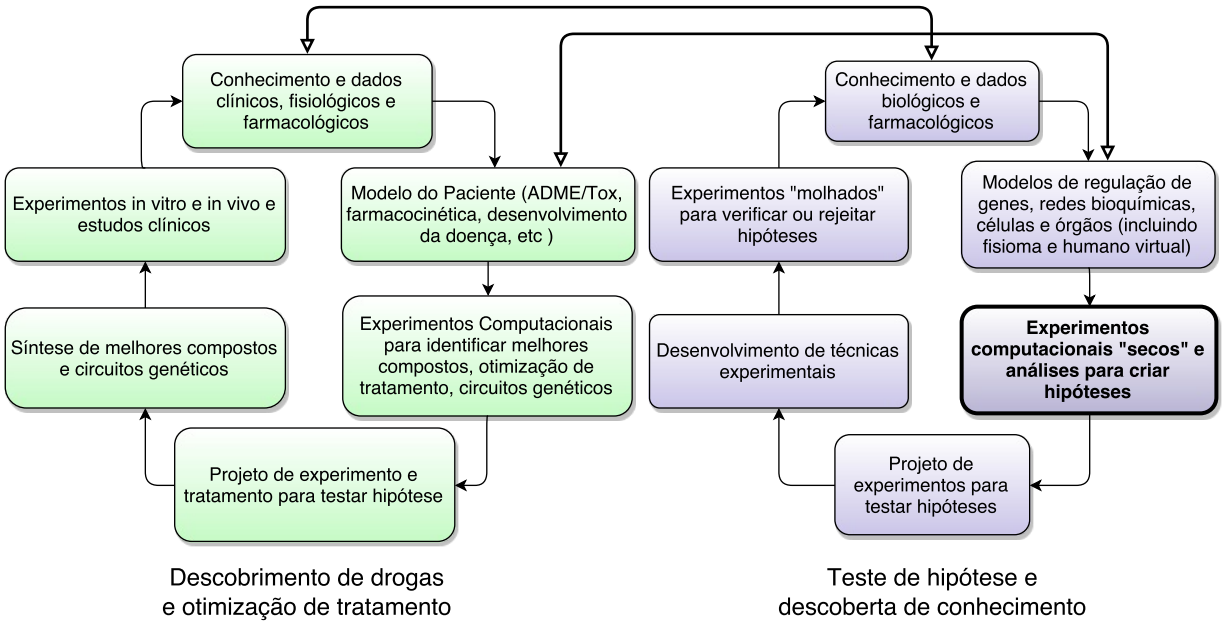


Figura 6 – Diagrama adaptado de Kitano (2002) apresentando uma possível ligação entre a modelagem de sistemas biológicos e a descoberta de novas drogas e tratamentos. Destaque para os pontos onde este trabalho pretende apresentar contribuições.

A biologia computacional busca criar conhecimento através de duas grandes vertentes: pela extração de conhecimento (ou mineração de dados, ou aprendizado de máquina) ou pela análise por simulação. A predição de estruturas proteicas a partir de uma sequência de DNA e a inferência de uma rede regulatória de genes são as principais ferramentas para a extração de conhecimento. Já a análise por simulação, se propõe a predizer a dinâmica do sistema estudado. Assim, hipóteses podem ser confrontadas pela validação do comportamento do modelo contra observações experimentais(KITANO, 2002).

Este trabalho foca na construção de experimentos *in silico* baseados em modelagem e simulação na proposta de um novo método para publicação de experimentos, o SEEM, amparado por um ambiente computacional. Apesar do método SEEM não restringir seu uso a modelos por simulação, técnicas de extração de conhecimento estão fora do escopo deste trabalho e não serão discutidas neste referencial teórico.

Independente da natureza e domínio, Medeiros, Vossen e Weske (1995) citam três requisitos fundamentais para um experimento científico: reprodutibilidade, metodologia e ambiente de execução controlado. Outros cientistas devem conseguir validar ou refutar os resultados ao conduzir experimentos semelhantes ou em futuras pesquisas. Ou seja, os experimentos devem ser bem caracterizados de forma que sejam reprodutíveis. Um cientista cria um experimento descrevendo uma série de passos para validar uma hipótese. O método empregado pode ser novo ou reaproveitar protocolos desenvolvidos previamente,

já que parte do experimento pode ser utilizada em um outro. Portanto, o reuso de métodos é uma importante característica para a experimentação. O terceiro requisito é controle da condução de um experimento. Isto é, o acompanhamento da execução dos passos individualmente. Realizar uma análise se tarefas podem ser realizadas em paralelo ou se devem ser refeitas devida a alguma falha. Como a caracterização de processos é uma preocupação recorrente no mundo dos negócios na forma de *workflows*, uma sequência lógica é aplicar suas técnicas e ferramentas ao ambiente científico.

2.3 Modelagem Computacional de Sistemas Biológicos

Segundo Jarrard (2001), um experimento científico busca confrontar uma hipótese a fim de mantê-la, modificá-la ou abandoná-la. Experimentos criam observações, na forma de dados, que provêem os testes necessários para se realizar um confronto entre a especulação e a realidade. Os testes podem ser realizados diretamente sobre a hipótese ou, mais comumente, sobre as predições ou implicações diretas da aceitação da hipótese.

Os experimentos científicos podem ser agrupados em três grandes grupos: os *in vitro*, onde o estudo acontece em um ambiente controlado fora do natural (tubos de ensaio ou placas de Petri); os *in vivo*, onde os estudos ocorrem dentro de um organismo vivo (efeitos de drogas e terapias sobre cobaias); e os *in silico*, onde os estudos são realizados sobre modelos matemáticos, simulações por computador ou análises de grandes conjuntos de dados. Ao grande conjunto dos estudos *in silico* no domínio biológico dá-se o nome de biologia computacional (JARRARD, 2001).

Para buscar o conhecimento sobre um fenômeno biológico através de uma simulação computacional, deve-se primeiro realizar uma observação do domínio em estudo e depois criar uma simplificação que capture algum aspecto de interesse na forma de um modelo. Modelos matemáticos foram usados ao longo do tempo para capturar os aspectos dinâmicos dos sistemas biológicos (HODGKIN; HUXLEY, 1952; TUSSCHER et al., 2004; INGALLS, 2013). Esta seção faz uma breve revisão sobre os tipos de modelagem encontrados na literatura e que estão relacionados com este trabalho.

2.3.1 Equações Diferenciais

Modelos matemáticos biológicos são frequentemente representados por equações diferenciais. As *Equação Diferencial Ordinárias* (EDO) são conjuntos de equações que dependem de apenas uma variável (BOYCE; DIPRIMA; HAINES, 1992). No caso de fenômenos físicos e biológicos é de interesse acompanhar a evolução de concentrações e conceitos do domínio ao longo do tempo. Portanto, frequentemente essa variável é o tempo (PERELSON; WEISBUCH, 1997).

Em Pigozzo et al. (2011) os autores utilizam um sistema de EDOs para modelar a interação entre os elementos do sistema imune inato humano em resposta à entrada de um agente estranho em uma região microscópica de tecido. Também existem estudos da dinâmica do sistema quando há uma bactéria suscetível à fagocitose e outra protegida por uma camada de biofilme (REED et al., 2011). No domínio da eletrofisiologia cardíaca, equações diferenciais são utilizadas para a modelagem do potencial de ação da membrana celular (HODGKIN; HUXLEY, 1952; TUSSCHER et al., 2004; BONDARENKO et al., 2004).

As *Equação Diferencial Parcials* (EDP) são equações que dependem de mais de uma variável (BOYCE; DIPRIMA; HAINES, 1992). Novamente, no caso de fenômenos biológicos, o interesse está nas concentrações de substâncias em função da posição espacial e do tempo. Existem aplicações no sistema imune ao explorar a distribuição tridimensional da resposta imune (ROCHA et al., 2012). Trabalhos mais recentes buscam estudar o acoplamento do sistema imune inato com outros modelos mais complexos como o sistema adaptativo (QUINTELA; SANTOS; LOBOSCO, 2014).

As equações diferenciais são utilizadas, direta ou indiretamente, em todos os modelos dos estudos de caso deste trabalho. O estudo de caso apresentado na Seção 5.1 no domínio da eletrofisiologia cardíaca é descrito por equações diferenciais. A abordagem também é utilizada por Pigozzo (2011) na Seção 5.2.2 para a construção de um modelo reduzido do Sistema Imune Inato Humano e também é a base para a abordagem de Dinâmica de Sistemas, que será detalhada na próxima seção.

2.3.2 Dinâmica de Sistemas

Dinâmica de Sistemas é uma linguagem gráfica desenvolvida por Jay Forrester nos anos 60 para expressar comportamentos econômicos e sociais complexos (FORRESTER, 1961). Usando a teoria de controle, cálculo numérico, diagramas gráficos simples e equações algébricas, a Dinâmica de Sistemas permite que pessoas, sem formação matemática, possam criar e simular modelos dinâmicos complexos. A Dinâmica de Sistemas é utilizada em diversas aplicações modernas como mercado e negócios (BARLAS, 2007; ÖZBAŞ; ÖZGÜN; BARLAS, 2014), simulações do meio ambiente (FORD, 1999; MUETZELFELDT, 2010), análise de recursos hídricos (KOJIRI et al., 2008; SUŠNIK et al., 2012), imunologia (WAKELAND; MACOVSKY; AN, 2007) e modelagem de processos de *software* (ABDEL-HAMID; MADNICK, 1991; KELLNER; R; D, 1999).

Um dos principais elementos da Dinâmica de Sistemas é o uso dos chamados diagramas de estoque e fluxo que, com equações algébricas bem simples e uma linguagem gráfica, conseguem descrever sistemas com comportamento dinâmico complexo, com não linearidades e fortemente acoplados. Os diagramas de estoque e fluxo definem uma linguagem formal e quantitativa para expressar os principais estados de um sistema e suas

relações internas de causa e efeito. Eles são compostos por quatro elementos básicos: estoques, fluxos, auxiliares e informações.

Um estoque expressa uma quantidade que pode ser acumulada no sistema. No modelo, por exemplo, pode-se expressar as populações de leucócitos como estoques. Um fluxo expressa como o nível do estoque varia durante o tempo. O modelo que nível de um estoque só pode ser alterado devido ao efeito de um fluxo de entrada (aumentando a quantidade armazenada) ou de um fluxo de saída (diminuindo a quantidade armazenada). Um auxiliar é uma forma de isolar relações importantes ao sistema e são calculados com base nos valores dos estoques. Um auxiliar não pode nunca alterar diretamente o valor de um estoque, mas pode ser utilizado para calcular um valor de um fluxo. Os elementos de informação são uma representação visual da interdependência dos elementos do modelo. Quando um fluxo ou auxiliar utiliza um nível de estoque em suas equações, traçamos uma fina seta do estoque para o fluxo ou para o auxiliar. Assim, pode-se acompanhar visualmente e ver os laços de realimentação e causalidade, o que aumenta em muito a compreensão do sistema.

A Figura 7 mostra um exemplo de um modelo genérico de um diagrama de estoque e fluxo. Os estoques são representados por retângulos, os auxiliares por círculos e os fluxos por válvulas atravessadas por setas largas. O símbolo da nuvem é um estoque infinito e representa um estoque que está fora dos interesses do modelo específico. Estoques infinitos são considerados capazes de sempre fornecer e receber quaisquer valores indefinidamente. Por trás dessa representação gráfica está um modelo matemático representado por um sistema de equações diferenciais ordinárias (EDO).

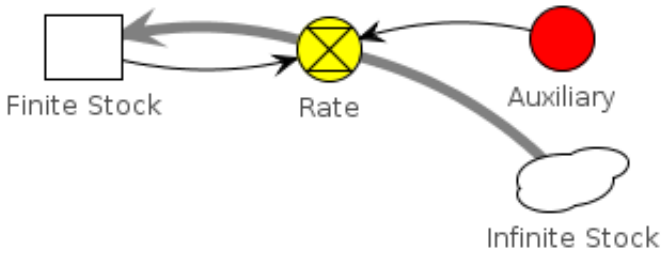


Figura 7 – Os elementos básicos de Dinâmica de Sistemas: estoque finito representado como uma caixa branca; estoque infinito representado como uma nuvem branca; um fluxo como uma válvula atravessada por uma seta grossa que liga dois estoques; variáveis auxiliares representadas por círculos; dependência de informações representadas por linhas finas.

2.3.3 Metamodelos de Dinâmica de Sistemas

Um modelo deve responder a um conjunto de perguntas em uma linguagem que o usuário entenda. Dinâmica de Sistemas é uma linguagem formal que pode ser utilizada

por grupos multidisciplinares como uma ferramenta de troca de conhecimento. Todos os elementos básicos e as relações de subsistemas podem ser capturados e associados para, só então, o comportamento do sistema completo emergir dessas inter-relações. Entretanto, quanto mais elementos de subsistemas forem adicionados, mais complicada fica a leitura do modelo. Márcio Barros propôs uma nova linguagem de modelagem chamada de metamodelagem de Dinâmica de Sistemas (BARROS, 2001) que auxilia no controle da complexidade crescente, melhora a legibilidade e manutenção dividindo o processo de modelagem em duas fases com interesses diferentes.

A Figura 8 mostra a relação entre as duas fases da criação de um modelo utilizando a Metamodelagem de Dinâmica de Sistemas: a definição de um modelo de domínio e a descrição do sistema a ser modelado por um modelo de instância; e a modelagem de domínio descreve estruturas de comportamento por meio de *classes* que, mais tarde, serão instanciadas para modelar todo o sistema.

Um modelo de instância usa as classes previamente definidas para criar elementos que podem ser associados e ter suas propriedades alteradas para descrever o sistema em estudo. Um modelo de instância pode ser simulado, pois é nele que ficam todos os estados do modelo. Já o modelo de domínio não pode ser simulado porque somente descreve os blocos de construção básicos disponíveis para a construção dos modelos de instância.

Um passo opcional na modelagem de domínio consiste na criação dos chamados modelos de cenários. Esses modelos são utilizados para encapsular algumas alterações estruturais do modelo ou das instâncias. Esses cenários podem ser reutilizados em vários modelos de instância de um mesmo domínio como uma forma de alterar as instâncias a que forem conectados. Uma utilização explorada neste trabalho é o uso de cenários para configuração de experimentos.

2.3.3.1 Modelo de Domínio

Um modelo de domínio tenta capturar as propriedades, o comportamento e as relações entre um conjunto de conceitos dentro de um domínio de conhecimento. É uma descrição abstrata e, por conseguinte, não pode ser executada em uma simulação. Para construir um modelo de domínio, o modelador deve saber criar diagramas de estoque de fluxo de Dinâmica de Sistemas.

O principal conceito dos modelos de domínio é a *classe*. Uma classe é uma representação de uma família de elementos que compartilha o mesmo comportamento e propriedades. Para descrever uma classe, utiliza-se um diagrama de estoque e fluxo como um sistema completo e isolado.

Durante a construção do modelo de domínio, pode-se definir relações entre as classes. Essas relações serão utilizadas durante a construção de modelos de instância para

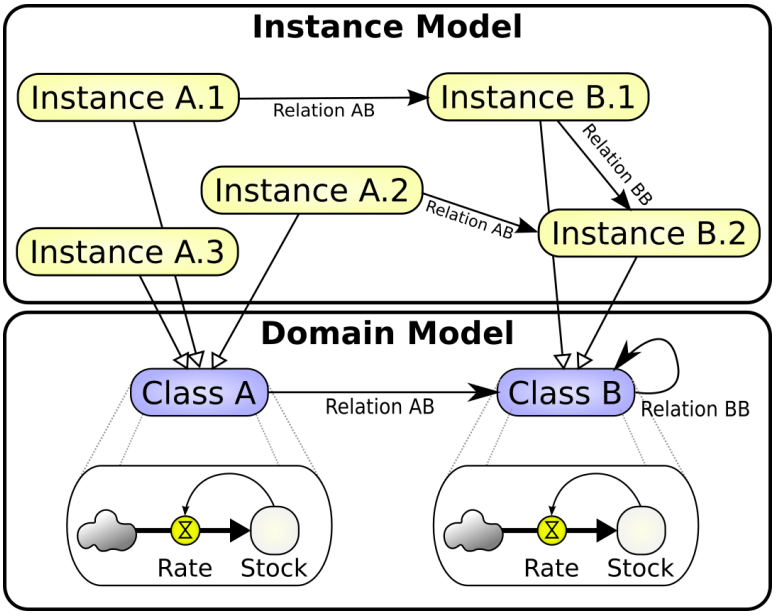


Figura 8 – Na metamodelagem de Dinâmica de Sistemas o modelos de domínio definem as estruturas de comportamento identificadas por classes que serão utilizadas nos modelos de instância para criar o modelo final.

definir uma ligação estrutural entre as instâncias. Isso permite que uma instância tenha acesso à informação ou estado de outra instância.

Para um exemplo simples, é possível considerar duas classes: `Class A` e `Class B`. A `Class A`, na Figura 9, possui um estoque finito, um fluxo e um auxiliar. Já a `Class B`, na Figura 9, tem apenas um auxiliar e uma propriedade. Os elementos auxiliares com linhas tracejadas são elementos externos, referenciados por relacionamentos, que irão referenciar outras instâncias.

Com a utilização de relacionamentos entre classes de um modelo de domínio, é possível construir estruturas complexas relacionando as suas instâncias. Essas inter-relações vão mudar o comportamento das instâncias e, assim, de todo o sistema. Entretanto, isso mantém a estrutura interna de classe simples e de fácil entendimento.

2.3.3.2 Modelo de Instância

Após a criação do modelo de domínio, é possível começar a modelar nosso sistema usando as classes definidas. Um modelo de instância é uma realização de um único modelo de domínio utilizando as classes nele definidas. Todas as estruturas das instâncias são, inicialmente, geradas com base em suas respectivas classes, mas cada uma possui suas próprias propriedades e estados independentemente umas das outras. A partir de uma única classe pode-se criar várias instâncias e estas podem ter suas estruturas alteradas posteriormente por modelos de cenários. As instâncias podem ser associadas pelos relacionamentos definidos no modelo de domínio e esta conexão irá definir o seu comportamento

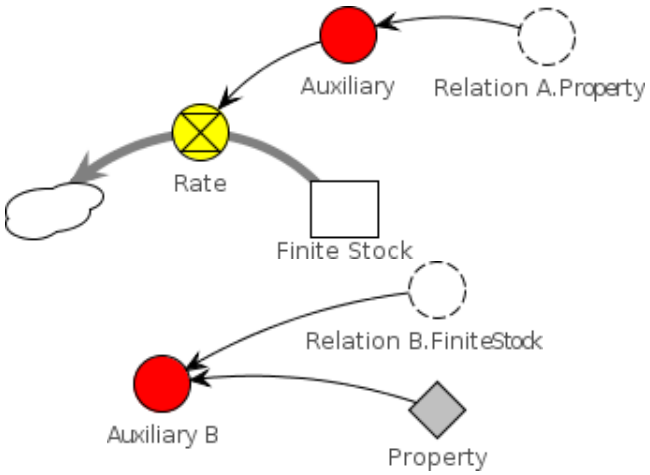


Figura 9 – Modelo de domínio para duas classes diferentes: a classe `Class A` possui um auxiliar, um estoque finito e um fluxo conectando a um estoque infinito. A classe `Class B` possui apenas um auxiliar e uma propriedade constante. Ambas as classes possuem referências para um auxiliar externo representado por um círculo tracejado, acessado via relacionamentos.

com base nas influências que seus estados individuais causam mutuamente.

A Figura 10 apresenta um exemplo simples de um modelo de instância genérico com três instâncias `Instance One.A`, `Instance One.B` and `Instance Two.A`. As duas primeiras são realizações da classe `Class A` e a última uma realização da classe `Class B`. Todas as instâncias são conectadas por relacionamentos.

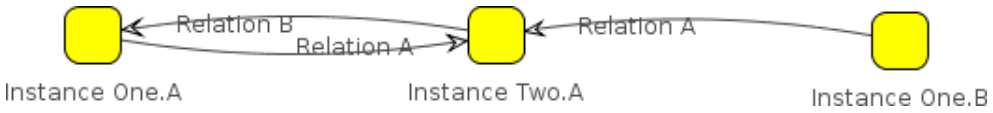


Figura 10 – Um modelo de instância simples com três instâncias: `Instance One.A` e `Instance One.B` são instâncias da classe `Class A` e a `Instance Two.A` uma instância da classe `Class Two`. Todas as instâncias podem ser conectadas pelos relacionamentos definidos em suas respectivas classes do modelo de domínio.

2.3.3.3 Modelo de Cenário

Um modelo de cenário é definido como uma alteração interna e reutilizável nas instâncias e que vai alterar o seu comportamento dinâmico. Esta mudança pode ser um novo elemento adicionado, uma modificação na estrutura dos elementos existentes ou uma redefinição de suas equações. Um modelo de cenário é criado para um modelo de domínio específico. Portanto, não pode ser compartilhado entre domínios diferentes. Um cenário está descrito no mesmo nível de abstração que as classes de domínio, mas é aplicado em modelos de instância. Depois de definir um modelo de cenário, ele pode ser reutilizado em qualquer modelo de instância criado a partir do mesmo modelo de domínio.

Um cenário é composto por uma lista de conexões para classes de domínio. Cada conexão definida pode adicionar elementos a uma instância ou ajustar as suas equações. Conexões de cenários são descritas da mesma forma que em classes de domínio: simples elementos de Dinâmica de Sistemas que serão adicionados às instâncias quando conectados. No modelo de instância, após a aplicação de um cenário em uma instância, esses novos elementos ou modificações nas equações alteram o seu comportamento, tornando-a diferente das suas outras instâncias da mesma classe.

A Figura 11 apresenta um modelo de domínio com duas classes, `Class A` e `Class B`, com relacionamentos mútuos indicados como setas sólidas.

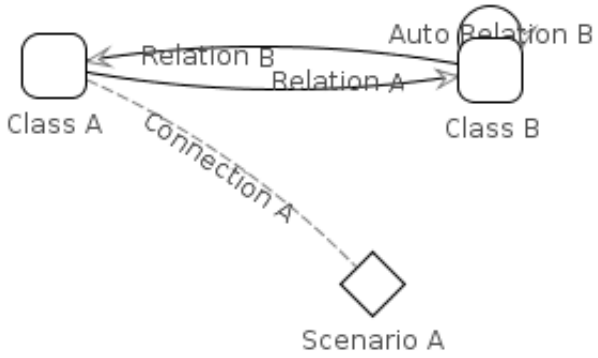


Figura 11 – Metamodelagem de Dinâmica de Sistemas: um modelo de cenário pode ser definido no modelo de domínio para alterar a estrutura de uma instância quando conectado a ela.

O cenário `Scenario A` é representado por um losango e é associado com a classe `Class A` por uma linha tracejada. Na Figura 12 é possível observar um modelo de instância com duas instâncias de classe. O cenário `Scenario A` é conectado com a instância `Instance A` e a conexão é representada pelo nome do cenário entre parênteses angulares duplos abaixo do nome de instância.

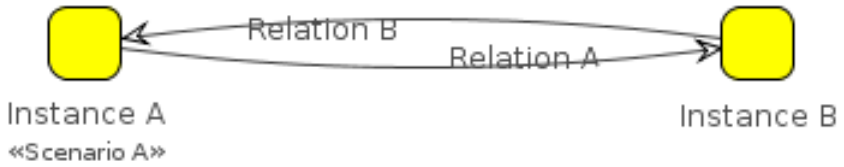


Figura 12 – Um modelo de instância com um cenário `Scenario A` conectado com a instância `Instance A`. O cenário `Scenario A` deve ser definido após a construção do modelo de domínio, assim todas as instâncias podem utilizá-lo.

Existem várias oportunidades para usar modelos de cenários durante a modelagem do SIH utilizando Dinâmica de Sistemas, tais como: simplificar a definição das condições

de contorno sem acrescentar mais elementos na definição de classe; adicionar efeitos da conformação celular; acrescentar elementos para discretização no espaço; efeitos de drogas agindo em parâmetros específicos; estímulos experimentais simulados; e comportamento deficiente da célula devido à doenças. Utilizando cenários também é possível reduzir a necessidade de adicionar novas classes de domínio mais específicas em nosso modelo. Os metamodelos de Dinâmica de Sistemas e os modelos de cenários foram utilizados no estudo de caso na Seção 5.2.3.

2.4 Workflows Científicos

O termo *workflow* surgiu e foi inicialmente utilizado no contexto de aplicações de negócio. Um processo de negócio é uma instância de uma tarefa bem definida, que seja frequentemente repetida como parte de um processo comum de uma empresa. A *Workflow Management Coalition* ou WfMC (LAWRENCE, 1997) define um *workflow* como a automação de um processo de negócio, como um todo ou em parte, durante o qual documentos, informações ou tarefas são passadas de um participante para outro para ação, de acordo com um conjunto de regras procedurais. As tarefas que estejam relacionadas com automação computacional dos processos de negócio são do domínio da gerência de *workflows* (BARGA; GANNON, 2007).

Os *workflows* criados para conduzir um experimento *in silico* ficaram conhecidos como *workflows científicos*. Um *workflow* científico facilita ou automatiza um processo científico como um todo ou em parte. Isto normalmente envolve a execução de um experimento em um ambiente controlado (em um laboratório, por exemplo), seguido por pós-processamento dos dados e a interpretação dos resultados. De forma mais especifica, pode-se definir um *workflow* científico como uma descrição dos processos que um cientista precisa executar, em uma certa ordem, para criar um "produto científico" como uma análise de dados, visualização e publicação (LUDÄSCHER et al., 2006).

As diferenças entre os *workflows* de negócio e científicos justificam a criação de um estudo e ferramentas em separado. Enquanto no âmbito dos negócios eles representam tarefas bem definidas dentro de uma organização, os científicos devem ser mais dinâmicos e adaptáveis para acompanhar o processo de exploração de novas técnicas, métodos ou modelos científicos (BARGA; GANNON, 2007). Um modelo de *workflow* científico também impõe uma grande demanda na conversão de dados, pois pode-se utilizar uma série de ferramentas diferentes. Os processos adaptadores lidam com conversões de formatos e unidades, movimento de dados, controle de laços e podem ser considerados processos de uso geral.

Do ponto de vista do usuário final, a automação é o maior benefício da abordagem com *workflows* científicos (LUDÄSCHER et al., 2006). Por exemplo, um cientista pode

querer executar a mesma fila de análise de dados repetidamente com vários conjuntos de dados de entrada diferentes ou alterando apenas alguns parâmetros. O sistema deve ser capaz de registrar e, posteriormente, reproduzir todas as interações do usuário durante a execução do *workflow*. Os relatórios gerados pela execução devem ser acompanhados por proveniência da informação para facilitar a interpretação dos resultados e auditoria. A descrição do *workflow* junto aos registros de execução pode servir de evidências ou explanação para os resultados da análise. LUDÄSCHER et al. (2006) ainda destacam que certos tipos de publicações científicas poderão incluir metadados de descrições e proveniência de *workflows* científicos de forma a confirmar evidências para os resultados publicados. Outros requerimentos comuns do ponto de vista do cientista incluem a necessidade de desenvolver o projeto de *workflows* de forma mais intuitiva e focada em reuso e readaptação. Além disso, a reconfiguração do *workflow* deve ser fácil, sem necessidade de ter que, praticamente, reescrever todos os parâmetros.

O ciclo de vida de um experimento pode ser visto como os passos necessários e múltiplas interações para a comprovar ou refutar uma hipótese (HULL et al., 2004). Diversas abordagens e objetivos implicam em fases de ciclos de vida diferentes. O experimento pode ser observado como parte da publicação de um artigo (KOOP et al., 2011), como um processo de montagem de software (FOMEL; HENNENFENT, 2007; FOMEL et al., 2013) ou como uma composição de serviços em *grids* de computador (FAHRINGER et al., 2007). Já Mattoso et al. (2010), de forma mais geral, divide o ciclo de vida de um experimento em três fases distintas: composição, execução e análise.

Durante a composição de um experimento, o cientista organiza os modelos, métodos, programas e dados de entrada de forma lógica para realizar o sequenciamento que vai gerar os dados de saída. Assim, durante a composição do experimento é que se cria, edita e manipula o *workflow* do experimento. Um *workflow* pode ser representado por grafos direcionados onde as tarefas a serem realizadas são representadas por vértices e suas arestas identificam as suas interdependências. Através desta representação o usuário consegue realizar a composição dos experimentos e visualizar a estrutura do *workflow*. Outras representações formais como Redes de Petri (AALST, 1998), álgebra de processos (BAETEN, 2005) e até Unified Modeling Language (UML) (DUMAS; HOFSTEDE, 2001) também permitem que a verificação da integridade do *workflow* ou do seu comportamento.

A fase execução é quando o experimento é executado sobre alguma plataforma computacional. Neste ponto, todos os processos e programas, bem como seus parâmetros de execução devem estar bem definidos para que o sistema de gerenciamento de *workflow* consiga executá-lo de forma correta (ZHAO et al., 2007). Nesta fase é que diversos modelos de execução podem ser utilizados (KEPLER PROJECT, 2004), bem como técnicas de paralelismo (OGASAWARA et al., 2009) sobre ambientes distribuídos e grades de computadores para computação de alto desempenho (MCGOUGH et al., 2004).

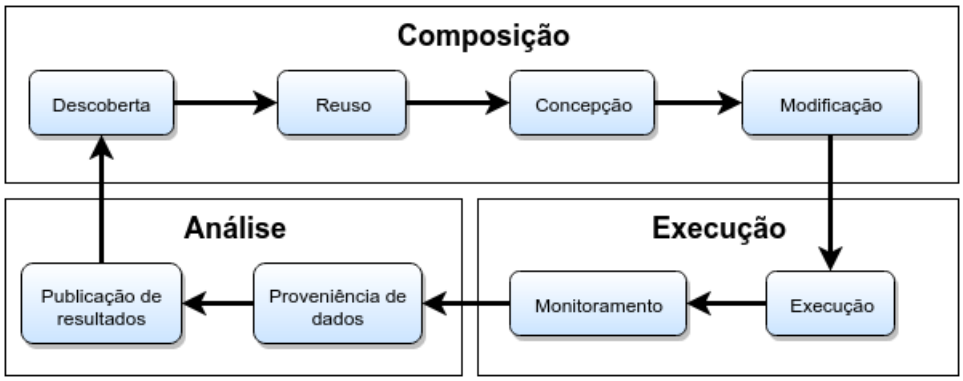


Figura 13 – As três fases principais do ciclo de vida de um experimento *in silico*: a composição, execução e análise.

A complexidade potencial de um *workflow* científico demanda uma profunda investigação de todas as atividades envolvidas e um exame detalhado das interdependências e dados. Consequentemente, é importante projetar *workflows* modulares agrupando atividades similares ou relacionadas com suas próprias variáveis, lógica, restrições de dependência, modelo de segurança e tratamento de exceções (AKRAM; MEREDITH; ALLAN, 2006). Um grande repositório de componentes inter-relacionáveis e reutilizáveis abre um extenso leque de possibilidades para processamento e relacionamento de diversos dados científicos. Ao se modularizar os *workflows*, componentes modulares podem ser criados de forma escalonável e confiável para servir como unidades reaproveitáveis.

O ciclo de vida de experimentos pode ser capturado por uma série de ferramentas conhecidas como Scientific Workflow Management System*s* (SWfMS). Entre as ferramentas mais populares utilizadas para composição, execução e análise de *workflows*, podemos destacar projetos como o Kepler (KEPLER PROJECT, 2004; BARSEGHIAN et al., 2010), VisTrails (VISTRAILS PROJECT, 2015), Taverna (TAVERNA PROJECT, 2007), Pegasus (DEELMAN et al., 2005) e Triana (SHIELDS; TAYLOR, 2004).

2.5 Repetição e Reprodução de Experimentos

Os dois propósitos básicos de uma publicação são anunciar um resultado e convencer os leitores de que ele está correto (MESIROV, 2010). A busca pela pesquisa reproduzível na computação tem início na década de 90 com a exigência de que artigos de geofísica, publicados na Stanford Exploration Project[1], fossem acompanhados de todos os arquivos de construção de código (SCHWAB; KARRENBACH; CLAERBOUT, 2000) via arquivos *makefiles* (STALLMAN; MCGRATH, 2002). A preocupação com a reprodução dos resultados dos experimentos computacionais publicados colocam a Computação como o terceiro vértice do triângulo da Ciência, junto da teoria e experimentação (LEVEQUE,

[1] A página do projeto pode ser encontrada em <http://sepwww.stanford.edu>

2006). Pode-se ainda destacar:

> Nestes dias, as revistas científicas e matemáticas são preenchidas com imagens bonitas de experimentos computacionais que o leitor não tem nenhuma esperança de repetir. Mesmo brilhantes e bem intencionados, cientistas da computação frequentemente fazem um trabalho pobre em apresentar o seu trabalho de uma maneira reprodutível. Os métodos são frequentemente definidos vagamente, e mesmo se eles são cuidadosamente definidos, eles normalmente teriam de ser implementados a partir do zero por parte do leitor a fim de testá-los. (LEVEQUE, 2006, tradução nossa).

Os conceitos de replicabilidade e reprodutibilidade aparecem como sinônimos em alguns trabalhos e são intercambiados livremente. Entretanto, há uma vertente onde a discussão é que são formas distintas de se revisar um experimento (DRUMMOND, 2009). Seguindo esse segundo ponto de vista, um experimento é repetido ou replicado quando um segundo experimento é criado, utilizando-se exatamente todo o aparato experimental e dados de entrada e configuração do original. Um experimento é reproduzido quando atinge os mesmos resultados, em total ou em parte, mas com algum componente de seu *workflow* substituído por qualquer motivo. Este trabalho utiliza essa segunda posição na qual há a distinção entre repetição e reprodução.

Mais especificamente em relação à simulação de processos biológicos, cientistas em diferentes escolas, com diferentes abordagens, se reuniram para descrever uma diretriz de relatório de experimentos denominado MIASE (WALTEMATH et al., 2011a). Esse protocolo descreve um conjunto mínimo de informações que deve ser provido para tornar uma descrição de um experimento de simulação disponível para outros pesquisadores. Um experimento compatível com o MIASE inclui: uma lista de modelos para uso (e modificações necessárias); todos os procedimentos para simulação e em qual ordem; o processamento dos dados numéricos resultantes e a descrição da saída final. A lista a seguir expõe a estrutura necessária para que a descrição de um experimento de simulação seja compatível com o MIASE:

1. Todos os modelos utilizados no experimento devem ser identificáveis, acessíveis e completamente descritos;
   a) A descrição deve ser acompanhada dos modelos ou uma forma inequívoca de acessá-los;
   b) Os modelos devem vir com todas as equações, parâmetros e condições iniciais e de contorno;
   c) Se o modelo não estiver em um padrão comum, o código deve ser disponibilizado ou uma descrição completa que permita reimplementá-lo.
2. Uma descrição precisa dos passos de simulação e outros procedimentos utilizados pelo experimento devem ser fornecidos
   a) Todos os passos devem ser descritos claramente, incluindo os algoritmos a serem utilizados, os modelos a serem utilizados em cada simulação, a ordem dos passos de simulação e o processamento de dados entre cada passo da simulação;
   b) Toda informação para a implementação de cada passo de simulação deve ser fornecida ou referenciar uma fonte com as informações de forma inequívoca;

c) Se um passo de simulação é realizado utilizando um programa de computador cujo código fonte não está disponível, toda informação necessária para a reprodução como algoritmos, métodos de discretização e integração deve ser fornecida;
d) Se for sabido que a simulação irá conduzir a diferentes resultados, quando executada em diferentes ambientes de simulação, uma explicação deve ser dada de como o modelo terá que atingir o propósito do experimento.

3. Toda informação necessária para se obter os resultados numéricos deve ser fornecida
   a) Todos os passos de pós-processamento aplicados nos resultados numéricos das simulações devem ser descritos em detalhes. Isso inclui a identificação dos dados a serem processados, a ordem que as modificações são aplicadas e também sua natureza;
   b) Se as informações resultantes dependem da relação entre resultados diferentes, esses devem ser identificados e especificados.

(WALTEMATH et al., 2011a, tradução nossa)

Os autores reforçam que o MIASE é uma diretriz de relatório que descreve de maneira não ambígua o que foi feito e quais as entidades envolvidas no experimento. Em contrapartida, não é responsável por ditar qual abordagem experimental é correta nem como um experimento deve ser realizado, isso fica a cargo do pesquisador. Também não impõe qualquer declaração sobre a escolha do modelo para representar um processo biológico, nem dita a escolha pelo método de simulação e nem discute a corretude dos resultados da simulação. O MIASE foca na reprodutibilidade do experimento simulado ao invés da repetibilidade. Ou seja, dá ênfase na reprodução do experimento, possivelmente com diferentes configurações de ambiente de simulação e ferramentas. Já a repetibilidade lida com a repetição em um mesmo ambiente de simulação.

Uma das grandes barreiras para a reprodução de experimentos é a não disponibilidade ou obsolescência de *software* (PENG, 2011). Uma das saídas para realizar a repetição de experimentos é o uso de virtualização para capturar o ambiente utilizado e posteriormente realizar a invocação do ambiente localmente ou utilizando um serviço em nuvem (OLIVEIRA et al., 2012; STODDEN; MIGUEZ; SEILER, 2015; MARWICK, 2016). Uma crítica a esta abordagem é que isso não propicia a reprodução do experimento e sim apenas a repetição (SANTANA-PEREZ; PÉREZ-HERNÁNDEZ, 2015). Já Santana-Perez e Pérez-Hernández (2015) adotam uma abordagem ligeiramente diferente: os autores novamente capturam a infraestrutura necessária para posterior reprodução do experimento, mas agrupam as funcionalidades equivalentes como uma só, mesmo que ligeiramente diferentes. Para reproduzir os experimentos, eles utilizam os equivalentes, permitindo que funcionalidades sejam substituídas por outras para cumprir o mesmo resultado.

2.6 Considerações parciais sobre a Revisão Conceitual

Este capítulo apresentou uma revisão dos principais conceitos relacionados com o tema desta tese e uma revisão de trabalhos que mostram a preocupação da academia com a reprodução de experimentos *in silico*. Os conceitos de *workflows* científicos são revisitados

e as principais técnicas empregadas. Adicionalmente, são introduzidos os conceitos de eletrofisiologia cardíaca e imunologia que serão usados posteriormente em nossos estudos de caso.

O próximo capítulo irá apresentar um método chamado *Simulation Experiment Environment Method* (SEEM) para captura de experimentos *in silico* com o objetivo de explicitar a reprodutividade de experimentos.

3 O Método SEEM

Criar, manter e distribuir os inúmeros modelos, simuladores e dados de trabalhos dentro de um grupo de pesquisa motiva na busca por uma solução para registro e publicação dos experimentos. Essa solução deve ser, ao mesmo tempo flexível e modular. Flexível no sentido de não limitar ou onerar o processo de busca, criação e execução dos experimentos virtuais. Modular no sentido assumir que uma característica intrínseca do meio científico é a criação de novas ferramentas e técnicas em ambientes heterogêneos e essas devem ser abraçadas e não combatidas.

Este capítulo descreve em detalhes o método Simulation Experiment Environment Method (SEEM) que é a principal contribuição conceitual deste trabalho. O SEEM serve de base para que novas ferramentas sejam construídas para automatizar, completa ou parcialmente, a reprodução de experimento *in silico* em suas diversas etapas. A adoção do SEEM permite descrever, manter e compartilhar experimentos simulados de forma a diminuir o retrabalho através da exposição formal do estado funcional de suas partes componentes.

3.1 Descrição Geral

O SEEM é um método para que os pesquisadores possam descrever e publicar seus trabalhos de modelagem computacional. Como trabalhos é considerado tanto os modelos, simuladores, dados e os resultados desenvolvidos durante uma pesquisa. Ou seja, todas as partes constituintes do experimento simulado. Tão importante quanto os dados utilizados e gerados, o fluxo de trabalho também deve ser descrito através das relações de consumo, produção e orquestração.

A principal preocupação do SEEM é capturar o mínimo conjunto de dados possível sobre um experimento para guiar a sua posterior reprodução. Esses dados devem ser colhidos e armazenados junto do experimento utilizando alguma forma de persistência. Entretanto, corre-se o risco de chegar em um ponto em que o esforço para realizar as anotações se torne desgastante se obrigar o pesquisador fazê-las durante o desenvolvimento do experimento. O desgaste pode, por fim, levar ao usuário a simplesmente abandonar a documentação do que está sendo realizado de fato. Grande parte desses dados pode ser colhida automaticamente durante o ciclo de vida do experimento se este for realizado dentro de um ambiente computacional.

Além da visão mais conceitual do método, são também apresentados os requisitos de um ambiente computacional no qual os pesquisadores poderão criar, alterar e publicar

seus trabalhos. Esse ambiente será responsável, ao mesmo tempo, pelo registro dos detalhes dos experimentos e pela crítica de sua adequação ao SEEM. Ao ser implementado, ele permite que os experimentos publicados sejam reproduzidos e tenham sua execução monitorada ou registrada para que seus resultados possam ser posteriormente validados. Daqui para frente, será feitas referências a um ambiente que implementa o método SEEM, o Simulation Experiment Environment (SEE). Apesar de alguns trechos deste Capítulo se referirem mais de uma vez ao "ambiente", destaca-se que o método proposto e seus modelos de dados são independentes de uma implementação específica.

É importante destacar que o SEEM não propõe, implanta ou indica técnicas e recomendações da Engenharia de *software* para o desenvolvimento dos componentes de um experimento computacional. Apesar das vantagens de qualidade e manutenção que os processos sistemáticos de desenvolvimento de *software* possam trazer para o desenvolvimento dos programas e ferramentas utilizadas dentro de um experimento científico, fica fora do escopo deste trabalho dar tal contribuição. É assumido que os programas utilizados nos experimentos já são o resultado de um processo de desenvolvimento.

O SEEM pode ser resumido conforme a listagem 3.1.

1. Descrever todos os artefatos utilizados ou gerados dentro do experimento;
2. Descrever todos os processos que realizam consumo e geração de artefatos ou sejam compostos por outros artefatos;
3. Cada processo deve ter sua implementação computacional implantada no ambiente;
4. Se algum elemento estiver ausente ou indisponível no ambiente, isso deve ser enfatizado;
5. Todos os elementos podem receber um conjunto de anotações (manual ou automática);
6. O experimento e seus elementos devem ser criticados continuamente, procurando por inconsistências.

Resumo dos passos do método SEEM.

3.1.1 Papéis dos Utilizadores

O desenvolvimento de um experimento *in silico* é um processo complexo, que envolve várias pessoas com perfis, obrigações e interesses diferentes para atingir um resultado final. Além do próprio desenvolvimento de *software* e toda sua complexidade inerente aos experimentos, há a necessidade de se montar toda uma infraestrutura computacional para sua execução, seja local ou remota. Os dados muitas vezes devem ser trafegados

pela rede e trabalhados antes de serem consumidos após sua geração. O SEEM agrupa as responsabilidades e poderes em papéis distintos para caracterizar as funções dos usuários do método. Em alguns projetos, uma mesma pessoa pode acumular dois ou mais papéis (ou mesmo ser o único envolvido no experimento), mas essa divisão permite categorizar melhor os passos do método.

É definido o papel de "Administrador do Ambiente" como o responsável pela manutenção do ambiente e sistema operacional no qual ele é executado. O Administrador do Ambiente tem acesso completo e total a toda infraestrutura computacional sobre a qual o ambiente é implantado. É exigido dele o conhecimento suficiente (ou acesso a quem detenha o conhecimento) para a implantação dos *softwares* requeridos para que o ambiente opere corretamente. Fica ainda sob sua responsabilidade atribuir os papéis aos demais usuários e equipe de manutenção do ambiente.

O papel "Administrador de Programas" é responsável pela instalação de *softwares* que serão utilizados nos diversos processos dos experimentos. É de sua responsabilidade entrevistar os usuários que solicitam uma nova funcionalidade e realizar a instalação e o acompanhamento das versões disponíveis quando não for possível para o sistema realizar a instalação automática. Ele também deve definir para cada *software* instalado no ambiente um protocolo de teste de funcionamento e verificação de versão para ser utilizado na curadoria dos experimentos.

A curadoria dos experimentos exige um papel de "Administrador de Curadoria" cuja função é acompanhar os testes automáticos realizados pelo ambiente (ou executá-los manualmente) para obter um conjunto de informações sobre o atual estado do experimento. A ele é permitido adicionar ou alterar anotações sobre a autoria e estado da reprodutibilidade dos experimentos. A Figura 14 exibe um diagrama de caso de uso para os três papéis envolvidos na administração global de uma implementação do ambiente.

O papel de "Pesquisador" recai sobre os usuários que irão criar, modificar e publicar experimentos no ambiente. Seu principal interesse é analisar os experimentos existentes, seja pela própria estrutura do fluxo de trabalho ou pelos resultados alcançados. Um Pesquisador também pode criar um novo experimento no sistema a partir do zero ou utilizar-se de um experimento existente já publicado, conforme será detalhado nas próximas seções.

Ao criar um novo experimento (ou importar um já existente), o pesquisador se torna dono desta nova instância do experimento. O dono do experimento controla as permissões de acesso aos outros pesquisadores colaboradores e decide quando publicá-lo. A Figura 15 apresenta um diagrama de casos de uso com as operações comuns do pesquisador referentes ao experimento.

Figura 14 – Casos de uso dos três papéis de administração com funções de manutenção e curadoria dos experimentos.

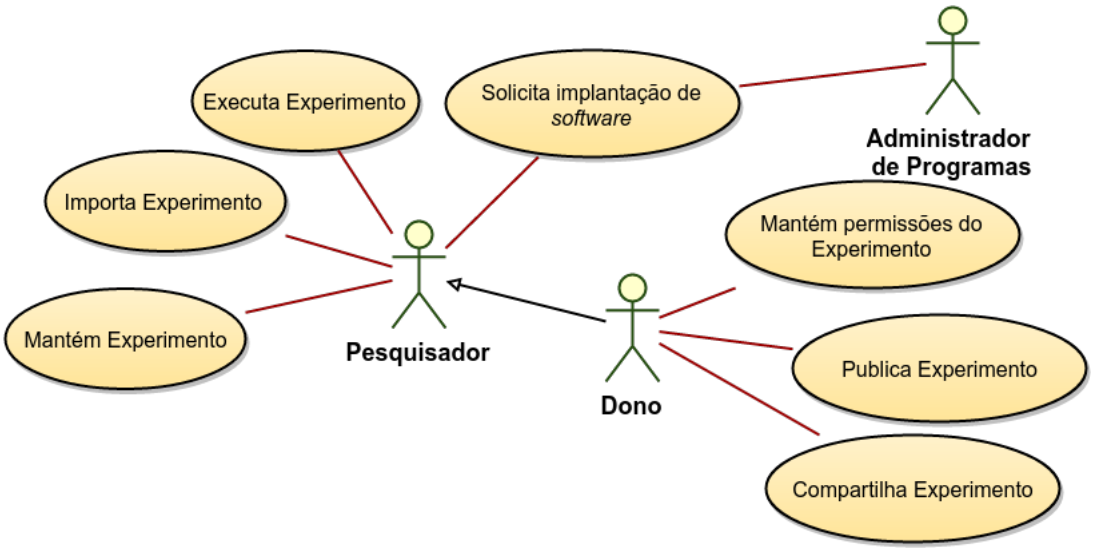


Figura 15 – Casos de uso do papel de Pesquisador com funções de criação e publicação dos experimentos.

3.1.2 Cenários de uso

Existem diversos cenários possíveis dentro deste contexto pelo método. No cenário mais comum, o pesquisador irá desenvolver novos programas para a confecção dos seus experimentos. Esses programas precisam do auxílio do Administrador de Programas para serem implantados no ambiente e torná-los disponíveis para execução. Uma vez implantados, diversos experimentos podem ser criados a partir deles.

Em um outro cenário, um pesquisador pode descrever seu experimento sem im-

plantar os programas de alguns processos. Processos com programas não implantados previamente pelo Administrador de Programas, não invalidam a descrição do experimento, mas invalidam a sua execução. Há portanto, o caso que o pesquisador informa como ele chegou a um resultado, mas o ambiente não é capaz de reproduzir esses resultados. Neste caso, os pontos que impedem sua execução e reprodução são destacados na forma de relatórios pelo ambiente.

Alguns experimentos são realizados exclusivamente através da interpretação de *scripts*. Em um cenário no qual a simulação pode ser realizada por *scripts* disponíveis dentro do experimento, os únicos programas necessários para serem implantados são os respectivos interpretadores.

Para o caso em que um experimento é gerado a partir de um outro existente, algumas situações podem ocorrer e influenciar diretamente na reprodução do experimento. A primeira é a pura restrição de acesso por parte do autor original a alguma parte do experimento. O autor pode, por exemplo, marcar explicitamente que um conjunto de dados possui caráter sigiloso ou pode causar um conflito de interesses, não devendo ser disponibilizado. Um processo é capaz de utilizar internamente um *software* proprietário e seu uso pode estar restrito para alguns experimentos de um projeto. Em ambos os casos, os motivos devem estar explicitados através de anotações informadas pelos pesquisadores ao descrever o experimento.

3.1.3 Crítica

Durante todas as fases de confecção do experimento é possível realizar uma série de críticas para guiar o autor de forma que seu experimento seja reprodutível. A crítica deve ser aplicada continuamente cada fase do ciclo de vida do experimento. Para o caso de uma implementação na forma de ambiente, as críticas podem ser adicionadas às ações disponíveis aos usuários durante a construção e execução do experimento ou serem realizadas posteriormente após à publicação do experimento, varrendo todos os seus elementos como parte de um processo de curadoria.

Nas próximas seções serão listadas as inconsistências que podem ocorrer durante o ciclo de vida do experimento, causadas ou pelo não preenchimento pelo criador do experimento ou ausência de elementos que não podem ser obtidos. Para cada inconsistência, um conjunto de anotações é gerado para auxiliar nas ações que devem ser tomadas para resolver o problema. As anotações geradas como resultado de uma crítica vão estar agrupadas em contextos diferentes ainda em relação ao ciclo de vida do experimento. Essas anotações vão compor o grupo de informações dadas ao pesquisador para que esse tenha completa ciência do estado de seu experimento.

3.2 Modelo de dados

Nesta seção é detalhado o modelo de dados mínimo requerido pelo SEEM para capturar um experimento *in silico* e toda a estrutura utilizada para publicação, edição e execução. Um ambiente que for implementar o SEEM deve ser compatível com esse modelo. Se outro ambiente possuir uma estrutura própria, ele deve fornecer as conversões necessárias para realizar a troca de dados de forma correta. A formalização escolhida para representar o modelo de dados foi a orientação a objetos e diagramas de classe oriundos da UML.

3.2.1 Autenticação

As ações que alteram o estado dos experimentos exigem que o usuário se identifique para que sejam feitos os registros corretamente. Entretanto, o autor pode expor total ou parcialmente os elementos do experimento para acesso público, excluindo a necessidade de autenticação. No modelo de dados aqui proposto, a classe `User` representa um indivíduo previamente cadastrado como utilizador do ambiente. O nosso modelo prevê duas opções para realizar a autenticação de usuário: a autenticação local, que confronta o nome de usuário com uma senha; e o segundo via um provedor de identidade externo, aceito pela implementação do ambiente.

As classes `Role` e `UserRole` servem para configurar quais papéis um usuário possui dentro do sistema, possibilitando que a manutenção do ambiente seja restrita aos administradores e que haja um controle fino de quais ferramentas estarão disponíveis entre os pesquisadores. As permissões de acesso relacionadas aos pontos de acesso do sistema nos permitem separar quais usuários vão poder cadastrar novas ferramentas ou criar novos experimentos. A Figura 16 apresenta um diagrama com ambas as classes responsáveis por esse controle e associa um conjunto de projetos a um usuário, bem como um registro de um conjunto de marcadores de acesso representados pela classe `AccessToken`[1]. Ao utilizar um provedor de identidade externo, o usuário pode associa-lo a um usuário (e senha) existente ou criar um novo usuário. Um mesmo usuário pode se identificar em diversos provedores diferentes.

3.2.2 Projeto

É através de um projeto que todo o experimento é descrito e publicado. O projeto agrupa todos os artefatos, processos, relações e anotações de um experimento de forma a expor toda a sua estrutura para permitir sua posterior reprodução. A Figura 17 apresenta o diagrama da classe `Project` na qual é possível ver que um projeto pertence a um usuário.

[1] Os marcadores de acesso são presentes em padrões e protocolo abertos para autenticação descentralizada de usuários entre diversos sistemas. Veja mais em <http://oauth.net/> e <http://openid.net>

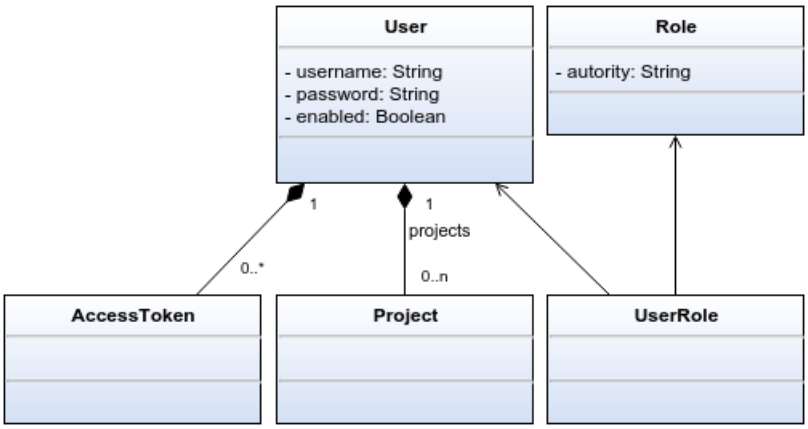
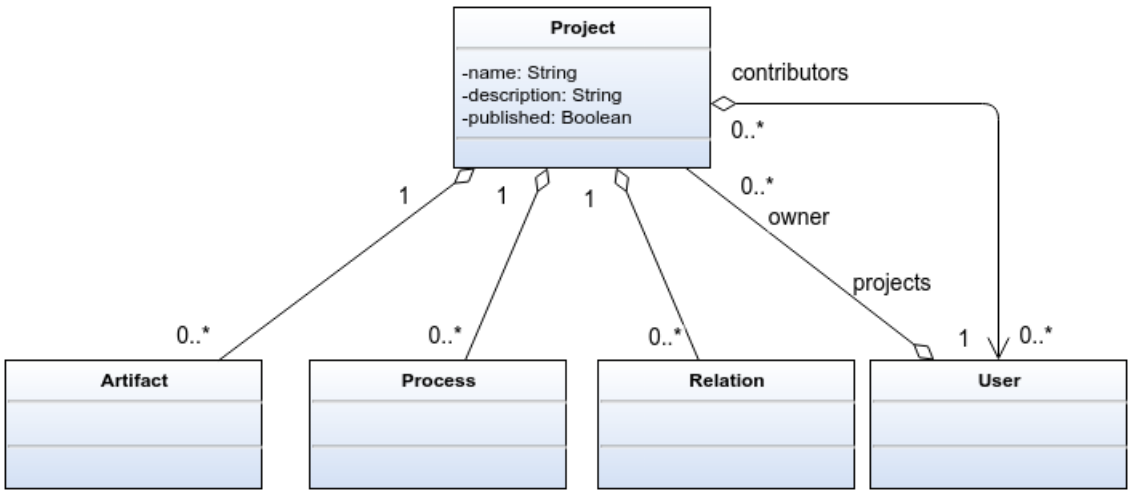


Figura 16 – Diagrama com as classes envolvidas na autenticação e papéis globais dentro do ambiente.

Esse é o dono do experimento e tem responsabilidade total de gestão da criação e descrição do experimento.

Além de estabelecer toda a composição do experimento, o projeto define o grau de acesso externo e qual seu estado atual. Quando o pesquisador julga que o experimento atingiu seus objetivos, ele inicia a publicação do experimento. Um experimento publicado só pode ter suas anotações alteradas. Toda sua composição, artefatos e processos são congelados e os processos não podem mais ser executados. Qualquer alteração deve ser realizada em um novo experimento derivado pois o experimento original publicado será utilizado como referência para posteriores reproduções.



1

Figura 17 – Diagrama de classes para `Project`.

3.2.3 Artefatos

Um dos conceitos centrais para o método é o de artefato. Um artefato é uma abstração que modela um conjunto de dados de interesse para o domínio do experimento. Artefatos são consumidos e gerados durante a execução de um experimento e representam desde dados temporários até o próprio resultado objetivo do experimento. Esse conjunto de dados varia entre tipos primitivos de dados, como números inteiros, números reais, valores lógicos a arquivos binários, como imagens ou arquivos de áudio. Cadeias de caracteres também podem ser vistas como artefatos, assim como arquivos de texto puro contendo dados em algum formato próprio das ferramentas utilizadas ou em formatos padronizados, como os estruturados XML e JavaScript Object Notation (JSON).

Artefatos podem ser persistidos de duas formas diferentes: como arquivos armazenados no sistema de arquivos da instância do ambiente, organizados em pastas próprias para o projeto em que são definidos, ou guardados no banco de dados interno da implementação do ambiente e manipulados durante a execução do experimento.

A Figura 18 apresenta um diagrama contendo a classe principal `Artifact` e suas especializações. As classes `IntegerArtifact`, `FloatArtifact`, `BooleanArtifact`, `StringArtifact` e `BinaryArtifact` representam, respectivamente, quatro artefatos de tipos primitivos e um artefato binário, que podem ser armazenados diretamente no banco de dados do ambiente. Um artefato disponível no sistema de arquivos é modelado através de um `FileArtifact`. Também é possível ver que cada artefato está associado a um único experimento e vai servir de entrada e saída para um processo, que será explicado a seguir.

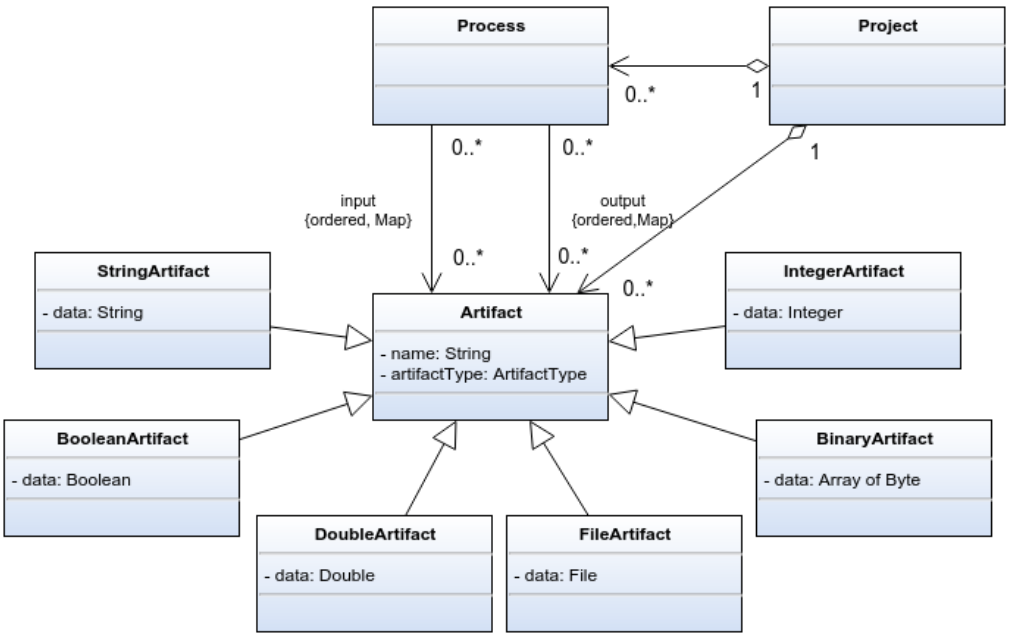


Figura 18 – Diagrama de classes para `Artifact` e suas especializações.

3.2.4 Processo

Um experimento pode ser visto como uma sequência de tarefas que consome um conjunto de artefatos de entrada como insumo (por exemplo, dados obtidos experimentalmente, modelos e parâmetros de configuração) para gerar outros artefatos (por exemplo, análise de dados, gráficos, correlações e filtragem de dados). A essas tarefas é associada uma abstração chamada de "Process". Um Processo modela uma operação, uma transformação ou simplesmente uma geração de artefatos. Essas operações podem ser de curta duração ou levar dias para serem realizadas. Podem ser executadas localmente ou remotamente em uma ou mais estações de computação. É de interesse saber quando um processo foi executado corretamente, por qual método e quais os artefatos foram consumidos e gerados. Entretanto, os detalhes de implementação não são explicitados e assim cada processo age como uma caixa preta.

Um processo é modelado pela classe `Process`. Conforme é possível ver no diagrama da Figura 19, as relações de consumo e produção de artefatos são descritas por dois mapas, um de entrada e um de saída. Os mapas associam uma chave com uma referência a um artefato. Como esses identificadores serão utilizados vai depender da implementação de processo e de como o Administrador de Programas realizou a sua implantação.

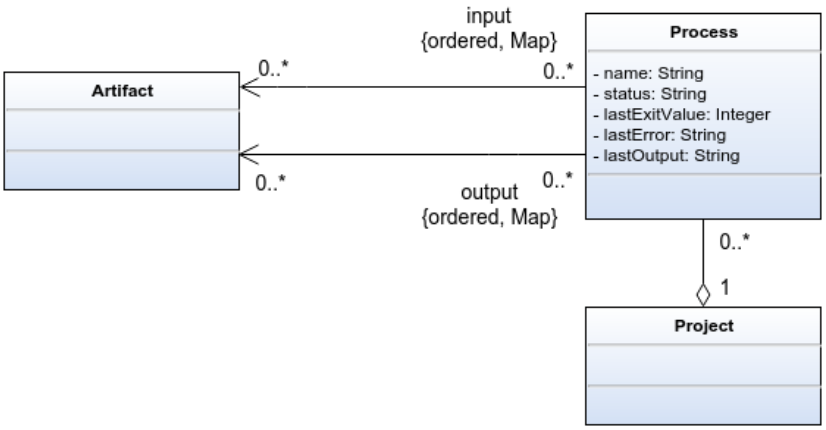


Figura 19 – Diagrama de classes para `Process`.

3.2.5 Programas e *Scripts*

Para que um processo seja executado é necessário que haja uma implementação descrevendo como os artefatos devem ser operados. É necessário, portanto, um *software* na forma de um programa executável ou um trecho de código em modo texto que o próprio ambiente consiga interpretar. Para tanto, a classe `Process` é estendida em subclasses conforme é possível ver no diagrama da Figura 20. A classe `ScriptProcess` representa um processo com acesso às operações do ambiente de simulação e deve ser escrita utilizando uma linguagem de *script* suportada pela implementação do ambiente. A classe

`SystemProcess` modela um processo que executa um programa previamente implantado no ambiente por um administrador. A classe `ManualProcess` representa um processo manual que exige a intervenção de um agente externo para realizar o processamento. Por exemplo, parte de um *workflow* pode exigir que o pesquisador entre com dados colhidos em laboratório ou ferramentas que não podem ser implantadas ao ambiente.

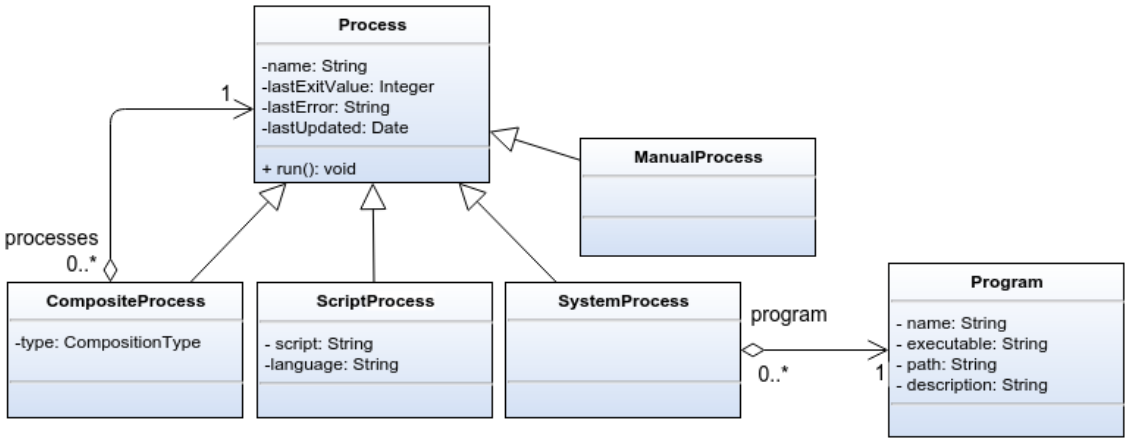


Figura 20 – Diagrama de classes para `Program` e as especializações de `Process`.

A utilização de processos regidos por *scripts* facilita a implementação e reprodução por parte do ambiente por envolver apenas a transmissão do texto entre instâncias do ambiente. Entretanto, não é possível assumir que pesquisadores vão deixar de desenvolver seus programas em diversas linguagens e tecnologias diferentes. Há, então, a necessidade do perfil do Administrador de Programas para, junto ao pesquisador, realizar a implantação e acompanhamento de novos programas, lançando uso de *wrappers* sempre que necessário.

3.2.6 Composição

O SEEM permite que a execução dos experimentos seja guiada completamente por ações do usuário. Adicionalmente, fornece um conjunto de construtores básicos para automatizar total ou em parte a execução dos processos. A execução automática ou guiada pelo usuário de processos de um *workflow* é objeto de estudo em diversas abordagens (OGASAWARA, 2011; MATTOSO et al., 2013; KAIL; KACSUK; KOZLOVSZKY, 2015). Os construtores do SEEM abstraem o modelo de execução permitindo que diversas abordagens sejam implantadas posteriormente.

Como um de seus construtores básicos para a execução, a classe `CompositeProcess` permite agrupar outros processos existentes em processos mais complexos. Ela representa um grupo de processos no qual a forma de execução dependerá do valor de seu parâmetro `CompositionType`.

Se configurado para realizar uma sequência simples, cada um de seus subprocessos será executado sequencialmente. Um processo só será executado após a conclusão do pro-

cesso anterior e o processo composto só é considerado bem sucedido se todos os processos internos tiverem sua execução finalizada com sucesso. Já na execução em paralelo, todos os processos são disparados simultaneamente e de forma assíncrona. O processo composto só finaliza se os processos internos terminam com uma execução bem sucedida.

Duas classes de processos compostos permitem definir controles de execução do *workflow*. Um processo condicional recebe dois processos internos para execução e decide qual dos dois irá executar com base em um artefato lógico consumido como entrada. Apenas um dos dois processos será executado de acordo com seu valor. Uma repetição pode ser implementada ao se utilizar um processo de repetição. Novamente ele recebe um artefato lógico que informa se o processo interno será ou não executado. É importante destacar que esse artefato deve ser alterado internamente para que a sua execução não fique presa em um laço infinito. Portanto é obrigatório que na composição seja fornecido um artefato de guarda e um processo que atualize este artefato.

3.2.7 Relacionamentos

Ao se descrever um experimento pode haver a necessidade de indicar um relacionamento entre artefatos diferente do relacionamento de consumo e produção pelos processos. A classe `Relation` modela uma relação conceitual, específica do domínio do experimento, entre dois artefatos. Como exemplo é possível citar arquivos de configuração associados a algum modelo específico ou modelos compostos por vários arquivos. No nível conceitual do *Simulation Experiment Environment Method*, as relações são apenas distinguíveis por uma identificação, tipicamente uma URI que pode ser usada para dar significado ao tipo de relacionamento dentro de um domínio. A Figura 21 mostra um diagrama no qual é possível definir esse tipo de relacionamento, direcional ou não, entre dois artefatos. Vale destacar que há a possibilidade de se definir um auto-relacionamento em um artefato se necessário.

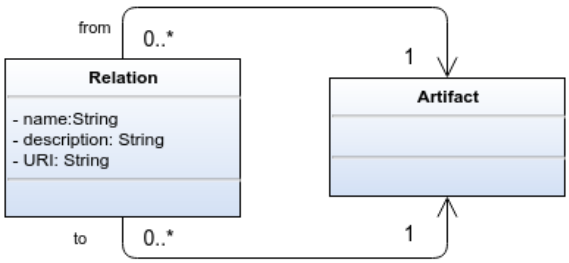


Figura 21 – Diagrama de classes para `Relation` que modela uma relação conceitual entre dois artefatos.

3.2.8 Disponibilidade

Alguns comportamentos são compartilhados entre classes para realizar o controle da execução e publicação dos experimentos e de suas partes componentes. Uma primeira operação que é necessária identificar é se as partes componentes estão prontas para uso ou apenas disponíveis no ambiente. Como dentro do ciclo de vida do experimento os componentes podem estar incompletos, seja pelo componente não ter sido ainda enviado pelo pesquisador, por não poder ser obtido ou estar indisponível por algum outro motivo, ele tem que ser indicado como não disponível.

Para identificar se um experimento está "pronto" para execução, é utilizada a interface `Readiness` que traz o método `isReady()` que irá retornar um valor lógico indicando o seu estado, conforme é observado na Figura 22. Um artefato será considerado pronto se tiver seu conteúdo disponível para uso imediato. Os artefatos de tipos primitivos estarão prontos se seus dados forem diferentes de nulo dentro do banco de dados do ambiente. Já os artefatos que representam arquivos só estarão prontos se fornecerem um caminho válido para um arquivo existente no sistema de arquivos do ambiente. Um processo, por sua vez, estará pronto quando todos os seus artefatos de entrada e seu programa estiverem prontos.

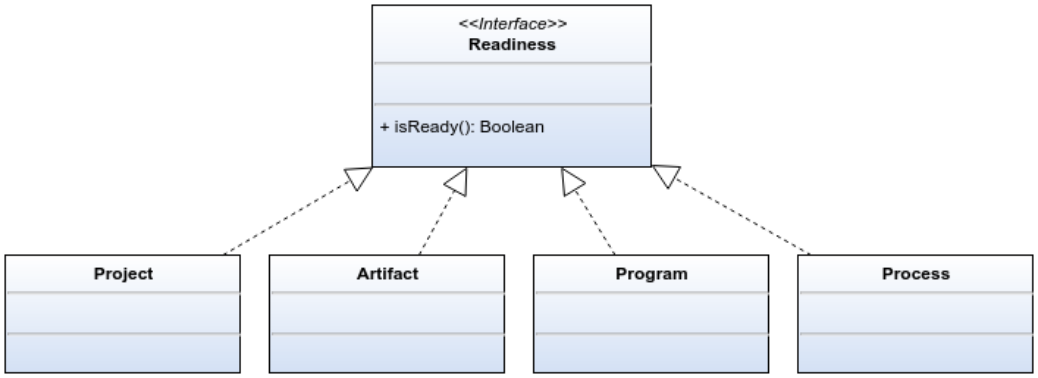


Figura 22 – Diagrama de classes para a interface `Readiness` que indica se a instância está pronta para uso dentro do experimento.

3.2.9 Anotações

Um ponto fundamental do SEEM é fornecer ao pesquisador uma forma de criar, armazenar e recuperar anotações para aumentar a descrição dos experimentos que ele cria. Uma anotação pode indicar desde comentários informais criados pelo autor do experimento até descrições formais geradas automaticamente por um *software* que realizará uma crítica sobre o estado atual. A Figura 23 mostra a classe `Annotation` e sua relação com os demais elementos do experimento.

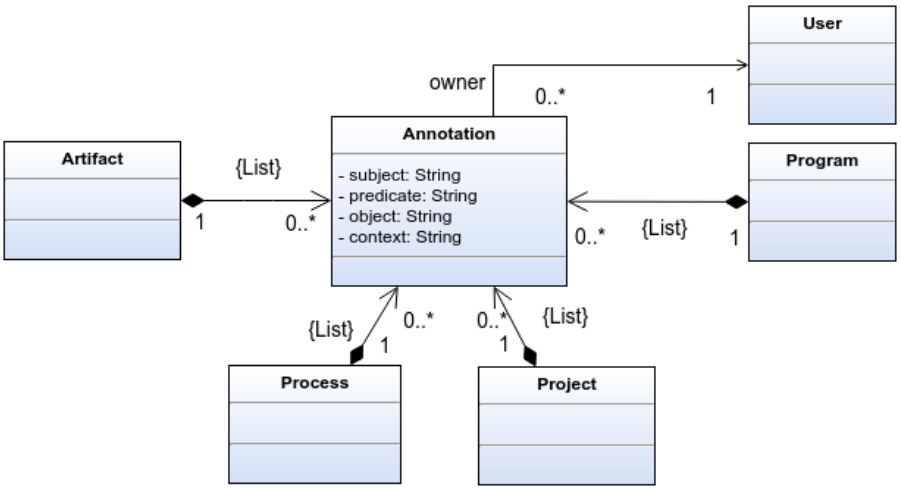


Figura 23 – Diagrama de classes da classe `Annotation` e as classes que podem ser anotadas.

Para compor uma anotação será utilizada a definição sugerida por Oren et al. (2006): uma anotação terá quatro propriedades fundamentais: o sujeito, alvo da anotação; o objeto, os dados anotados ao sujeito; o predicado, que é a relação entre o sujeito e objeto; e o contexto, que é o domínio em que o predicado está inserido. Cada um dos campos é um bloco de texto simples, permitindo que seja fornecido desde uma anotação informal até um texto formal que possa ser lido e interpretado pela máquina. Uma anotação formal, inclusive, pode ser uma URI para uma ontologia.

As anotações podem ser manipuladas diretamente através de uma interface com o pesquisador ou geradas automaticamente pelo sistema durante a crítica dos experimentos. Durante o projeto da interface com o usuário, algumas anotações podem ser exibidas ou não. Essa decisão fica a cargo da implementação e, de preferência, de acordo com o contexto atual. Por exemplo, uma anotação informal criada pelo autor não será relevante durante a execução do experimento e nem para a recuperação do mesmo. Uma anotação derivada de uma crítica pode não ser relevante dentro do contexto de exibição dos resultados do experimento. Portanto, deve ser possível realizar uma filtragem das anotações em diferentes fases da criação, documentação, recuperação e execução dos experimentos.

A Figura 24 apresenta um diagrama no qual as anotações referentes às críticas são organizadas em cinco contextos diferentes. O contexto de "Criação" agrupa as anotações referentes a autoria e identificação do experimento. Já o "Estrutura", está associado aos inter-relacionamentos estruturais dos elementos do experimento. No contexto de "Execução" aparecem as anotações criadas como resposta à execução do experimento dentro do *Sistema Gerenciador de* Workflow (SGWf). Já a derivação do experimento em novos fica dentro do contexto de "Reuso". E por fim, o contexto de "Importação Remota" trata das anotações de derivação mas entre instância diferentes do ambiente.

Figura 24 – Diagrama de anotações com uma divisão em ramos por contexto utilizado.

3.3 Criação de Experimentos

O primeiro passo que um pesquisador deve tomar para publicar seu experimento no ambiente é criar um novo projeto e prover algumas informações básicas como um nome identificador e uma breve descrição. O pesquisador pode criar seu experimento no ambiente usando um que ele já possua em seu computador. Também é possível que deseje investigar algum experimento já existente e publicado na plataforma a fim de obter algum novo resultado. Do ponto de vista do ambiente, um novo experimento surge de duas formas distintas: um experimento completamente novo ou uma nova versão de algum experimento já existente no ambiente.

3.3.1 Implantação de Programas

Para o caso de se gerar um novo experimento, além do envio da descrição de todo o fluxo de trabalho, é necessário realizar a implantação dos programas que serão utilizados pelos processos. O aceite e os detalhes de implantação desses programas devem ser tomados em conjunto com o Administrador de Programas, pois além de questões de segurança há a compatibilidade de versões e dependências de *software* a serem mantidas. Após a implantação dos programas, outros experimentos que utilizem esse mesmo ferramental podem ser criados sem a necessidade de se contactar os administradores.

3.3.2 Importação de Experimentos

Duas instâncias do ambiente podem trocar experimentos entre elas através da rede de forma automática ou semi-automática. Se um pesquisador solicitar um experimento de uma outra instância do ambiente, todos os artefatos e processos são transferidos e seus programas necessários agendados automaticamente para implantação pelo administrador. Caso não seja possível obter automaticamente os programas para os processos, o

Tabela 1 – Tabela com dados criticados pelo SEEM durante a criação de um experimento novo.

Dado	Caso Ausente	Contexto
Identificador único	Impedir a criação	Base
Descrição Resumida do Experimento	Sugerir o preenchimento	
Descrição Estendida do Experimento	Sugerir o preenchimento	
Autoria Principal	Associar ao usuário atual	Base
Autoria Secundária	Associar ao usuário atual	Base

ambiente deve marcar o experimento como não reprodutível e sugerir a sua instalação ou substituição.

3.3.3 Críticas

Durante a etapa de criação de um experimento, algumas críticas devem ser realizadas de imediato. Um identificador para o experimento deve ser fornecido para auxiliar a todos os usuários a identificar e recuperar o experimento. Um identificador único deve ser gerado imediatamente e esse deve ser único para referenciar o experimento. Uma descrição resumida pode ser fornecida nesse momento, mas de caráter opcional. Caso ausente, uma anotação é atribuída ao experimento para que o usuário a adicione posteriormente. A descrição estendida também é sugerida e segue a mesma ideia da resumida, adicionando uma anotação no contexto de identificação Base.

3.3.4 Autoria e Propriedade

A propriedade do experimento é vinculada obrigatoriamente e automaticamente ao usuário atual, criador do experimento. Fica a cargo do pesquisador o papel de dono do experimento e todo o poder de gestão de permissões de acesso. Também é possível associar ao experimento mais pesquisadores como colaboradores dele. O dono do experimento poderá atribuir-lhes direitos de administração e gestão. A Tabela 1 resume as anotações que podem ser criadas automaticamente durante a etapa de criação de um novo experimento.

3.4 Publicação de Experimentos

Um ponto central do SEEM é o ato de publicação do experimento. Apesar de um experimento possuir vários estados durante seu ciclo de vida, há um interesse especial no estado em que o autor define que o mesmo está concluído, que atingiu os objetivos planejados pelo experimentador para ser útil em trabalhos futuros. Ao atingir esse estado, é dito que o experimento foi publicado. A partir da publicação de um experimento, o autor permite que outros pesquisadores possam repeti-lo ou reproduzi-lo em novos experimentos.

A Figura 25 apresenta um diagrama de estados no qual é possível ver um ciclo de vida de um experimento dentro do SEEM.

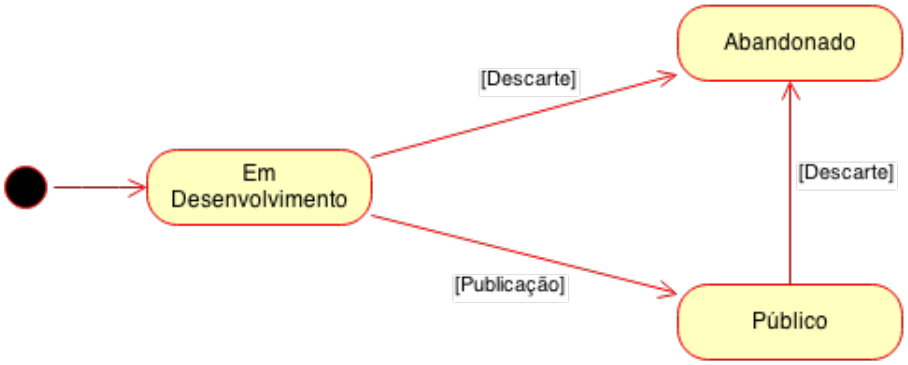


Figura 25 – Diagrama de estados dos experimentos no SEEM.

Um projeto publicado tem seu estado congelado para servir como um documento do experimento que ele representa. Um experimento publicado pode ser usado como base para o desenvolvimento de novos experimentos. Portanto, há a necessidade que o projeto publicado seja imutável. Se o dono do experimento desejar realizar quaisquer modificações após esse ponto, ele deverá criar um novo ciclo de desenvolvimento a partir do experimento original.

Cada experimento publicado deve expor seu *workflow* para os demais pesquisadores na forma de processos e artefatos. Todos os seus elementos componentes devem estar descritos como parte da composição e estar devidamente relacionados com quaisquer recursos necessários para a sua execução. Entretanto, algum elemento componente pode estar ausente ou não acessível. Portanto, um experimento pode possuir partes ausentes, mas essas devem ser descritas explicitamente. Isso conduz a um experimento não reproduzível, mas essa condição deverá ficar clara para qualquer outro pesquisador. A Tabela 2 apresenta um conjunto de anotações que podem ser geradas durante a publicação de um experimento.

3.4.1 Críticas

Como toda a estrutura do experimento fica descrita dentro do ambiente, o sistema é capaz de realizar uma crítica sobre seus componentes e responder quando um experimento pode ser reproduzido ou não. A nossa abstração do problema considera que um experimento é formado por um conjunto de processos que consomem e geram artefatos. Assim, uma primeira crítica que pode ser realizada sobre o experimento é se todos os artefatos estão disponíveis para os processos que os consomem. Adicionalmente, cada processo exige que o ambiente conheça uma implementação na forma de um programa executável para operar os artefatos disponíveis. Esses programas também devem estar disponíveis e funcionais dentro do sistema operacional no qual o ambiente estiver em execução. Se

Tabela 2 – Tabela com dados criticados pelo SEEM durante a publicação de um experimento.

Dado	Caso Ausente	Contexto
Artefato Isolado não consumido ou gerado	Maior descrição necessária	Publicação
Artefato Consumido Ausente	Repetição Impossibilitada por falta de dados de entrada	Publicação
Artefato Produzido Ausente	Repetição Comprometida por falta de dados de saída	Publicação
Processo Ausente	Repetição Comprometida por falta de detalhes de execução	Publicação
Implementação de Processo Ausente	Repetição Comprometida	Publicação
Implementação de Processo Não Funcional	Repetição Comprometida	Publicação
Implementação de Processo em versão diferente	Repetição Comprometida	Publicação

algum programa não puder ser executado, todos os experimentos que possuam processos vinculados a ele estão comprometidos para repetição.

Uma primeira análise a respeito do grafo do experimento deve ser realizada sobre os artefatos e suas ligações aos processos. Os artefatos que são consumidos por processos devem estar disponíveis para consumo. Se por algum motivo seus dados se perderam antes da publicação, o artefato é anotado como ausente e o projeto com uma anotação de que sua repetição está prejudicada pela falta de dados de entrada. Por exemplo, um arquivo do disco pode ter sido apagado acidentalmente ou os dados são sigilosos e não podem ser publicados.

Com relação aos artefatos produzidos pela execução de processos, se seus dados não estiverem disponíveis, também significa que a reprodução está prejudicada, pois não será possível comparar os resultados de outro experimento com o de referência. O artefato ausente é então anotado como ausente e o experimento como não podendo ser repetido por não ter dados de saída explicitados.

Se um artefato estiver isolado, ou seja, não for utilizado como entrada nem como saída por algum processo, ele deve ser anotado com essa informação. Essa informação pode exigir que seja fornecida uma melhor descrição para a existência do artefato. Anotações como essas devem ter um significado único dentro do contexto e devem ser criadas formalmente.

Se um processo se torna indisponível durante o uso contínuo do ambiente ou não há uma descrição de sua implementação, o experimento é anotado com repetição impossibilitada, pois os resultados não podem ser obtidos posteriormente. Uma anotação de advertência é também adicionada ao processo para enfatizar que o pesquisador deve

fornecê-lo.

Mesmo quando presente, a implementação dos processos pode impedir a repetição ou reprodução de um experimento. O primeiro caso é que o *software* não funciona, seja por alguma atualização externa ao ambiente ou uma implantação problemática durante uma importação. Se uma implantação é detectada como não pronta para uso, o seu processo é anotado como problemático e o experimento como não apto à repetição. Adicionalmente, todos os demais experimentos que utilizam essa implementação dentro do ambiente também devem ser anotados com problemas de repetição. Portanto, o administrador deve ser contactado para apurar o ocorrido ou marcar todos os experimentos como não reprodutíveis indefinidamente.

Há o caso em que o *software* está presente, mas a versão que foi implantada pelo administrador difere da atualmente disponível no ambiente. Novamente há uma série de anotações em cascata para informar que houve uma alteração nos experimentos que utilizavam o *software* em seus processos. Os experimentos devem ser marcados como não sendo possíveis de repetição mas, possivelmente, passíveis de reprodução. Novos experimentos devem ser criados para acomodar a nova versão do *software*. A identificação da versão e a busca por alterações são de responsabilidade do administrador encarregado pela implantação de novos programas no sistema. A verificação de atualizações não planejadas pode ser realizada periodicamente e automatizada nos casos em que os programas consigam informar suas versões por linha de comando.

3.5 Repetição de Experimentos

O ambiente permite a repetição de experimentos localmente ou em uma outra instância do ambiente através de uma cópia completa do projeto. A repetição de experimentos local ou remota é realizada de forma que todo o fluxo de trabalho seja idêntico ao experimento original, objetivando obter os mesmos resultados. O fluxo de trabalho e todos os artefatos devem ser copiados de forma automática. Caso não seja possível realizar a cópia automaticamente, deve-se sugerir a instalação manual das funcionalidades necessárias. Entretanto, o sistema deverá atestar a integridade do *workflow* do experimento para repetição.

Quando um artefato, processo ou programa não estiver disponível, completa ou parcialmente, um elemento falso é gerado para ocupar o espaço do elemento ausente. Esse elemento falso é preenchido com o máximo de informação que puder ser obtida do *workflow* original. Esse elemento falso deve ser anotado para que seja fornecido posteriormente ou substituído por um equivalente. Sempre que houver a necessidade de intervenção manual, uma anotação deve ser gerada para indicar que os administradores do ambiente precisam ser contactados. Sempre que não for possível obter todos os componentes do experimento,

o seu projeto é anotado como incompleto, passível de reprodução, mas impossibilitado de ser repetido dentro da instância do ambiente atual.

3.6 Reprodução de Experimentos

A reprodução de um experimento significa obter os mesmos resultados quando algum elemento do fluxo de trabalho foi substituído por outro equivalente. Um pesquisador que obtém um experimento existente anotado por uma implementação do SEEM deve ter em mãos todos os passos e modelos necessários para sua reprodução. Se algum artefato, processo ou *software* estiver ausente, eles devem constar em uma lista de tarefas a serem resolvidas pelo pesquisador.

O ambiente deve anotar o experimento com uma referência para o outro experimento que ele está tentando reproduzir. Através dessa anotação de referência, o ambiente deve ser capaz de verificar quais modificações foram realizadas sobre cada elemento do *workflow*. Os artefatos devem ser idênticos e um processo de verificação por *checksum* pode ser aplicado para compará-los. Em caso de divergência, uma anotação de advertência deve ser gerada para informar ao pesquisador que o experimento estará conduzindo, possivelmente, a um resultado diferente. Se um processo utiliza uma implementação diferente, uma anotação no processo deve alertar para a situação.

Artefatos de saída podem ser anotados como utilizados para validar o resultado do experimento com o de um experimento existente. Além da verificação por *checksum*m outro protocolo de equivalência pode ser empregado ou até mesmo uma validação manual dos resultados. Entretanto, o tipo específico de validação deve ser descrito pelo experimentalista; a ausência dessa informação deve gerar uma nova anotação de advertência.

A Tabela 3 apresenta um resumo das anotações geradas durante a crítica quanto à reprodução de um experimento. Algumas das execuções de críticas podem exigir um consumo de uma série de recursos pela implementação do sistema (como por exemplo, comparar arquivos muito grandes ou que devam ser trafegados pela rede). Então, parte das ações podem não ser realizadas automaticamente e vão exigir uma ação manual por parte do administrador ou depender de algum agendamento de processamento.

3.6.1 Federação e procedência

Os experimentos publicados devem estar disponíveis remotamente para que outros pesquisadores os identifiquem e os obtenham. Portanto, cada experimento possui um identificador único, gerado internamente no ambiente e exposto externamente na forma de uma Uniform Resource Identifier (URI). Isso permite que os experimentos estejam disponíveis externamente ao ambiente, podendo ser recuperados através da rede. Internamente ao experimento, cada um dos seus elementos também deve ser acessível externamente através

Tabela 3 – Tabela com dados criticados pelo SEEM durante a reprodução de um experimento.

Dado	Caso Ausente	Contexto
Artefato do experimento original ausente no experimento atual	Reprodução comprometida	Reprodução
Artefato do experimento original diferente do experimento atual	Resultados comprometidos	Reprodução
Implementação de Processo Ausente	Reprodução impossibilitada	Reprodução
Implementação de Processo Diferente	Reprodução possível	Reprodução
Artefato de saída difere do original	Reprodução falha	Reprodução
Experimento original não disponível	Reprodução impossibilitada	Reprodução

de um identificador próprio para obtenção e anotação de seus metadados. Isso permite reaproveitar o experimento na totalidade ou apenas parte dos elementos componentes ao acrescentar anotações de forma independente do projeto atual.

O método prescreve que os elementos do modelo sejam federados, ou seja, possam ser acessados externamente à instância da implementação. Portanto todos os elementos são identificados por uma URI para acesso externo e, internamente, podem ter um número de identificação local ou gerado por um Universally Unique Identifier (UUID). Assim todos os projetos, experimentos, programas, processos, artefatos e anotações podem ser selecionados pelo seu UUID dentro do servidor.

Através de um UUID é possível obter acesso aos dados de um experimento ou qualquer de suas partes constituintes. Entretanto, uma camada de segurança deverá bloquear acesso a elementos ainda não publicados ou definidos explicitamente como sendo de acesso restrito. A Figura 26 apresenta a interface `Addressable` que define o comportamento para os objetos que devem ser identificados no ambiente de experimentos através de uma URI.

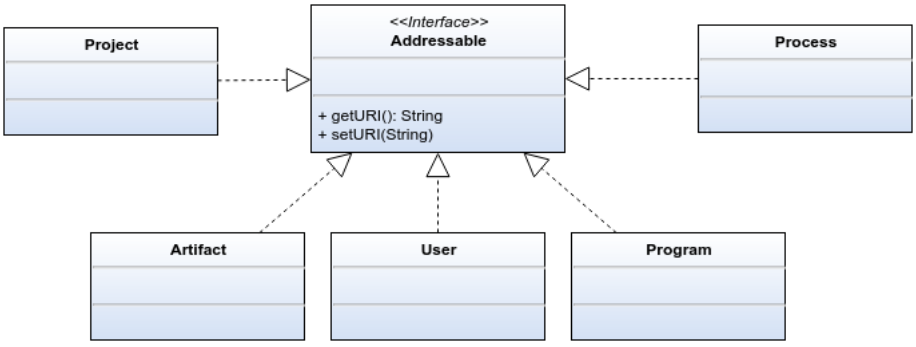


Figura 26 – Diagrama de classes da interface `Addressable`.

Ao se derivar um novo experimento a partir de um outro já existente, tem-se o interesse de manter a procedência direta para os elementos que os originaram. A Figura 27 apresenta a interface `Derived` que modela o comportamento necessário para se obter o identificador do objeto que deu origem ao objeto atual. Destaca-se que isso não é uma forma de versionamento interna ao experimento. É um relacionamento que permite comparar um experimento publicado com um outro em desenvolvimento ou também publicado.

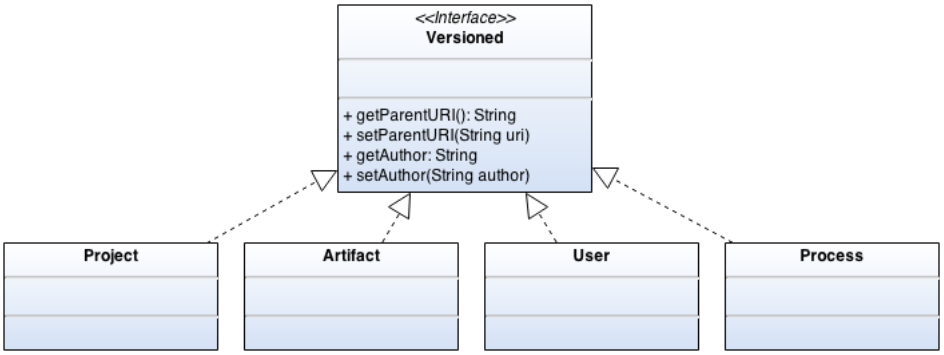


Figura 27 – Diagrama de classes da interface `Versioned`.

3.7 Arquitetura conceitual

Um cenário de aplicação para o SEEM pode ser idealizado a partir de um grupo de pesquisa que tem o interesse em registrar e disponibilizar experimentos computacionais, em diversos domínios e áreas de aplicação, com pesquisadores internos e externos. Espera-se que ao compartilhar formalmente o fluxo de trabalho completo junto aos resultados gerados, se acelere a produção científica e diminua o retrabalho.

Paralelamente à pura disponibilização, o ambiente deve manter um relatório dinâmico do estado dos elementos constituintes do experimento a fim de expor o seu estado geral de reprodutibilidade atual. Um pesquisador com acesso a um experimento deve conseguir visualizar exatamente quais elementos estão funcionais. Se parte do experimento estiver inacessível ou com problemas, deve estar claro quais são os elementos que impedem a sua reprodução.

Para dar suporte ao cenário proposto, uma arquitetura mínima exige uma série de componentes de *software* para ser viável. Essa arquitetura pode ser observada na Figura 28. O modelo de dados proposto anteriormente consegue capturar todo o fluxo de trabalho e o conjunto de anotações realizado manualmente pelos pesquisadores ou automaticamente por um processo de crítica e curadoria.

Para realizar toda a criação e compartilhamento dos experimentos foi assumido desde o princípio que o ambiente deve estar baseado em uma aplicação *web*. Essa restrição

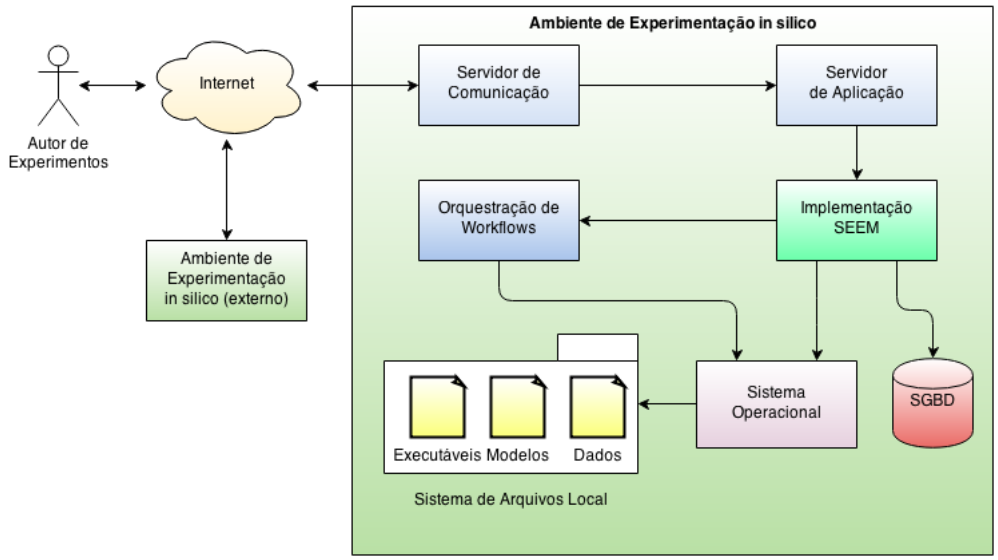


Figura 28 – Arquitetura conceitual para um ambiente computacional para dar suporte ao SEEM.

se justifica pela própria natureza colaborativa da *Internet* e da pesquisa, bem como a inquestionável aproximação da computação com a rede. Portanto, o primeiro componente da arquitetura tem que fornecer um servidor de comunicação para tratar as requisições dos usuários através da rede. É desejável que este servidor possua uma arquitetura aberta e utilize protocolos gratuitos de comunicação para maior integração com outras abordagens e ferramentas.

Um servidor de aplicação [2] é o componente responsável por executar a lógica de implementação do SEEM. Não há qualquer restrição em se utilizar uma solução proprietária, pois as requisições serão tratadas internamente ao servidor e uma interface de comunicação comum isola essa dependência.

O componente para persistência de dados deve estar definido e a abordagem utiliza um *Sistema Gerenciador de Banco de Dados* (SGBD) para tal. Toda a estrutura do *workflow*, alguns artefatos e metadados gerados pela aplicação serão armazenados no SGBD. Adicionalmente, todo o controle de acesso também é armazenado em banco de dados, mesmo que provedores externos sejam utilizados. Não há restrição para a escolha do paradigma de persistência e essa decisão fica a critério do desenvolvedor que irá implementá-lo junto ao componente do servidor de aplicação.

Para realizar a composição concreta, o ambiente irá necessitar de um sistema de gerenciamento de *workflows*. É através dele que o fluxo de trabalho é construído e encaminhado para a execução. O sistema de gerenciamento de *workflows* se encarrega que os processos sejam executados de forma correta e que a geração do fluxo de dados cumpra

[2] Um software que atua como servidor, provendo uma infraestrutura para execução de aplicativos, integridade de dados, segurança e controle de transações.

com os objetivos do experimento.

Servindo de base para toda a arquitetura, o sistema operacional é utilizado para o acesso ao sistema de arquivos e provê o ambiente no qual os programas dos processos serão executados. A dependência do sistema operacional irá impor restrições aos programas compilados para código nativo. Isso implica que instâncias diferentes de ambientes diferentes podem não ser capazes de repetir ou reproduzir todos os experimentos. Entretanto, devem ser capazes de informar essa restrição e indicar quais elementos devem ser substituídos para realizar a reprodução.

3.8 Considerações parciais sobre o SEEM

Este capítulo apresentou um método conceitual, o SEEM, que dá diretrizes de como descrever experimentos computacionais em função de seu fluxo de trabalho. Ao seguir as diretrizes, um pesquisador descreve como as ferramentas computacionais e artefatos utilizados foram arranjadas de maneira lógica para cumprir os objetivos do experimento. De posse dessa descrição, críticas podem ser aplicadas e os componentes do fluxo de trabalho são anotados em busca de problemas que podem dificultar ou impedir a posterior reprodução do experimento. O método prevê o amparo de um ambiente computacional e uma arquitetura conceitual foi proposta ao final do capítulo. Este ambiente prevê a integração e execução dos programas utilizados nos experimentos para que a coleta das anotações seja feita de maneira automática ou semi-automática, permitindo que relatórios sejam gerados sobre o estado da reprodutibilidade dos experimentos.

No próximo capítulo é apresentada uma implementação concreta, desenvolvida como prova de conceito, de um ambiente computacional que segue as diretrizes do SEEM.

4 O Ambiente SEE

Este capítulo apresenta a infraestrutura computacional criada como prova de conceito para o SEEM, o SEE, no qual é possível realizar a criação, execução e compartilhamento de experimentos científicos *in silico*. É apresentada a arquitetura computacional utilizada, bem como seu relacionamento com o método. O ambiente também serve de repositório de experimentos e realiza a crítica automática e semi-automática durante diversas fases do ciclo de vida do experimento. Desta forma, tanto os dados quanto ao estado de sua reprodução são colhidos e utilizados para relatar o estado das informações fornecidas pelo pesquisador.

4.1 Arquitetura do Ambiente de Experimentação

O protótipo construído como prova de conceito para esta Tese fornece os recursos necessários para a captura do ciclo de vida de um experimento *ad hoc* completo, mas dá especial enfoque a quatro fases: criação, modificação, publicação e reuso. A Figura 29, adaptada da Figura 13 vista anteriormente, destaca as fases nas quais se concentram os esforços deste trabalho.

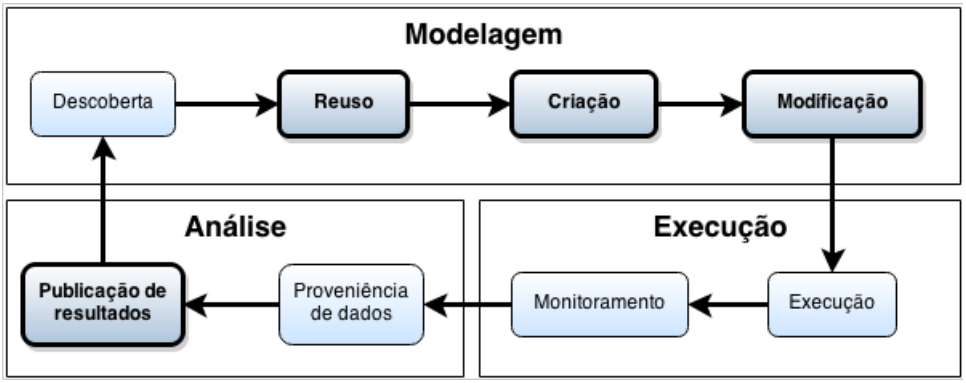


Figura 29 – Ciclo de vida de um experimento *in silico*. Destaque para os pontos os quais este trabalho apresenta contribuições.

Para dar suporte às fases de interesse do ciclo de vida do experimento, foi implementada uma solução concreta baseada na arquitetura conceitual vista previamente na Figura 30 da Seção 3.7. A implementação do ambiente utiliza uma pilha de tecnologias abertas na sua infraestrutura para realização da criação, execução, anotação, publicação e reprodução dos experimentos. Entretanto, é permitido que cada experimento tenha componentes de *software* proprietários ou código fechado em sua composição, apesar de isso influenciar diretamente em sua reprodutibilidade.

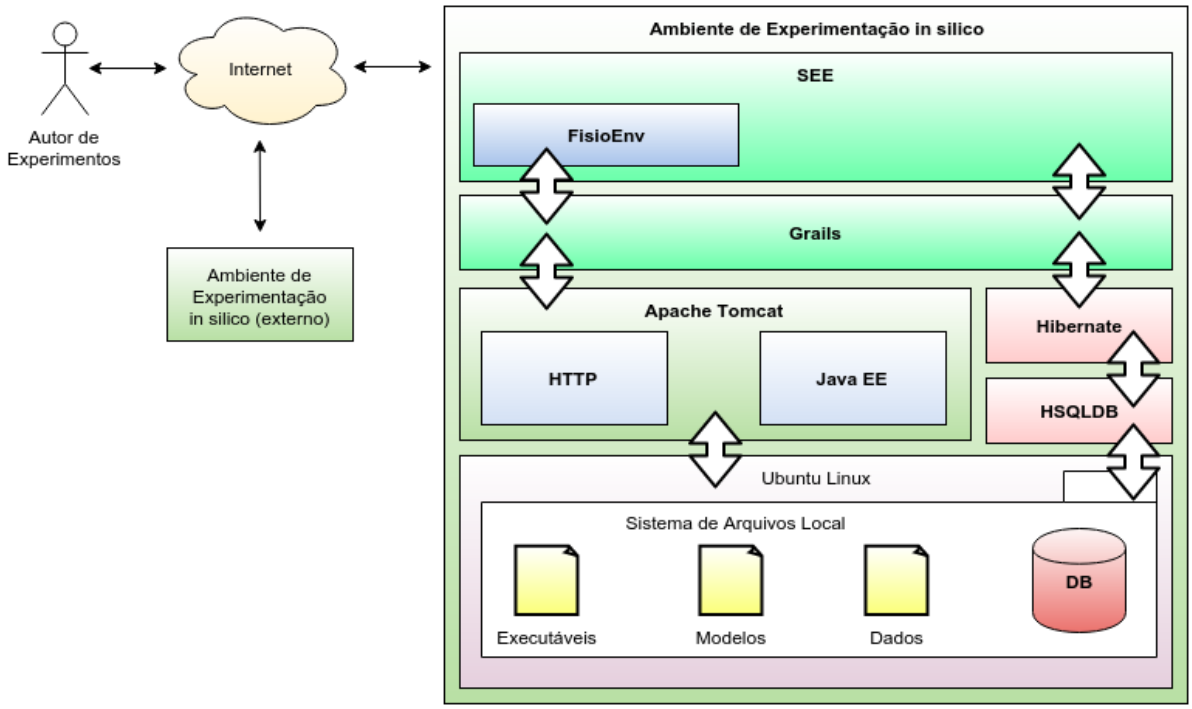


Figura 30 – Infraestrutura para o protótipo do ambiente de experimentação SEE.

Na implementação de protótipo SEE foi utilizado um servidor HyperText Transfer Protocol (HTTP) através do servidor Apache Tomcat (APACHE SOFTWARE FOUNDATION, 2016). A interface com o usuário é uma aplicação *web* acessada via navegador. Porém, isso não é uma restrição do método e outra implementação pode criar seu próprio meio de interação com o usuário.

Como servidor de aplicação, o protótipo SEE utilizou novamente o Apache Tomcat, através do arcabouço *web* Grails (SMITH; LEDBROOK, 2009; GRAILS PROJECT, 2016) sobre a linguagem Groovy (GROOVY PROJECT, 2016). Esta aplicação gerencia todos os demais módulos.

O protótipo utiliza o HyperSQL DataBase (HSQLDB) (HSQL DEVELOPMENT GROUP, 2016) através do *framework* Hibernate (BAUER; KING, 2006; HIBERNATE COMMUNITY, 2016) e exposto externamente via Grails.

Como SGWf foi utilizado o FISIOENV (KNOP et al., 2012a) como infraestrutura capaz de descrever e executar *workflows*. Ele também serviu de base para a construção da aplicação *web* que serve como interface com o usuário e ambiente de concepção de experimentos.

O protótipo foi construído sobre uma distribuição GNU/Linux (GNU/Linux) Ubuntu e foram integradas em nosso ambiente as ferramentas de desenvolvimento em C e C++ como o GNU C Compiler (GCC) e GNUplot (WILLIAMS; KELLEY et al., 2010). Visando a integração com sistemas existentes, todas as funcionalidades para criação e ma-

nutenção dos *workflows* são exportadas como serviços *web* via interfaces Simple Object Access Protocol (SOAP) (BOX et al., 2000) e Representational State Transfer (REST) (RICHARDSON; RUBY, 2008). Atualmente, apenas os serviços REST são utilizados para troca de experimentos entre instâncias diferentes do ambiente.

4.2 Definição dos Artefatos e Processos

O FISIOENV permite a concepção de novos experimentos utilizando o modelo conceitual visto na Seção 3.2, no qual as ferramentas são encapsuladas como processos da classe *Process* e os modelos e arquivos de dados digitais são representados por artefatos da classe *Artifact*. Com essa estrutura podemos construir fluxos de trabalho complexos utilizando os artefatos de saída de um processo como artefatos de entrada em outro processo. Toda essa composição pode ser realizada utilizando uma interface *web* do FISIOENV. A Figura 31 apresenta um exemplo de um processo simplificado. São utilizados três artefatos como entrada e são gerados dois artefatos de saída após uma execução bem sucedida.

Figura 31 – Desenvolvimento de um processo simplificado utilizando o FISIOENV: três artefatos de entrada e dois de saída.

O FISIOENV também gera um diagrama que representa o fluxo de trabalho e as conexões entre todas as entradas e saídas de processos utilizando internamente a linguagem *dot* do projeto Graphviz (AT&T, 2016). A Figura 32 apresenta o diagrama, associado com o processo descrito na Figura 31, na qual o processo é apresentado como um paralelepípedo verde e três tipos de artefatos de entrada são representados graficamente: um arquivo no sistema de arquivos como uma folha de papel; um dado numérico real como um círculo; e um número inteiro como um retângulo. Os dois artefatos de saída são também representados como folhas de papel.

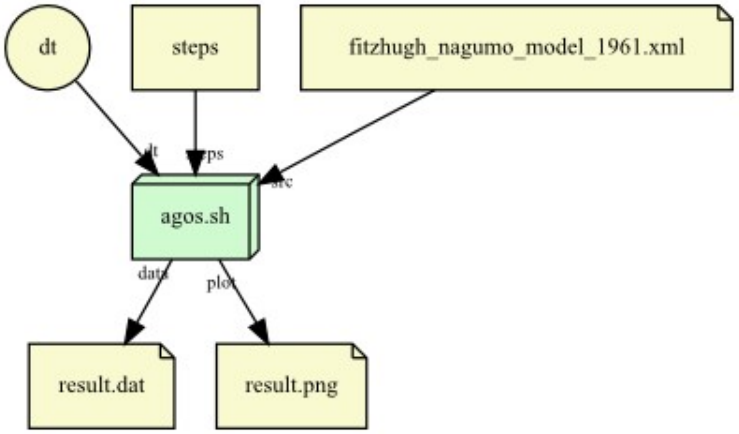


Figura 32 – Diagrama de um processo dentro do FISIOENV: três artefatos, um arquivo em disco, um número real e um número inteiro são utilizados de entrada para um processo que gera dois artefatos de arquivo no disco dos sistema de arquivos.

4.3 Composição e Execução dos Processos

Cada processo é criado e executado de forma isolada dentro do ambiente. Os dados sobre a execução de cada processo (data e hora de início e fim da execução, texto da saída padrão e de erros, usuário que disparou a execução, entre outros) são armazenados em um banco de dados para posterior investigação da proveniência dos resultados do experimento. A Figura 33 apresenta os processos armazenados em um projeto.

Listar Processo

Novo Processo

Id	Nome	Program	Status	lastExitValue	lastError	Last Updated	Actions
5	agoscc	agoscc	Done	0		15/03/2016 12:10:12 BRT	
6	g++	GNU C++ wrapper	Done	0		15/03/2016 12:10:25 BRT	
7	FNSimulator	Runner	Iddle			15/03/2016 12:06:25 BRT	
8	gnuplot	GNUplot2PNG	Iddle			15/03/2016 12:06:25 BRT	
18	GNUPlot Wrapper	Groovy Script	Iddle			15/03/2016 12:06:25 BRT	
19	AGOSGNUPlot	Groovy Script	Iddle	0		15/03/2016 12:10:42 BRT	
20	AGOSSensitivityAnalysis	Groovy Script	Iddle			15/03/2016 12:06:25 BRT	
21	templateGNUPlot	Groovy Script	Iddle			15/03/2016 12:06:25 BRT	
26	Run All	Composite BLOCK	Iddle			15/03/2016 12:06:25 BRT	
27	Run preconf	Composite BLOCK	Iddle			15/03/2016 12:06:25 BRT	
28	Run posconf	Composite BLOCK	Iddle			15/03/2016 12:06:25 BRT	

Figura 33 – Listagem de processos dentro do FISIOENV: cada processo dentro de um projeto pode ser executado ou anotado de forma independente.

Processos compostos podem ser criados para realizar a subsequente execução de outros processos de forma a automatizar, em parte ou como um todo, a execução do experimento de acordo com o método idealizado pelo pesquisador. A Figura 34 apresenta três

processos compostos que realizam a execução de outros processos de três forma distintas: em ordem sequencial; condicionada a um valor lógico de um artefato; e repetidamente com base em um valor lógico de um artefato.

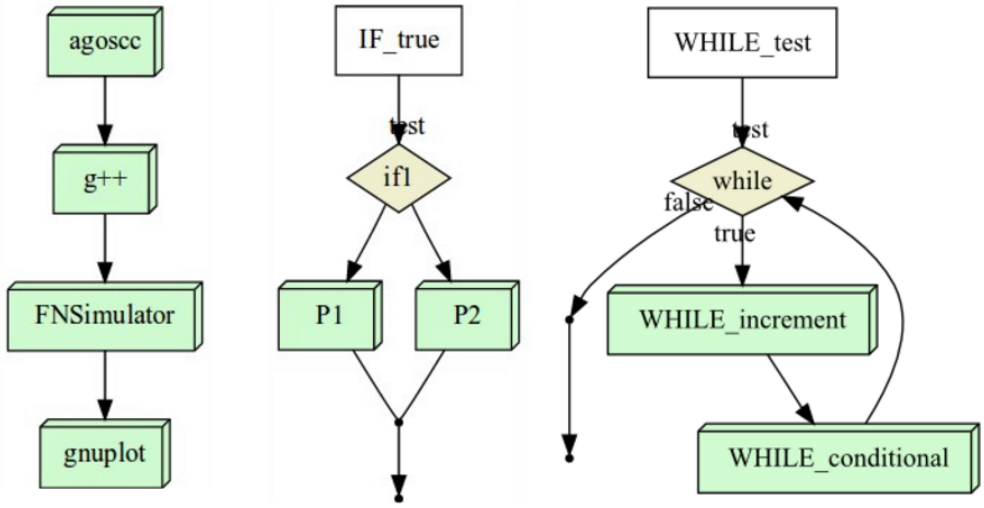


Figura 34 – Processo composto dentro do FISIOENV: os processos podem ser compostos para execução sequencial, condicional ou repetida de uma lista de outros processos.

Através da execução independente e da composição de processos, o pesquisador é capaz de interagir com o ambiente para executar seu experimento como um todo ou compor a execução de etapas complexas em separado. Processos custosos podem ser executados apenas uma vez e a parte de processamento dos resultados pode ser construída ou adaptada incrementalmente de acordo com os resultados parciais da pesquisa.

4.4 Reuso dos experimentos

Um projeto que já tenha sido publicado previamente permite que outros experimentos sejam criados a partir dele. Ao realizar a cópia do projeto, todos os artefatos e processos que não tenham sido explicitamente marcados pelo autor como privados são copiados. A Figura 35 apresenta um projeto copiado dentro de uma mesma instância do ambiente.

Se houver algum impedimento para a cópia dos artefatos ou os programas dos processos não estiverem disponíveis, o FISIOENV gera um artefato ou processo vazio. Isso permite capturar a estrutura do fluxo de trabalho proposta pelo autor original, mas destaca que o novo experimento derivado está incompleto, Havendo, de forma explicita, uma indicação visual para os pontos do experimento derivado, o pesquisador é conduzido a lidar os elementos problemáticos para ter o experimento completo. A Figura 36 mostra um diagrama de um projeto no qual não foi possível obter todos os artefatos.

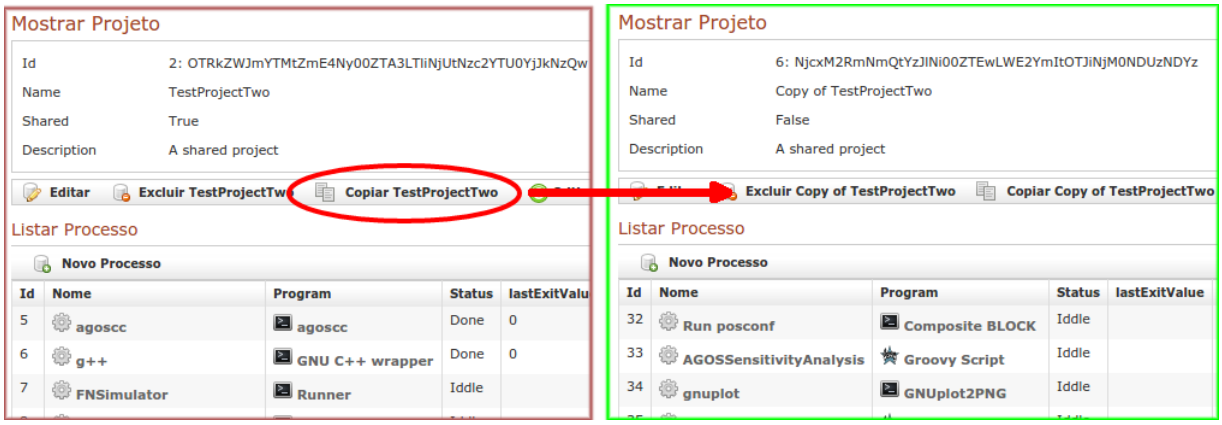


Figura 35 – Um projeto publicado pode ser copiado como um todo.

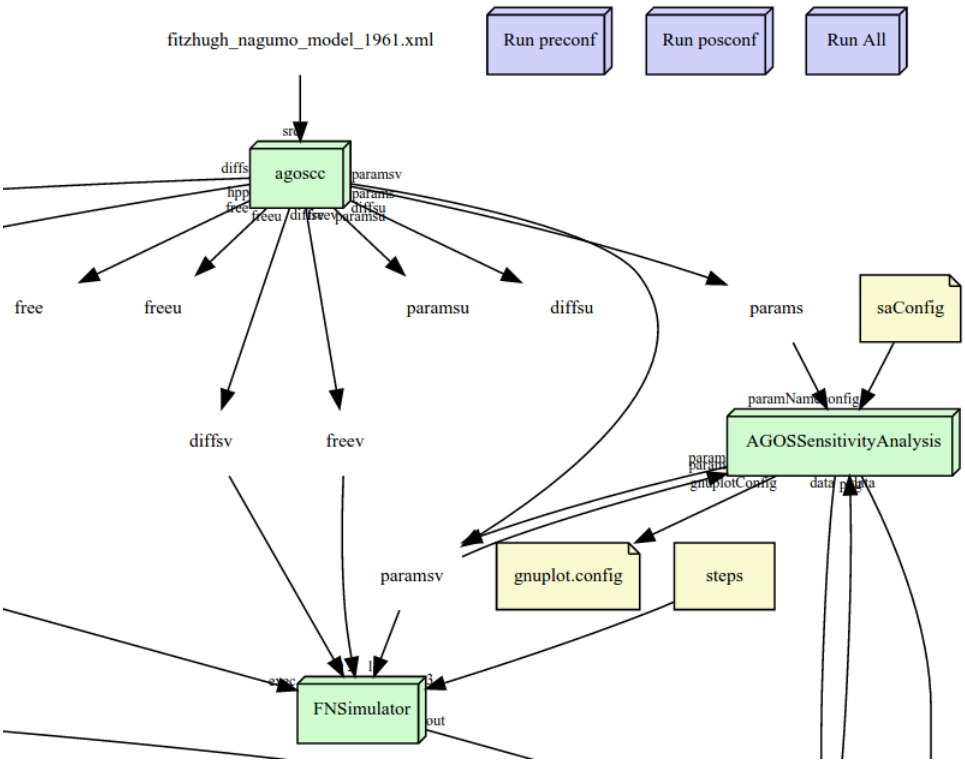


Figura 36 – Recorte do diagrama de um experimento derivado no qual os artefatos `free`, freeu, diffsv, freev, paramsu, diffsu, params e fitzhughnagunomodel1961 não puderam ser obtidos automaticamente.

4.5 Anotações sobre os elementos dos experimentos

As anotações podem ser realizadas sobre todos os elementos de um experimento. Cada projeto, artefato, processo ou programa é capaz de receber anotações informais ou formais. Na implementação do SEE sobre o FISIOENV, cada campo de uma anotação é considerada formal se for URI ou um UUID. Um UUID é utilizado internamente como identificador único para projetos, programas, processos, artefatos, relações ou mesmo anotações. Quando presente como sujeito ou objeto de uma anotação, ele referencia um ele-

mento local da instância do ambiente. Já uma URI, se presente na forma de predicado ou contexto da anotação, é utilizada como um identificador para um conceito semântico do método. A Figura 37 apresenta uma lista de anotações formais sobre um processo.

Figura 37 – Lista de anotações formais sobre um processo: cada campo é uma URI ou um identificador único.

É importante destacar que as anotações são armazenadas dentro dos projetos, mas podem referenciar elementos fora dos limites deles. Por exemplo, um UUID pode referenciar um artefato de outro projeto, indicando que houve um processo de derivação ou cópia. Para os casos nos quais uma URI é fornecida como sujeito ou objeto, o ambiente pode inclusive identificar se parte desta é um UUID. Se sim, o sujeito ou objeto é considerado como um elemento em outra instância do ambiente que deve ser acessado remotamente.

Outros elementos fora dos conceitos capturados pelo método também podem ser referenciados, formal ou informalmente, pelo pesquisador. Basta que esse os relacione ao experimento por uma anotação manual. Como exemplos de referências externas pode-se citar identificadores para artigos ou livros nos quais valores de parâmetros foram obtidos, páginas com instruções para instalação do *software* utilizado por um processo e autoria ou fonte dos dados de entrada do experimento. Essas anotações permitem que outras aplicações extraiam informações ou estendam o ambiente para agregar funções adicionais.

4.6 Críticas sobre o estado do experimento

As anotações são utilizadas durante todo o ciclo de vida do experimento para colher informações sobre o estado da reprodutibilidade dele e servir como guia para o pesquisador. A qualquer momento, o pesquisador pode solicitar que uma crítica seja realizada sobre o projeto para anotar seus elementos com base em seu estado. A Figura 38 apresenta um relatório após a crítica sobre o estado dos artefatos de um experimento, destacando quais apresentaram problemas e as respectivas anotações.

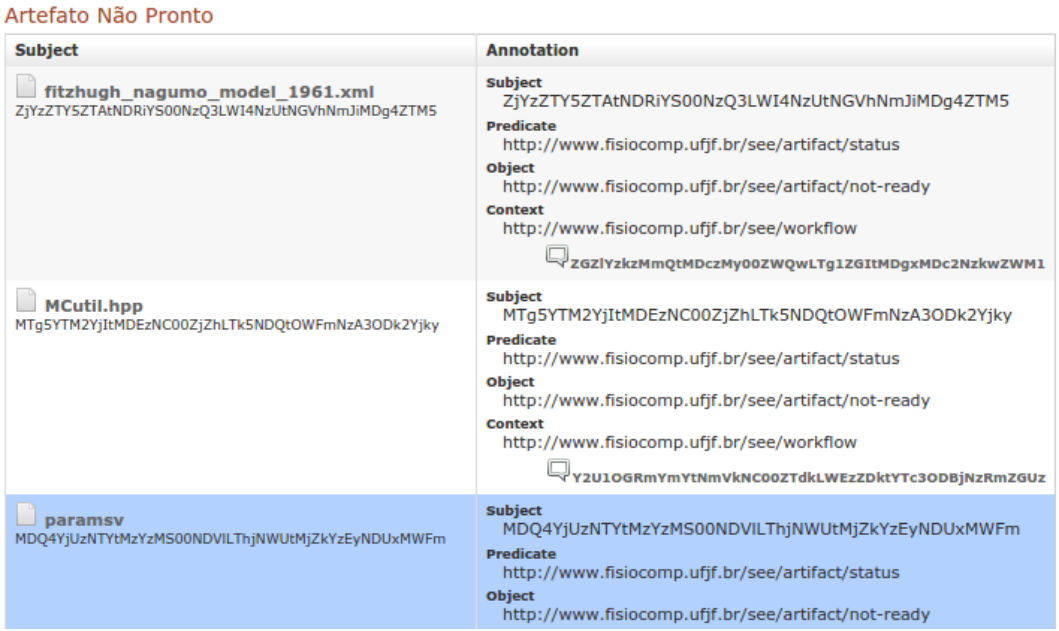


Figura 38 – Relatório da crítica sobre os artefatos de um experimento: destaque quais estão contribuindo negativamente para a reprodução do experimento.

Um artefato não pronto é aquele cujo conteúdo não está disponível para uso imediato pelos processos. Isso implica que os processos, por sua vez, não estarão prontos para execução se seus artefatos de entrada não estiverem também prontos. Um relato de processos que não estão prontos, por problemas em seus artefatos de entrada, pode ser visto na Figura 39.

Se o ambiente detecta que um processo não possui um programa associado, ou se o programa associado não é executável no momento da crítica, uma anotação é criada avisando que o processo não é executável. A Figura 40 apresenta esse cenário no qual dois programas tiveram, intencionalmente, suas permissões de execução retiradas do sistema operacional antes da crítica.

Ao realizar uma cópia de um projeto existente, uma anotação é gerada para referenciar cada elemento do projeto original. Isso permite que todos os elementos do experimento sejam comparados com o original de forma a verificar o estado da reprodução. A Figura 41 apresenta os resultados da comparação entre um experimento derivado de outro, em que

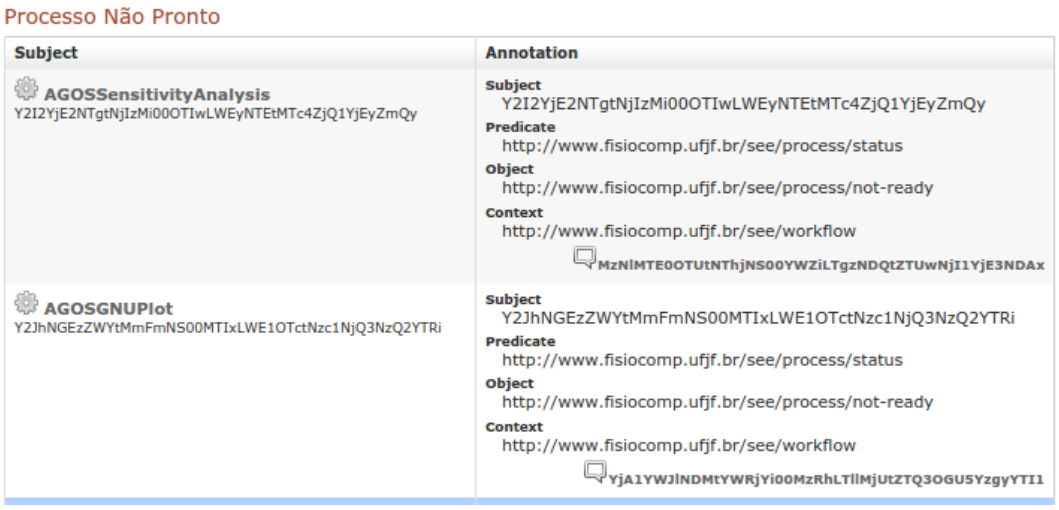


Figura 39 – Os processos que não possuírem os seus artefatos de entrada não são considerados prontos para execução.

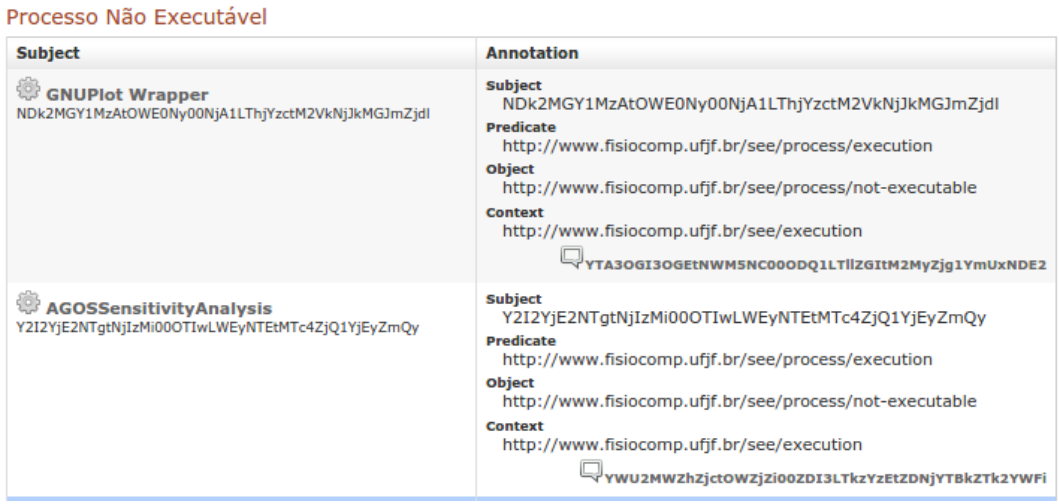


Figura 40 – Os processos que não possuírem os seus programas executáveis não são considerados aptos para execução.

cada artefato é confrontado com o original. Sempre que o conteúdo não for exatamente igual é gerada uma nova anotação além da indicação visual, sugerindo ao pesquisador que uma análise deva ser feita para verificar os dados de entrada ou resultados.

Ao criar um experimento derivado, muitas vezes há o interesse de experimentar uma ferramenta diferente para se obter os mesmos resultados. Esse cenário pode ser avaliado ao verificar se os processos utilizam os mesmos programas do experimento anterior. A Figura 41 apresenta uma comparação na qual os processos são relacionados buscando-se quais programas utilizados foram, ou substituídos por outros, ou se tiveram versões alteradas em relação ao experimento original.

Por fim, fica à disposição do pesquisador um relatório compacto para acompanhar o estado da reprodutibilidade do experimento conforme pode ser visto na Figura 43. O

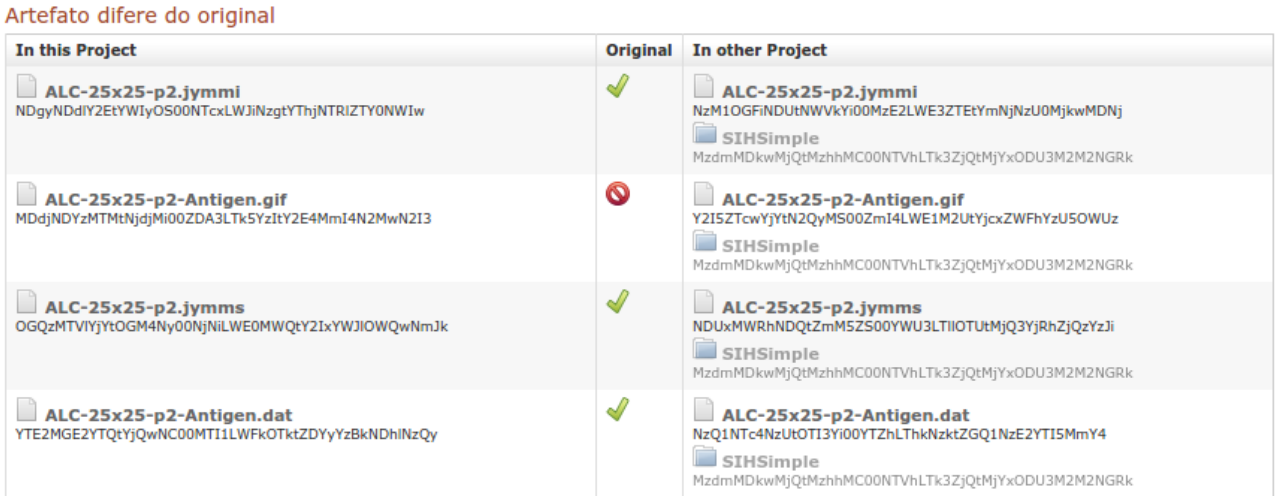


Figura 41 – Como os experimentos mantêm referências para os projetos originais, é possível comparar os artefatos entre eles.

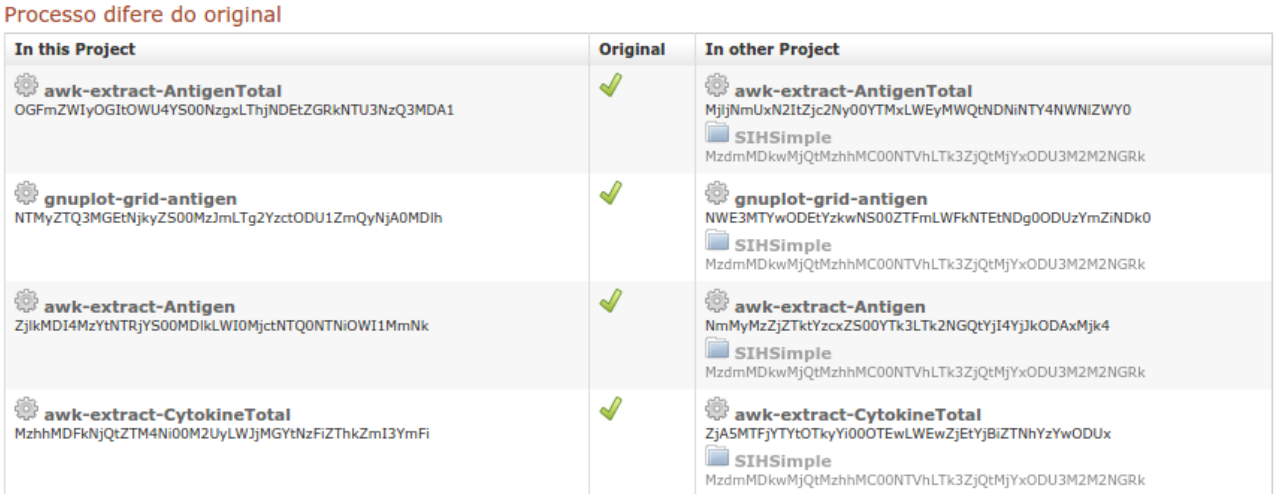


Figura 42 – Como os experimentos mantêm referências para os projetos originais, é possível comparar os processos entre eles.

pesquisador é guiado (mas não obrigado) a só publicar seu experimento quando todos os elementos estiverem acessíveis dentro do ambiente. Para o caso de uma importação, esse relatório também permite traçar uma diretriz para implantação de programas e comparações dos resultados com o experimento original.

4.7 Considerações parciais sobre o SEE

Este capítulo apresentou uma implementação de um ambiente computacional que segue as diretrizes do SEEM. A abordagem não restringe a escolha de tecnologias utilizadas pelo pesquisador para a confecção de seus experimentos. As tecnologias empregadas são baseadas na pilha de *software* Java, que permite sua implantação em diversos sistemas

Figura 43 – Um relatório com a proporção de elementos problemáticos permite acompanhar o estado atual da reprodutibilidade do experimento. O pesquisador é guiado a eliminar todas as pendências antes da publicação.

operacionais nos quais a máquina virtual esteja disponível. A independência do sistema operacional possibilita que instâncias do ambiente possam ser criadas para se adequar a conjuntos de ferramentas e equipamentos utilizados nos experimentos.

Ao se criticar continuamente os experimentos, o sistema gera um conjunto de anotações que guia as ações do pesquisador de forma a conseguir reproduzir ou criar um novo experimento derivado. Através do sistema é possível isolar casos em que dados fundamentais para obtenção dos resultados estão ausentes ou quais ferramentas utilizadas não estão mais funcionais. Ao expor visualmente esses problemas, o uso sistemático do ambiente aponta qual o estado de reprodutibilidade dos experimentos realizados pelos pesquisadores.

No próximo capítulo serão apresentados dois estudos de caso nos quais o ambiente computacional prova de conceito foi aplicado para a captura dos fluxos de trabalho para posterior reprodução e readaptação.

5 Estudos de Caso

Como estudo de caso, dois experimentos reais tiveram seus fluxos de trabalho capturados pelo ambiente construído como prova de conceito. São utilizadas duas linguagens de modelagem para a definição dos experimentos: a primeira usa modelos descritos em CELL-ML e simulados através da ferramenta AGOS; a segunda usa os metamodelos de Dinâmica de Sistemas (BARROS, 2001) e como motor de simulação a JynaCore API (KNOP, 2011). Como domínio de aplicação, são utilizados a eletrofisiologia cardíaca para a primeira (KNOP et al., 2012a) e a dinâmica de modelos do sistema imune inato para a segunda (KNOP et al., 2012b; KNOP et al., 2012c).

5.1 Aplicação em Eletrofisiologia Cardíaca

Esta seção apresenta um estudo de caso onde a ferramenta de linha de comando AGOS (BARBOSA et al., 2006) é integrada no ambiente de experimentos. O AGOS é utilizado para simulações da eletrofisiologia cardíaca. Ele foi concebido para gerar automaticamente código fonte em diversos métodos de integração numérica à partir de modelos descritos no formato CELL-ML (GARNY et al., 2008).

5.1.1 Fluxo de trabalho do AGOS

O fluxo de trabalho associado com a utilização da ferramenta AGOS começa à partir de um arquivo CELL-ML que contém os dados, condições iniciais e metadados de um modelo dinâmico. Este modelo é interpretado pelo binário `agoscc` que utiliza um modelo de código para uma linguagem alvo, gerando o código fonte de um simulador *ad hoc* para as equações diferenciais e um conjunto de arquivos texto que apresentam detalhes de suas variáveis. Esse código fonte precisa ser compilado para gerar um programa executável que consome os valores contidos no arquivo texto como os identificadores, unidades e valores iniciais.

O resultado da saída da execução é um conjunto de dados com o comportamento das variáveis do modelo. Dependendo da natureza do experimento, esses dados devem ainda ser convertidos ou utilizados por outros aplicativos para realizar análises ou desenhos de gráficos como parte dos resultados. A Figura 44 apresenta um diagrama com um exemplo de utilização do AGOS e como esse fluxo de trabalho pode estar contido dentro de um outro experimento. Um experimento pode ser muito maior ou mais complexo e a discussão de um trabalho ou publicação pode omitir trechos dos fluxos de trabalho específicos de ferramentas auxiliares por questões de espaço.

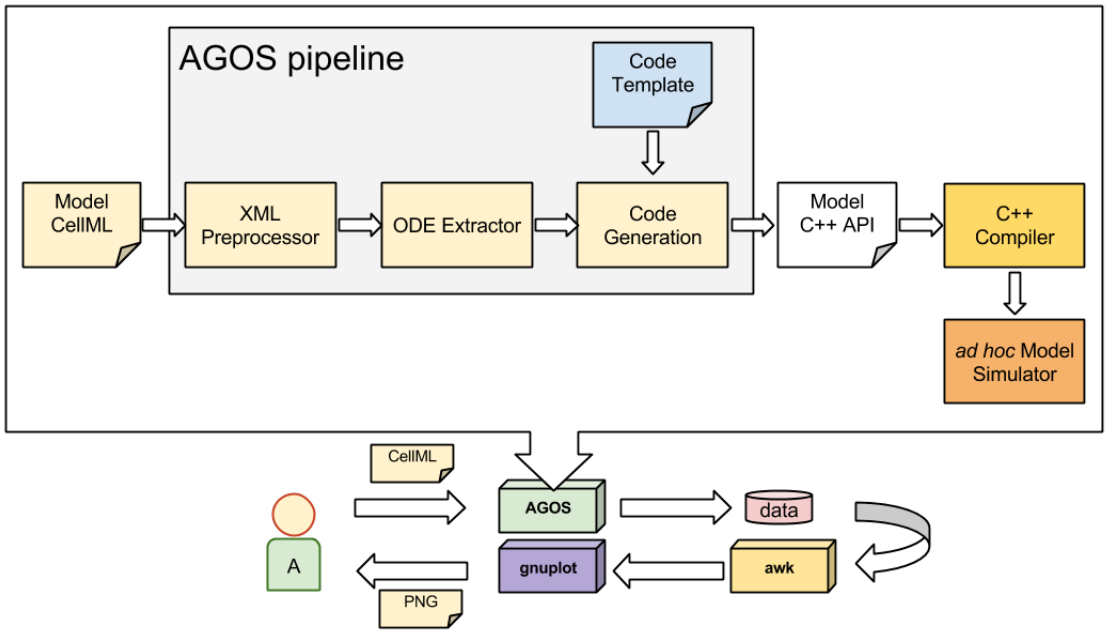


Figura 44 – Detalhes dos passos necessários para se obter o resultado de um experimento utilizando o AGOS

.

A Figura 45 apresenta um exemplo da captura de um *workflow* do AGOS dentro do ambiente. O modelo descrito em CELL-ML de um neurônio excitável (FITZHUGH, 1955 apud KEENER; SNEYD, 1998) é utilizado pelo processo que encapsula o compilador do AGOS. A execução do compilador gera os arquivos texto com os parâmetros (os arquivos `params`, `paramsu` e `paramsv`), condições iniciais (os arquivos `diffs`, `diffu` e `diffsv`), as variáveis livres (os arquivos `free`, `freeu` e `freev`) e arquivos de código fonte do simulador para o modelo (arquivo `FitzhugNaguno.cpp` e `MCUtil.hpp`). Os arquivos de código fonte são consumidos pelo processo que utiliza o compilador GCC e gera o artefato `FNSimulator.exe`. Este, por sua vez, é executado por um processo `FNSimulator` que permite gerar os dados de saída.

Durante a execução do *workflow*, o artefato `result.dat` é gerado pela execução correta do processo `FNSimulator` e é consumido por um processo que utiliza o *GNUplot* para gerar o artefato com um arquivo de imagem `result.png`. A Figura 46 apresenta parte do conteúdo do arquivo de resultado à esquerda e sua representação gráfica à direita.

5.1.2 Análise da reprodutibilidade

O experimento foi continuamente criticado até que a implantação dos programas e artefatos permitissem que o relatório de críticas não apresentasse nenhum problema, conforme pode ser visto na Figura 47. Esse experimento foi publicado no ambiente protótipo para permitir derivá-lo em novos experimentos.

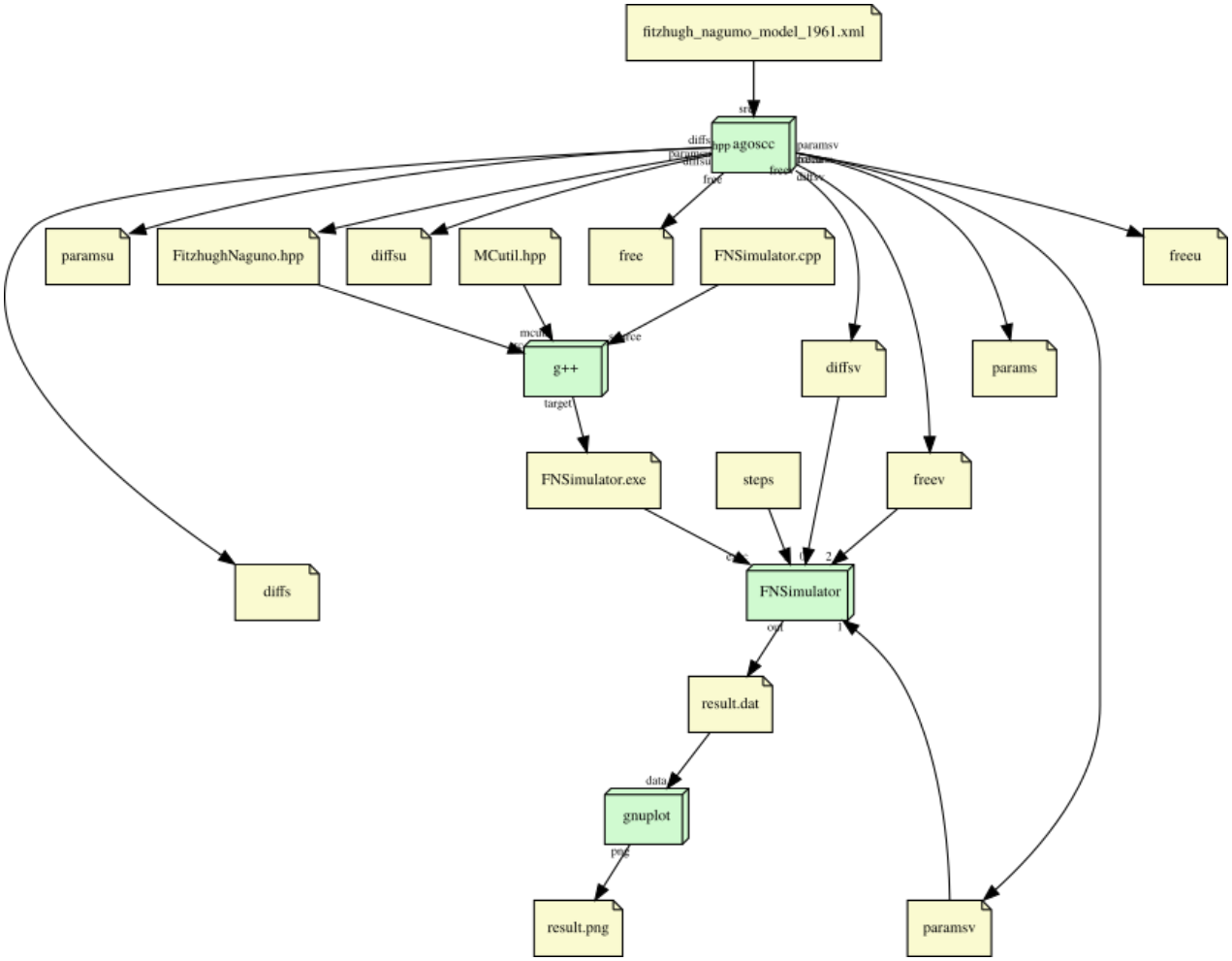


Figura 45 – Fluxo de trabalho modelado na ferramenta FISIOENV uma execução do AGOS como forma de saída de um modelo em CELL-ML.

Um novo experimento foi derivado no mesmo ambiente como uma cópia. Como essa derivação utiliza os mesmos programas, o experimento derivado teve permite a repetição imediata do experimento. A Figura 48 apresenta um trecho do relatório detalhado onde é possível ver a referência (via UUID) para o experimento anterior e a referência para cada um dos seus elementos originais.

Para avaliar a reprodução entre instâncias de ambientes diferentes, uma segunda instância foi configurada, utilizando a mesma versão do *software* do ambiente e mesmo sistema operacional mas sem os todos os programas instalados. A cópia foi realizada para a nova instância, identificando o UUID do experimento original prefixado pelo endereço IP da instância original. Os dados do experimento foram trocados entre instâncias via serviço *web* REST.

Toda a estrutura descrita no experimento original foi importada pelo novo ambiente. Entretanto, conforme pode ser visto na Figura 49, o relatório resumido informou que o experimento tem sua reprodução prejudicada: um artefato difere do referenciado no experimento original e três de seus processos não estão prontos para uso.

```
1.000000e-01 1.673438e-01 9.350000e-04
2.000000e-01 3.332358e-01 1.871275e-03
3.000000e-01 4.976719e-01 2.808809e-03
4.000000e-01 6.606480e-01 3.747584e-03
5.000000e-01 8.221606e-01 4.687583e-03
6.000000e-01 9.822064e-01 5.628791e-03
7.000000e-01 1.140783e+00 6.571190e-03
8.000000e-01 1.297887e+00 7.514763e-03
9.000000e-01 1.453516e+00 8.459493e-03
1.000000e+00 1.607670e+00 9.405364e-03
1.100000e+00 1.760345e+00 1.035236e-02
1.200000e+00 1.911542e+00 1.130046e-02
1.300000e+00 2.061259e+00 1.224965e-02
1.400000e+00 2.209496e+00 1.319991e-02
1.500000e+00 2.356252e+00 1.415123e-02
1.600000e+00 2.501528e+00 1.510359e-02
1.700000e+00 2.645324e+00 1.605697e-02
1.800000e+00 2.787641e+00 1.701135e-02
1.900000e+00 2.928480e+00 1.796672e-02
2.000000e+00 3.067843e+00 1.892307e-02
2.100000e+00 3.205730e+00 1.988036e-02
2.200000e+00 3.342145e+00 2.083860e-02
2.300000e+00 3.477088e+00 2.179776e-02
2.400000e+00 3.610564e+00 2.275782e-02
2.500000e+00 3.742574e+00 2.371876e-02
```

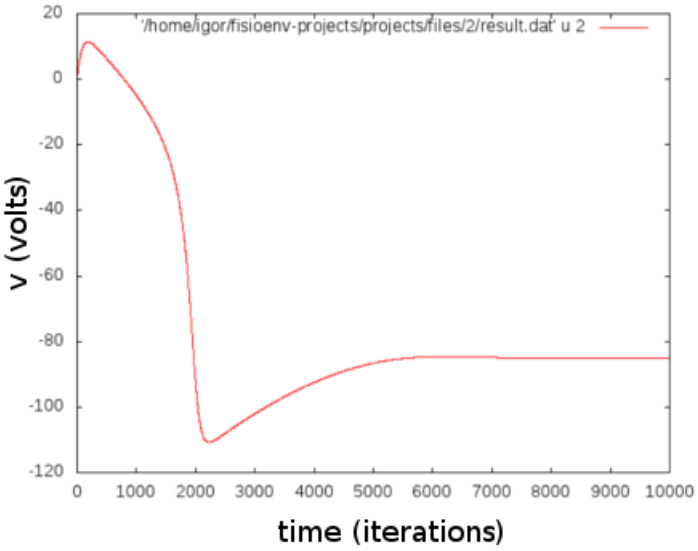


Figura 46 – Conteúdo dos artefatos `result.dat` (à esquerda) e `result.png` (à direita) do fluxo de trabalho do AGOS capturado com o FISIOENV.

O detalhamento permite acompanhar os motivos pelos quais o experimento difere do original. No caso de teste, o artefato foi alterado manualmente após a importação e os processos utilizam programas que não estão implantados na segunda instância do ambiente. A Figura 50 apresenta um trecho do relatório detalhado do estado dos artefatos mas não dos processos Figura 51.

O uso do SEEM permite acompanhar o estado de reprodutibilidade de um experimento em duas frentes: o pesquisador autor do experimento se policia a implantar todos os programas utilizados pelos processos e todos os artefatos consumidos e gerados para publicação. Já o pesquisador que deseja realizar um experimento derivado a este, utiliza o SEEM para guiar seus esforços, de forma sistemática, a obter ou substituir todos os componentes do experimento anterior. Assim, o SEEM e o conjunto de anotações gerado atuam como facilitadores para a obtenção de experimentos mais reprodutíveis.

Na seção seguinte, os metadados e infraestrutura computacional desenvolvida é são utilizados para a criação de uma ferramenta de readequação de experimentos, utilizando esse enfoque na reprodução dos experimentos.

5.1.3 Readequação da infraestrutura de execução

A abordagem permite que os elementos de um *workflow* no FISIOENV sejam reutilizados como arcabouço para se desenvolver novos experimentos. O *workflow* pode inclusive servir como base para a construção de aplicações mais específicas onde sua estrutura é escondida do pesquisador ou estudante interessado nos resultados do experimento. A Figura 52 apresenta uma visão em duas camadas de um aplicação. Na camada supe-

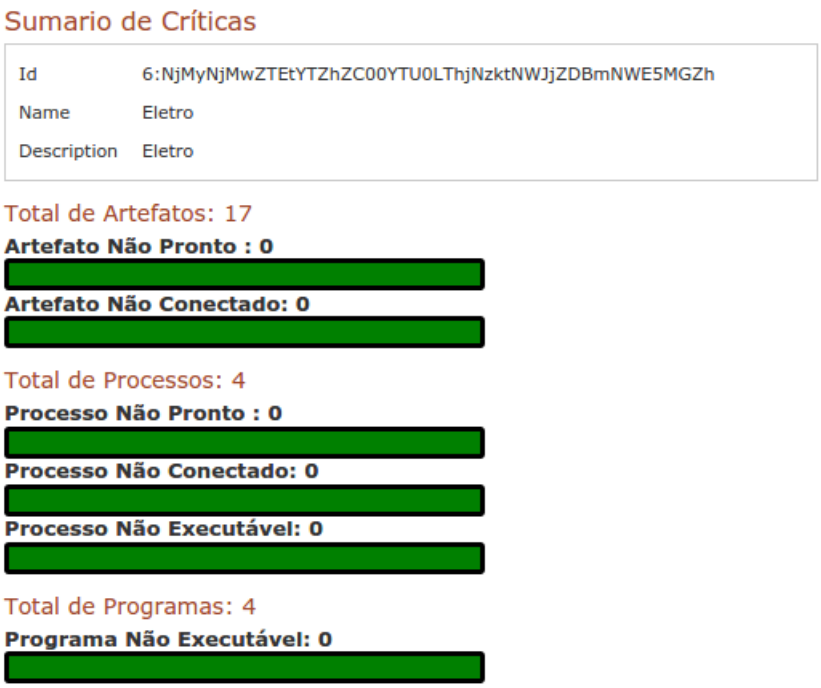


Figura 47 – Relatório de críticas resumido que indica que todos os artefatos e processos estão disponíveis e prontos para uso na instância do ambiente.

rior, os conceitos mais específicos do domínio são expostos ao usuário final e internamente traduzidos em um fluxo de trabalho na camada inferior utilizando os construtores padrão de *workflows*.

Com essa abordagem, foi construída uma nova aplicação capaz de reaproveitar um experimento existente como base para outros experimentos. A aplicação foi criada para observar a sensibilidade de parâmetros de modelos de eletrofisiologia cardíaca ou outro domínio. Essa aplicação fornece uma interface *web* para readequar o *workflow* do AGOS, visto na Seção 5.1.1. O usuário pode realizar o envio de um modelo em CELL-ML e os arquivos gerados pelo processo do `agoscc` são usados para identificar e configurar os detalhes da simulação. O usuário pode selecionar um dos parâmetros do modelo que será variado um determinado número de vezes dentro de uma porcentagem de seu valor definido no modelo. A cada variação do parâmetro, um novo conjunto de dados é gerado e seu comportamento é exposto em um gráfico de saída.

Como exemplo de aplicação, foi utilizado um experimento *in silico* que procura reproduzir a prolongação do potencial de ação observada em experimentos *in vitro* com camundongos chagásicos na fase aguda da doença. Um modelo de miócito cardíaco de camundongo proposto por Bondarenko et al. (2004) é utilizado para verificar se a variação na condutividade do canal responsável pela corrente I_{to} reproduz o comportamento sugerido pelo experimento com miócitos cardíacos de cães no estágio agudo da doença de Chagas (PACIORETTY et al., 1995).

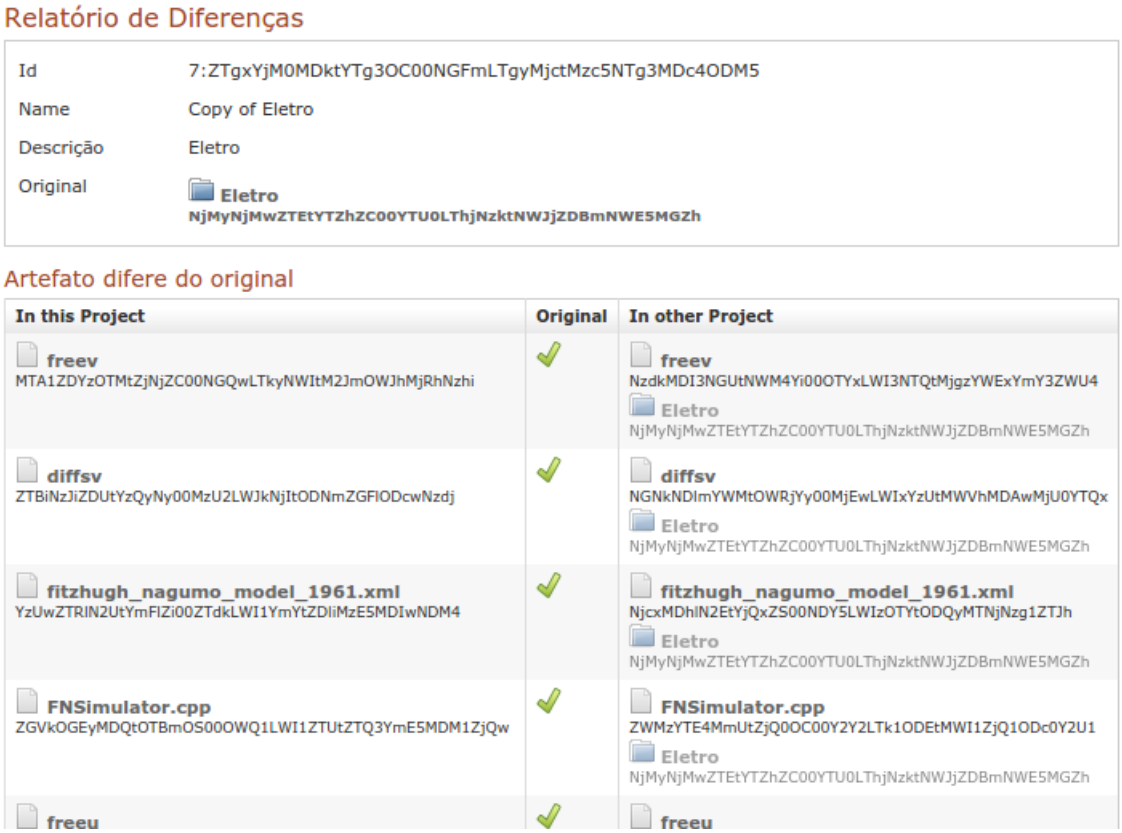


Figura 48 – Relatório detalhado do experimento derivado no qual é possível observar que há a referência para o experimento original e para cada um de seus elementos de origem. Isso permite que o ambiente realize uma comparação entre o conteúdo de cada artefato e verifique se os programas necessários são os mesmos e se estão implantados.

A aplicação foi construída de forma que, para o usuário final, nenhum termo do domínio do FISIOENV (programas, processos e artefatos) apareça para o usuário do AGOSWeb. Para o usuário final do AGOSWeb existem apenas conceitos como modelo, arquivos de configuração e gráfico de saída. O cenário para uso de tal aplicação assume que o *workflow* de um experimento é bem definido, mas a interação com seus parâmetros e modelos é realizada de forma explorativa. É possível utilizá-lo para criar experimentos rapidamente como forma de se habituar ao comportamento do modelo ou como ferramenta dentro de um ambiente educacional. A Figura 53 apresenta a tela de configuração do experimento logo após o envio do arquivo CELL-ML. Os parâmetros do modelo são expostos para edição e há a possibilidade de se executar a variação dos valores de um parâmetro sem expor como a estrutura do *workflow* é criada.

Internamente ao AGOSWeb, um *workflow* é construído pela aplicação com o intuito de realizar a variação dos parâmetros por manipulação direta dos artefatos. A Figura 54 mostra uma amostra do diagrama, automaticamente gerado, que representa a composição de processos e seus artefatos de entrada e saída que vão formar um fluxo de trabalho

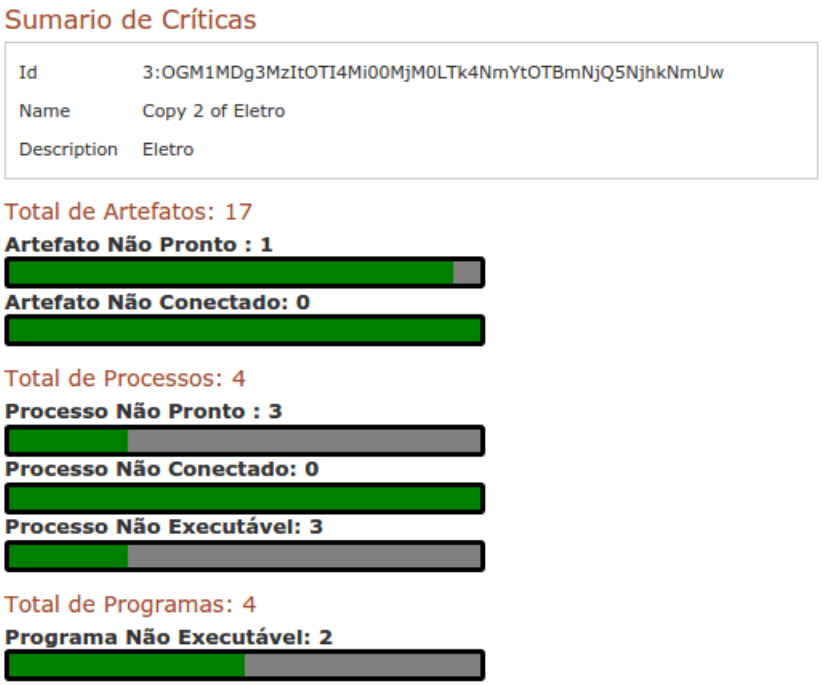


Figura 49 – Relatório resumido de críticas de um experimento que foi derivado para uma instância de ambiente diferente. Um dos artefatos foi alterado manualmente e três processos não possuíam os programas implantados no ambiente.

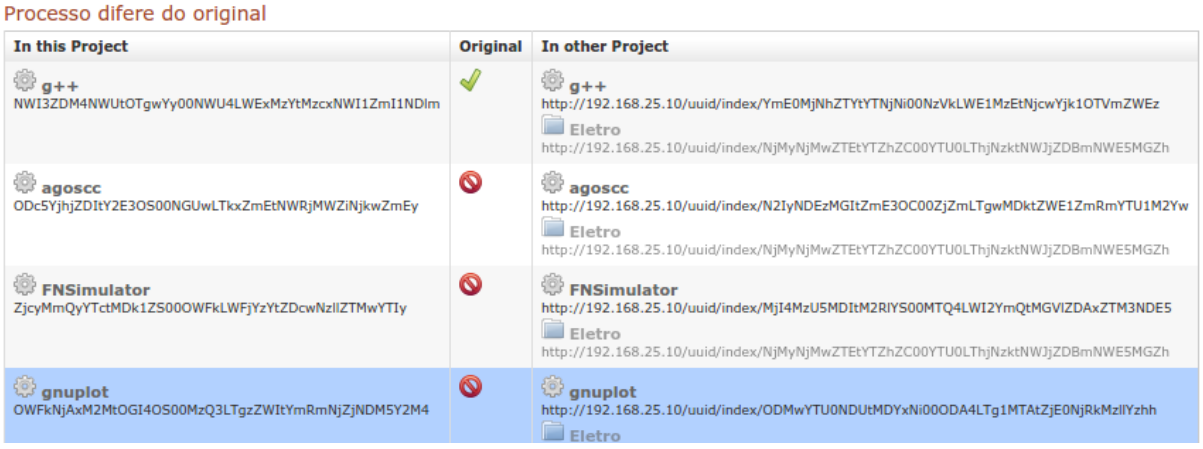


Figura 50 – O relatório detalhado de processos indica que os processos `agoscc`, `FNSimulator` e `gnuplot` não estão disponíveis na segunda instância do ambiente e devem ser implantados pelo Administrador de Programas. Todos os processos também mantêm uma referência para os processos no experimento original na outra instância de ambiente.

na aplicação *web* final.

Após o envio de um modelo em CELL-ML para o modelo Bondarenko et al. (2004), a execução do *workflow* gera a saída apresentada pela Figura 56. Esta figura mostra como a forma de onda do modelo é afetada pela baixa regulação dos diferentes canais iônicos. É possível observar que diminuindo a condutância máxima de I_{to} (representada por g_{Kto}

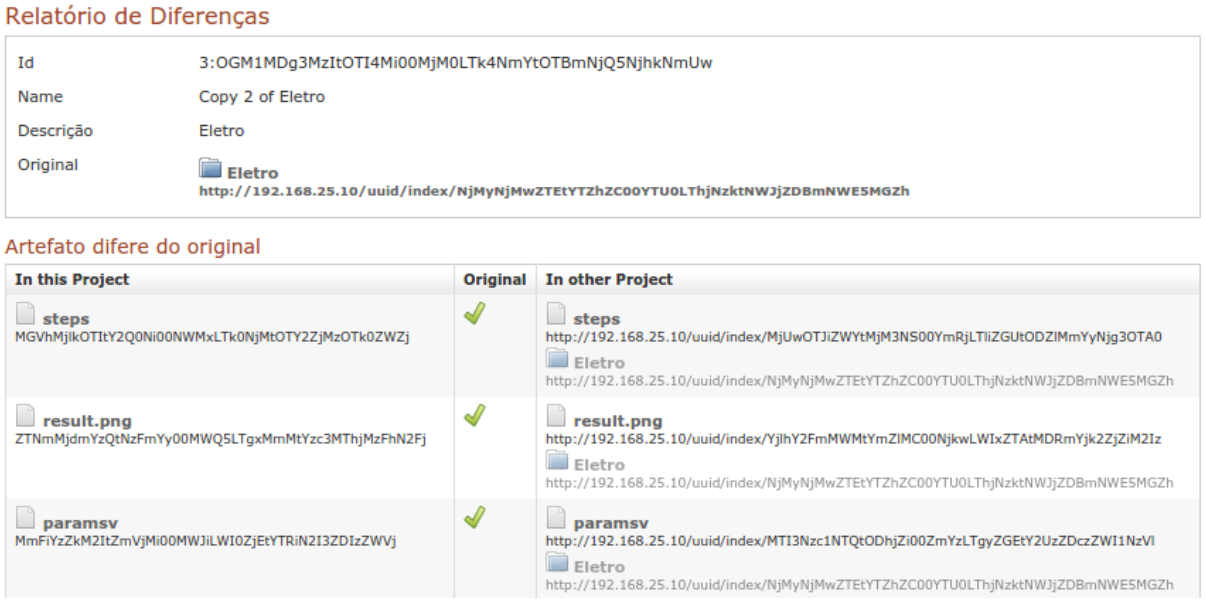


Figura 51 – Parte do relatório detalhado de artefatos onde se compara o estado atual de cada artefato com o referenciado em outro ambiente via anotações. Os artefatos disponíveis são comparados uma a um e se procura por divergências entre dois experimentos.

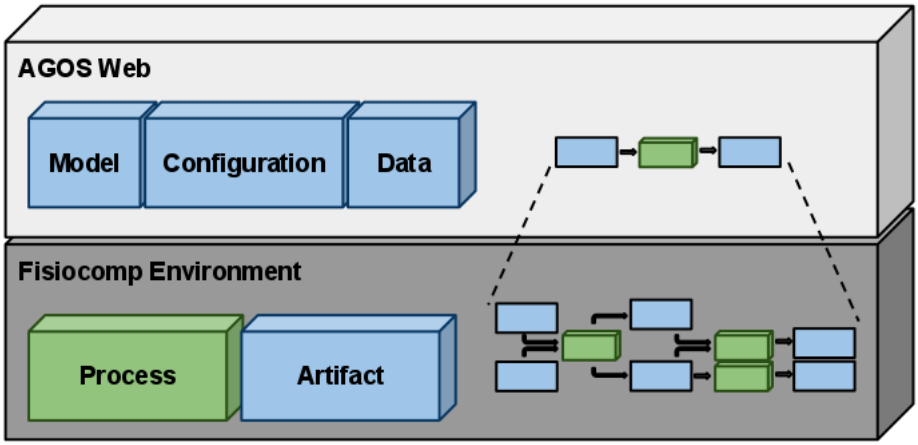


Figura 52 – Modelo de arquitetura do AGOSWeb: há um mapeamento direto dos conceitos utilizados para o desenvolvimento do sistema com os conceitos de processos e artefatos do FISIOENV.

do modelo matemático e `g_K_to` no computacional) a duração do potencial de ação é prolongada, em conformidade com os experimentos *in vitro* apresentados anteriormente na Figura 5, vista na revisão conceitual e reproduzida aqui como Figura 55.

Entretanto, os experimentos *in silico* não foram capazes de reproduzir quantitativamente as mudanças observadas na forma de onda do potencial de ação dos miócitos chagásicos, em particular o longo prolongamento da duração do potencial de ação (mais que o dobro). Portanto, trabalhos subsequentes são necessários para melhor caracterizar e

Figura 53 – Recorte da interface *web* para a configuração de um modelo enviado para o AGOSWeb.

entender as mudanças eletrofisiológicas geradas durante a fase aguda da doença de Chagas ou a adequação do modelo aos dados experimentais.

5.1.4 Conclusões parciais de Eletrofisiologia

O FISIOENV foi bem sucedido para capturar o *workflow* de experimentos *in silico* utilizando ferramentas de linha de comando como o AGOS, GCC e *GNUPLOT*. O *workflow* capturado vai além da leitura de um modelo do disco por uma ferramenta *ad hoc*. Dentro do ambiente, além de sua repetição imediata, é possível realizar sua reprodução ao readequar seu fluxo de trabalho para a construção de novos experimentos. Um experimento pôde inclusive ser utilizado como arcabouço para a construção de uma ferramenta de experimentação, exclusiva para a experimentação da variação de parâmetros para análise de sensibilidade em modelos de eletrofisiologia cardíaca baseados em CELL-ML.

5.2 Modelagem e Metamodelagem do Sistema Imune Inato Humano

Esta seção apresenta um estudo de caso onde uma ferramenta de modelagem e simulação de metamodelos de Dinâmica de Sistemas, o JynaCore API (KNOP, 2011) é utilizada para capturar a dinâmica básica de parte das células do sistema inato humano.

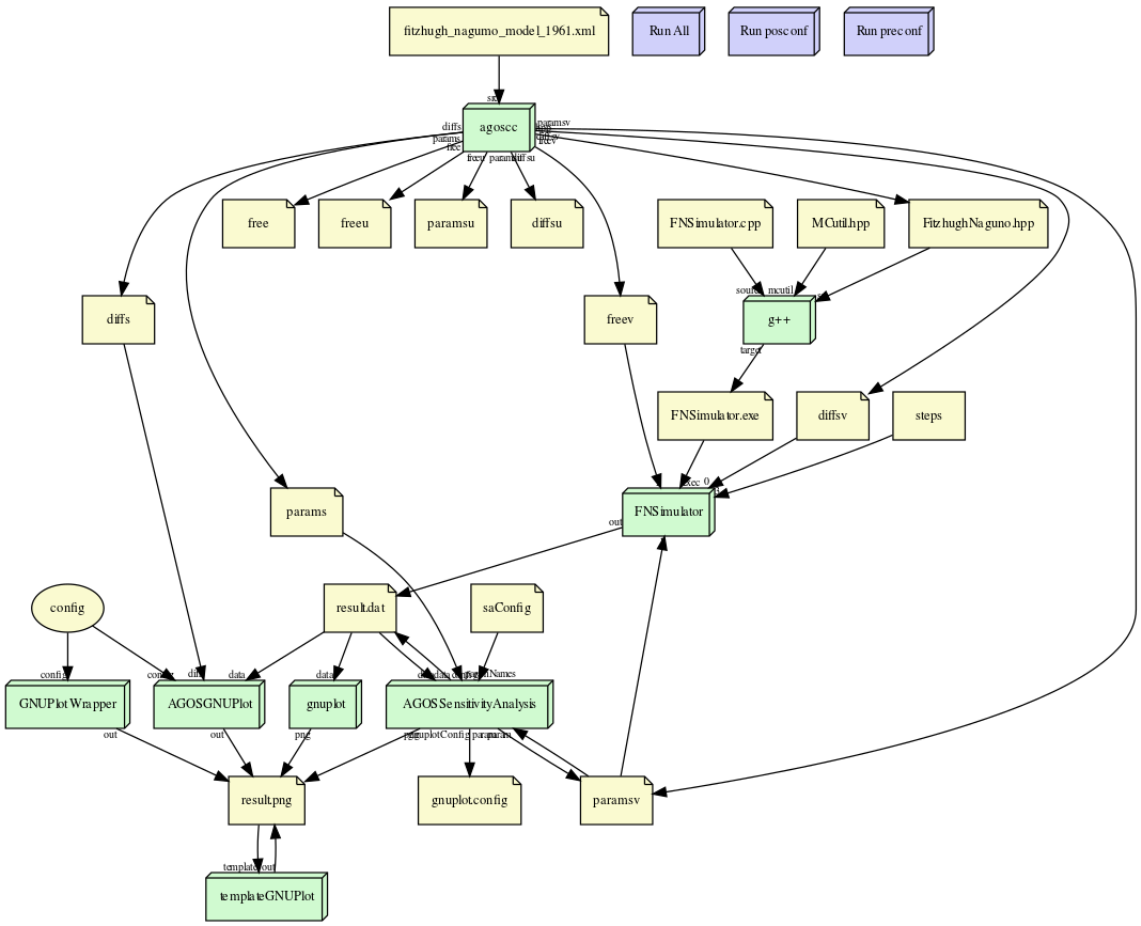


Figura 54 – Fluxo de trabalho interno gerado pelo AGOSWeb para reaproveitar um *workflow* existente e os construtores do FISIOENV.

Os metamodelos de Dinâmica de Sistemas foram utilizados para montar uma discretização de uma configuração planar de seções de tecido (KNOP et al., 2012c). O *workflow* do experimento é capturado utilizando o ambiente FISIOENV e vários experimentos derivados são criados proporcionando o reuso de modelos e ferramentas à partir do experimento inicial.

5.2.1 Construção do Modelo do Sistema Imune Inato

O experimento *in silico* desenvolve um modelo do sistema imune inato sob os efeitos causados pela administração repetida da endotoxina LPS, simulando uma resposta imune aguda equivalente a que uma infecção bacteriana causaria. O estudo da administração de LPS é importante pois, dependendo da dosagem, uma injeção pode ser letal(SENALDI et al., 1999), mas efeitos dinâmicos podem conduzir à tolerância e potencialização dos efeitos(DAY et al., 2006). Nesta seção é detalhado o método utilizado para a construção do modelo utilizando os metamodelos de Dinâmica de Sistemas.

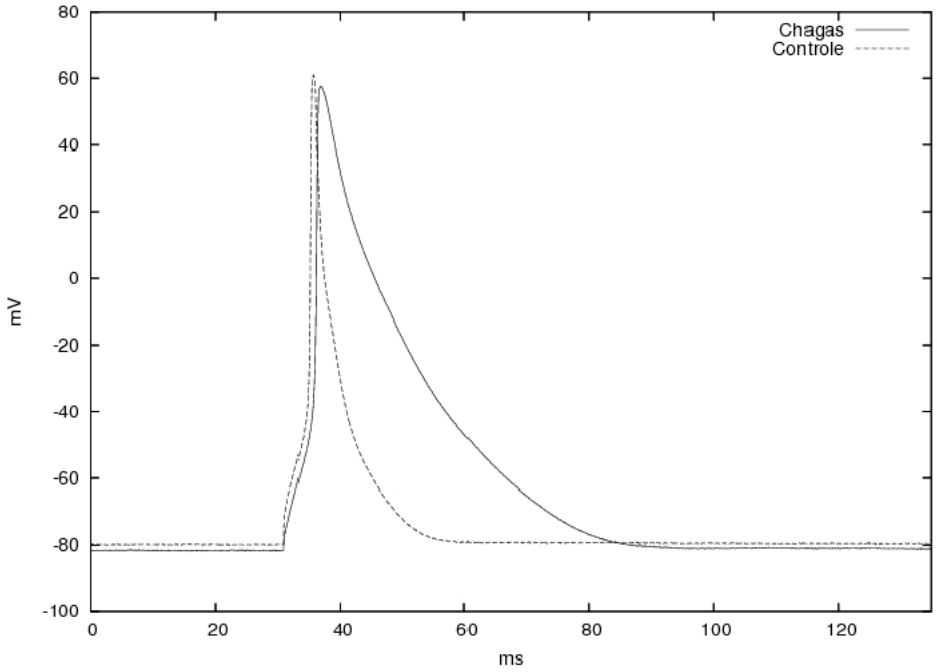


Figura 55 – Experimentos *in vitro*: formas de onda de controle e miócitos de ratos chagásicos. Adaptado de Campos (2011).

5.2.2 Modelo Matemático Reduzido do Sistema Imune Inato Humano

O metamodelo de Dinâmica de Sistemas proposto se baseia em um sistema de EDPs, originalmente proposto por Pigozzo (2011) e identificado como modelo reduzido do sistema imune inato. O modelo é dito reduzido por expor um conjunto de equações que descreve a resposta dinâmica do sistema imune inato ao LPS em uma secção microscópica de tecido pela interação de apenas três tipos de células. São modeladas as interações entre os leucócitos (representando os macrófagos e neutrófilos), as citocinas produzidas pelos mesmos e o LPS que servirá como um antígeno sem geração própria. A Figura 57 apresenta um esquema com a relação entre leucócitos, citocinas pró-inflamatórias e LPS.

O LPS difunde pelas seções do tecido e provoca a ação dos leucócitos, que reconhecem o LPS e realizam a sua fagocitose. Assume-se que o processo de fagocitose induz, de uma forma rápida, a apoptose de leucócitos, ou seja, uma morte programada da célula branca necessária para manter o equilíbrio interno da população de forma a evitar que ela continue agindo por mais tempo cause danos não desejados ao organismo. A citocina pró-inflamatória é produzida após os receptores da membrana dos leucócitos reconhecerem o LPS. Essa citocina induz um aumento na permeabilidade endotelial, permitindo que mais leucócitos possam sair dos vasos sanguíneos e entrar no tecido infectado. Além disso, a citocina pró-inflamatória é quimioatraente de leucócitos, guiando seu movimento em um processo que é conhecido como quimiotaxia, ou seja, o deslocamento de células em função de um estímulo químico. Como resultado, os leucócitos se movem no sentido do gradiente de concentração de citocina pró-inflamatória para o local onde a infecção está sendo com-

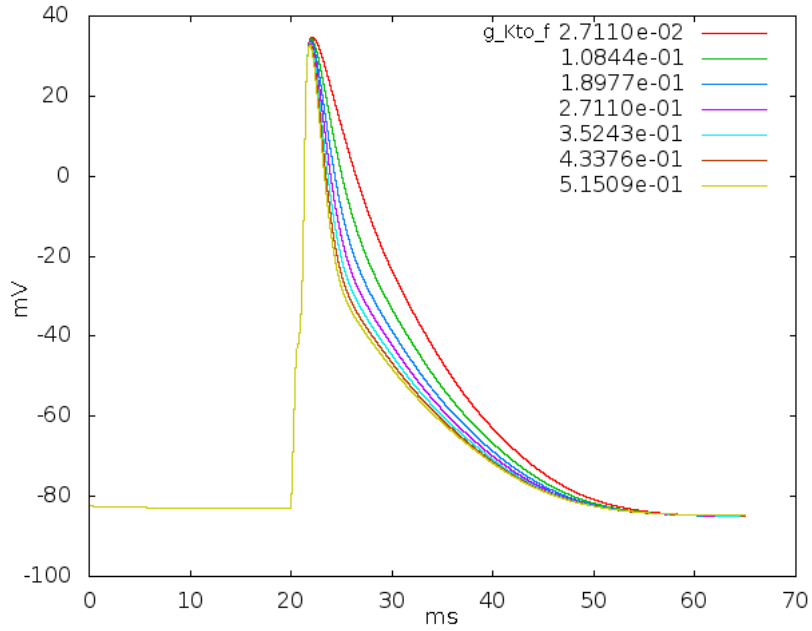


Figura 56 – Análise de sensibilidade apresentando os efeitos na forma de onda do potencial de ação para parâmetros g_{Kto_f} associados à máxima condutância de I_{to}.

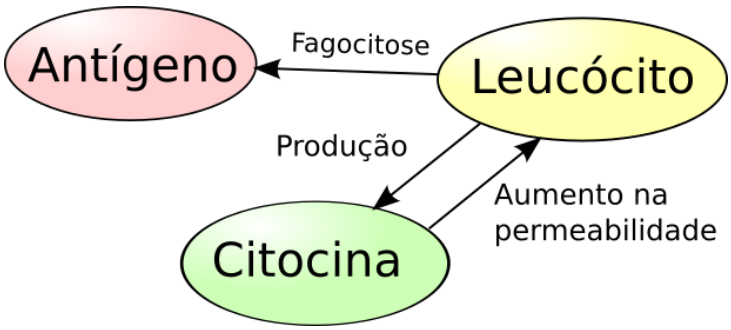


Figura 57 – Relações causais entre leucócitos, antígenos e citocina. Adaptado de Pigozzo (2011).

batida (PIGOZZO, 2011). O autor destaca que as principais características capturadas pelo modelo matemático são:

- Os leucócitos interagem com as citocinas pró-inflamatórias e com o LPS;
- A interação entre os leucócitos e LPS aumentam a produção de citocinas;
- As citocinas induzem um aumento na permeabilidade endotelial e permitem que mais leucócitos cheguem a seções de tecido infectado;
- No tecido, os leucócitos movem-se na direção do gradiente de citocina pro-inflamatória (quimiotaxia);
- A citocinas atraem os leucócitos para as regiões nas quais a concentração de LPS é maior.

O conjunto de equações é dado nas Equações 5.1, 5.2 e 5.3 onde A, L e C representam, respectivamente, a população de LPS, leucócitos e citocinas pró-inflamatórias. A equação do LPS é apresentada na Equação 5.1.

$$\begin{cases} \frac{\partial A}{\partial t} = -\mu_A A - \lambda_{L|A} A.L + D_A \Delta A \\ A(x,0) = A_0 \quad | \quad 0 \leq x < 1, \frac{\partial A(.,t)}{\partial n}|_{\partial\Omega} = 0 \end{cases} \tag{5.1}$$

O termo $\mu_A A$ modela o decaimento do LPS, onde μ_A é sua taxa de decaimento. O termo $\lambda_{L|A} A.L$ modela a fagocitose dos LPS pelos leucócitos, onde $\lambda_{L|A}$ é a taxa de fagocitose. O termo $D_A \Delta A$ modela a difusão dos LPS no tecido onde D_A é o coeficiente de difusão e $DeltaA$ o gradiente de concentração de antígenos. Esse coeficiente de difusão é um fator de proporcionalidade representando a quantidade de substância difundindo por uma área unitária devido a um gradiente de concentração por um tempo unitário.

A equação dos leucócitos é apresentada na Equação 5.2.

$$\begin{cases} permeability = ((Pmax - Pmin).C/(C + Keqch) + Pmin) \\ sourceL = permeability.(LmaxTissue - L) \\ \frac{\partial L}{\partial t} = -\mu_L L - \lambda_{A|L} A.L + D_L \Delta L + sourceL - \nabla.(\chi_L L \nabla C) \\ L(x,0) = L_0, \frac{\partial L(.,t)}{\partial n}|_{\partial\Omega} = 0 \end{cases} \tag{5.2}$$

O termo $((Pmax - Pmin).C/(C + Keqch) + Pmin)$ usa a equação de Hill (S MAURIN M, 2008) para modelar como a permeabilidade do endotélio para os vasos sanguíneos depende da concentração local de citocinas. As equações de Hill também são utilizadas para, por exemplo, modelar as relações entre dose-resposta a drogas (WAGNER, 1968 apud PIGOZZO, 2011). A ideia é modelar o aumento da permeabilidade do endotélio de acordo com a concentração de citocinas pró-inflamatórias. Na equação de Hill, $Pmax$ representa a taxa máxima de aumentos na permeabilidade induzida pelas citocinas pró-inflamatórias, $Pmin$ representa a taxa mínima de aumento na permeabilidade que exerce 50% do efeito máximo no aumento da permeabilidade.

O termo $\mu_L L$ modela a apoptose do leucócito onde μ_L é a taxa de apoptose. O termo $\lambda_{A|L} A.L$ modela a apoptose induzida pela fagocitose, onde $\lambda_{A|L}$ representa a taxa de apoptose induzida. O termo $D_L \Delta L$ modela a difusão do leucócito, onde D_L é o coeficiente de difusão. O termo $sourceL$ representa a fonte de leucócitos, ou seja, a quantidade de leucócitos que estão entrando no volume da seção de tecido a partir de um vaso sanguíneo. Esse número depende da permeabilidade do endotélio, representada por $permeability$, e da capacidade de leucócitos que o tecido consegue suportar, representada por $LmaxTissue$, que também pode ser interpretada como a concentração de leucócitos no sangue. Nesse modelo nós consideramos que $LmaxTissue$ é constante ao longo do tempo.

O termo $\nabla.(\chi_L L \nabla C)$ modela o processo de quimiotaxia dos leucócitos, onde χ_L é a taxa que regula o efeito da quimiotaxia. A quimiotaxia é o efeito que permite aos leucócitos se guiarem em direção à região de infecção em função do gradiente de citocinas.

A concentração de citocina é apresentada pela Equação 5.3. O termo $\mu_C C$ modela o decaimento natural da citocina pró-inflamatória, onde μ_C é a taxa de decaimento. Já o termo $\beta_{C|L}.L.A$ modela a produção de citocina pró-inflamatória pelos leucócitos, onde $\beta_{C|L}$ é a taxa de produção. O termo $D_C \Delta C$ modela a difusão da citocina entre as seções de tecido, onde D_C é o coeficiente de difusão.

$$\begin{cases} \frac{\partial C}{\partial t} = -\mu_C C + \beta_{C|L}.L.A + D_C \Delta C \\ C(x,0) = 0, \frac{\partial C(.,t)}{\partial n}|_{\partial\Omega} = 0 \end{cases} \tag{5.3}$$

A Tabela 4, adaptada de Pigozzo (2011), apresenta as condições iniciais e valores utilizados para os parâmetros nas simulações.

Tabela 4 – Condições iniciais e parâmetros

Parâmetro	**Valor**	**Unidade**
L_0	$2 < x < 5$	células
C_0	$0 < x < 5$	células
A_0	$50 < x < 1$	células
$Pmax$	1	1/dia
$Pmin$	0,001	1/dia
$LmaxTissue$	250000	células/mm^3
$keqch$	5	células/mm^3
μ_A	0,01	1/dia
$\lambda_{L\|A}$	0,55	1/(células/mm^3).dia
D_A	0,05	μm^2/dia
μ_L	0,67	1/dia
$\lambda_{A\|L}$	0,55	1/(células/mm^3).dia
X_L	10	mm^2/dia
D_L	10	mm^2/dia
μ_C	12	1/dia
$\beta_{C\|L}$	1	1/(células/mm^3).dia
D_C	6	mm^2/dia

5.2.3 Desenvolvimento dos metamodelos de Dinâmica de Sistemas

A metamodelagem de Dinâmica de Sistemas foi aplicada para a construção de um modelo básico do sistema imune inato usando o JynaCore API como ferramenta principal de simulação. O JynaCore Simulator (JynaSim) (KNOP, 2011) foi utilizado para a montagem dos diagramas e simulação da dinâmica para os modelos que não envolvem a distribuição espacial. Para as aplicações mais complexas, como a simulação das malhas

maiores, um simulador em linha de comando foi construído e o GNUPLOT (GNUPLOT) novamente foi utilizado para realizar o desenho da evolução das quantidades de leucócitos, antígenos e citocinas ao longo do espaço e tempo. Com o modelo base construído, foram definidos vários experimentos para investigar a dinâmica associada com a resposta do sistema imune inato em função da injeção de antígenos. Cada experimento foi criado ao definir um modelo de cenário que altera do modelo base para alterar a injeção de antígenos.

O modelo adimensional apresentado na Seção 5.2.2 para a dinâmica de leucócitos, antígenos e citocinas foi implementado utilizando os diagramas de estoque e fluxo da Dinâmica de Sistemas, como mostrado na Figura 58.

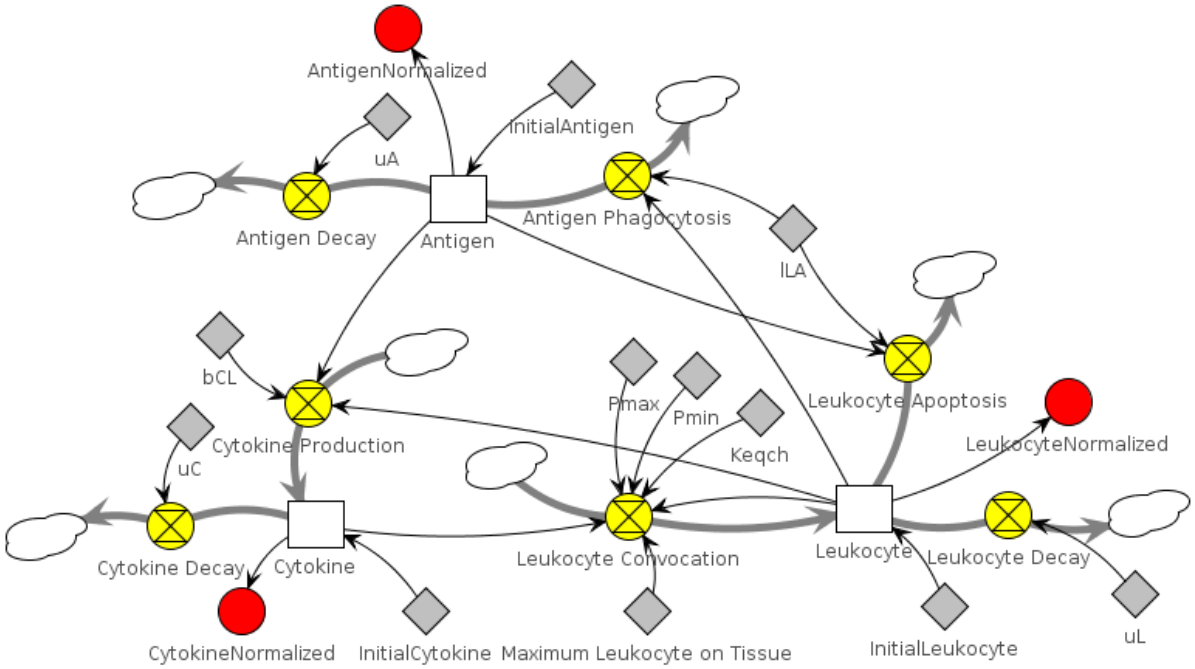


Figura 58 – O modelo em Dinâmica de Sistemas dos antígenos (`Antigen`), leucócitos (`Leukocyte`) e citocina (`Cytokine`).

A Figura 59 mostra a dinâmica do modelo do sistema imune inato após a adição de 200 antígenos durante o estado de repouso do sistema. Pode-se observar o aumento na concentração de leucócitos devido ao aumento da permeabilidade do tecido que, por sua vez, foi causada pelo aumento da concentração de citocinas. A Figura 59 também mostra a diminuição na concentração de antígenos até a sua eliminação total, devido à resposta imune dada pela fagocitose realizada pelos leucócitos. Com a eliminação do antígeno, o número de leucócitos move-se para o estado de equilíbrio após alguns dias.

Esse modelo apresenta as mesmas características do modelo matemático reduzido proposto em Pigozzo et al. (2011) e serve como modelo base para os próximos experimentos. As próximas seções incluem as características espaciais no modelo base para

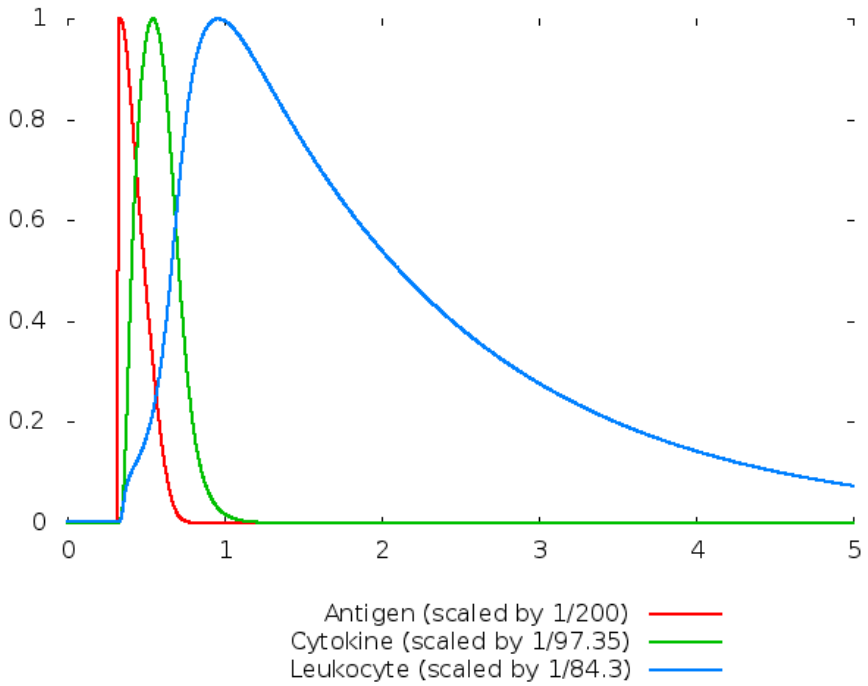


Figura 59 – A dinâmica temporal entre os antígenos (`Antigen`), leucócitos (`Leukocyte`) e citocina (`Cytokine`).

acrescentar os efeitos de difusão e quimiotaxia das células envolvidas na resposta imune.

5.2.3.1 Modelagem da difusão em uma dimensão

Para descrever o comportamento dinâmico de uma seção microscópica do tecido, foi definido o modelo do sistema imune define através de uma classe `Section`. Para modelar uma distribuição espacial unidimensional foram definidos dois relacionamentos entre as instâncias de `Section` em um espaço discretizado, como conexões "west" e "east" entre as seções vizinhas. A Figura 60 apresenta como a difusão unidimensional de antígenos é modelada utilizando a ferramenta de metamodelagem de Dinâmica de Sistemas. Cada classe `Section` possui duas variáveis externas, $east.Antigen$ e $west.Antigen$, representadas por circunferências com linhas tracejadas, que indicam referências a estoques externos que serão utilizados para calcular a entrada (ou saída) do fluxo da população através das fronteiras desse volume de tecido. Os valores dos fluxos de difusão dependem dos valores $Antigen$ dos vizinhos. Para expressar esse efeito, os auxiliares são automaticamente associados durante a definição de um relacionamento entre duas instâncias.

A população inicial de antígenos é definida pela propriedade $InitialAntigen$, que é usada como o valor inicial do estoque $Antigen$ antes do primeiro passo de simulação. Os fluxos de difusão na Equação 5.4c e na Equação 5.4d usam as propriedades dx, discretização do volume de tecido, e D_A, o coeficiente de difusão de antígenos, para calcular a população de antígeno que sai ou entra na seção. Ambos os fluxos são baseados nos

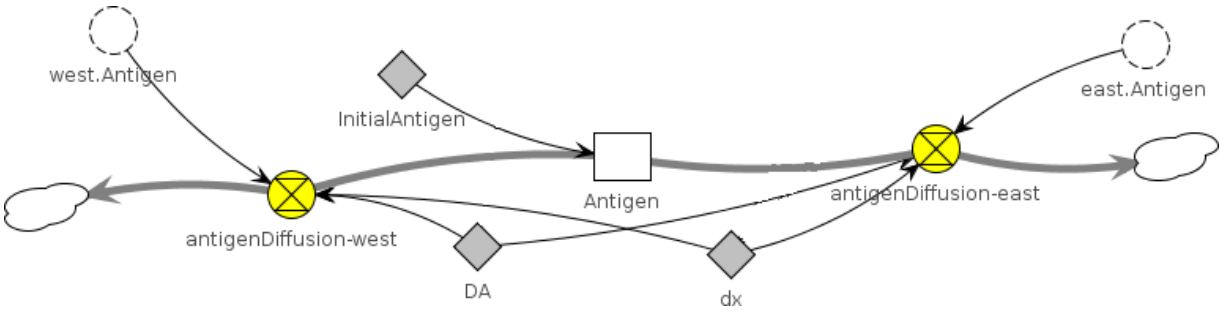


Figura 60 – Difusão de antígenos no volume da seção. Foi utilizado um estoque para modelar a quantidade de antígenos neste volume e dois fluxos para descrever a difusão para as seções vizinhas.

valores de *Antigen* das seções vizinhas.

$$InitialAntigen = 5.0 \tag{5.4a}$$

$$Antigen(0) = InitialAntigen \tag{5.4b}$$

$$antigenDiffusion_{west} = \frac{D_A \cdot (Antigen - west.Antigen)}{dx^2} \tag{5.4c}$$

$$antigenDiffusion_{east} = \frac{D_A \cdot (Antigen - east.Antigen)}{dx^2} \tag{5.4d}$$

Essa abordagem por relacionamentos entre instâncias é também usada para modelar a difusão das populações de leucócitos e de citocinas. O uso dos metamodelos de Dinâmica de Sistemas apresenta uma abordagem intuitiva para a modelagem planar e espacial de substância pois o inter-relacionamento entre volumes finitos fica explícito pelos diagramas. A Figura 61 mostra um modelo de instância com cinco seções dispostas de forma linear. Cada uma das instâncias de classe pode ter seus próprios estados e propriedades, mas o comportamento é definido pela sua estrutura interna descrita no modelo de classe de domínio. Se no modelo de domínio a classe muda, todas as suas instâncias terão seu comportamento alterado.

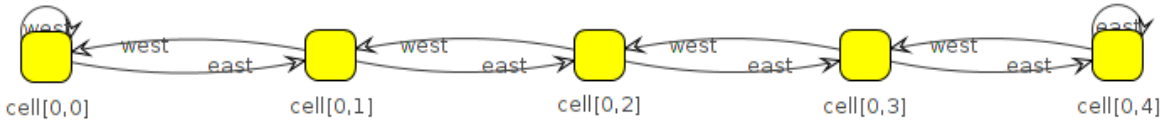


Figura 61 – Modelo de instância com cinco seções de tecido.

5.2.3.2 Modelo da quimiotaxia dos leucócitos em uma dimensão.

Semelhante à implementação do operador de difusão, o operador matemático de quimiotaxia modela o deslocamento de leucócitos para as seções vizinhas. Entretanto, o

movimento se dá no sentido do gradiente de citocinas ao invés do gradiente de leucócitos. O operador de quimiotaxia foi implementado em nosso metamodelo de Dinâmica de Sistemas utilizando o mesmo método utilizado para a difusão.

Cada classe "Section" possui, além de seus estoques locais *Leukocite* e *Citokyne*, referências para quatro estoques finitos através dos relacionamentos: *east.Leukocyte*, *west.Leukocyte*, *east.Cytokine*, *west.Cytokine* conforme visto na Figura 62. Os estoques externos são definidos durante a ligação de instâncias por um relacionamento (como apresentado na Figura 61, anteriormente) e são representados por auxiliares com linhas tracejadas. Estes serão utilizados para calcular a entrada (ou saída) do fluxo da população através dos limites deste volume de tecido. Para a quimiotaxia, os valores dos fluxos vão depender dos valores de seus estoques finitos *Leukocyte* e *Citokine* quando comparados aos valores desses estoques nos vizinhos, conforme pode-se observar nas Equações 5.5 e 5.6. Esta abordagem é uma implementação do cálculo dos fluxos pelo esquema *upwind* para diferenças finitas (HEATH, 1997).

$$InitialLeukocyte = 1.0 \tag{5.5a}$$

$$cytokine_{v-west} = \frac{(west.Cytokine - Cytokine)}{dx} \tag{5.5b}$$

$$cytokine_{v-east} = \frac{(east.Cytokine - Cytokine)}{dx} \tag{5.5c}$$

$$Leukocyte(0) = InitialLeukocyte \tag{5.5d}$$

$$leukocyteChemiotaxis_{west} = \begin{cases} \text{if}(cytokine_{v-west} > 0.0) \\ \qquad X_N \cdot \frac{Leukocyte}{dx} \cdot \frac{(west.Cytokine - Cytokine)}{dx} \\ \text{else} \\ \qquad X_N \cdot \frac{west.Leukocyte}{dx} \cdot \frac{(Cytokine - west.Cytokine)}{dx} \end{cases} \tag{5.6a}$$

$$leukocyteChemiotaxis_{east} = \begin{cases} \text{if}(cytokine_{v-east} > 0.0) \\ \qquad X_N \cdot \frac{Leukocyte}{dx} \cdot \frac{(east.Cytokine - Cytokine)}{dx} \\ \text{else} \\ \qquad X_N \cdot \frac{east.Leukocyte}{dx} \cdot \frac{(Cytokine - east.Cytokine)}{dx} \end{cases} \tag{5.6b}$$

5.2.3.3 Modelo de tecido em duas dimensões do sistema imune inato

A abordagem utilizada para criar a estrutura do modelo linear foi estendida para incluir dois outros vizinhos para cada classe de seção de volume de tecido: norte e sul. Isso permitiu modelar e simular a resposta do SIH em um tecido planar levando em

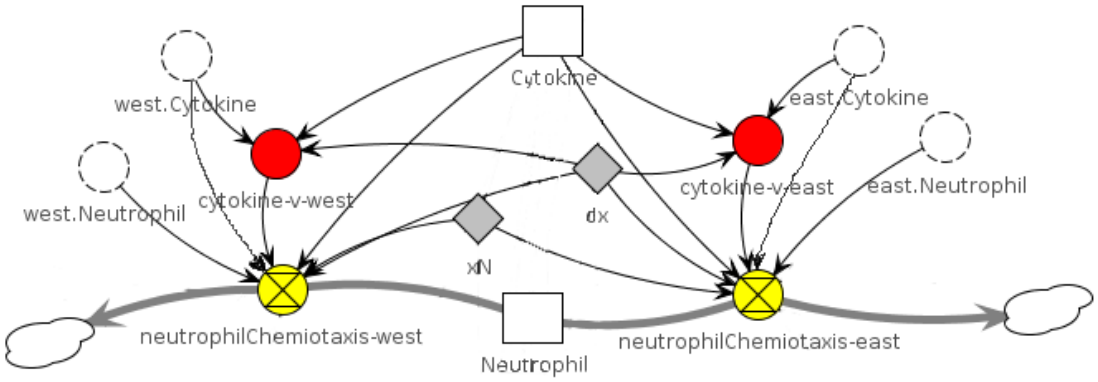


Figura 62 – Modelo de domínio da quimiotaxia no tecido em uma dimensão. A quimiotaxia é controlada pela concentração de citocina (`Cytokine`) que influencia os dois fluxos de saída e assim a concentração de leucócitos na região.

.

conta que cada seção possui quatro vizinhos. Na Figura 63 foram utilizados os quatro relacionamentos, de forma análoga com o modelo linear.

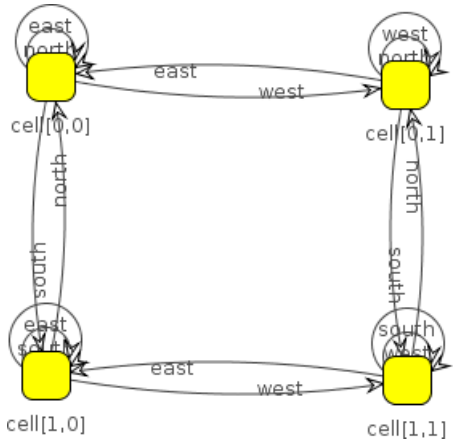


Figura 63 – Um modelo de quatro seções de tecido usando os relacionamentos “north”, “south”, “west” e “east” para representar uma distribuição espacial discreta. Auto-relacionamentos são usados para modelar a região de contorno da região do tecido.

Essa abordagem pode ser estendida para um modelo com qualquer número de seções. A Figura 64 mostra um modelo com 25 seções de tecido distribuídos em uma malha de 5 por 5, gerada por um aplicativo que realiza a associação das instâncias de classes de metamodelos.

5.2.3.4 Novos experimentos com o uso de cenários de metamodelos

De posse do modelo de domínio e sua implementação na forma de modelo de instância, o comportamento do sistema pode ser explorado. Entretanto, seria necessário

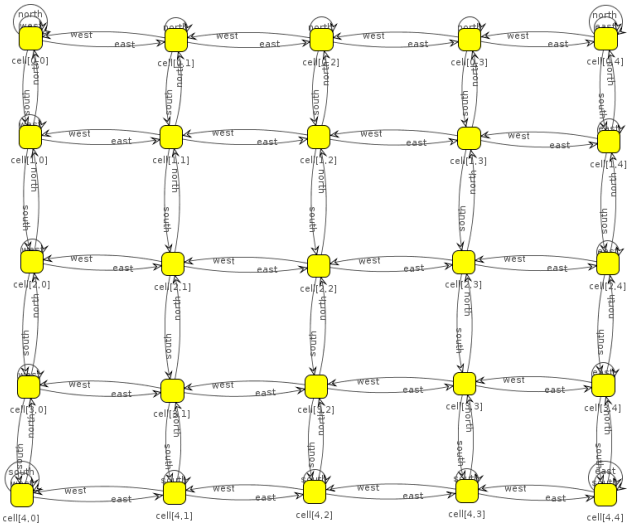


Figura 64 – Modelo de instância com vinte e cinco instâncias de seções em um tecido planar. Configurando um modelo discreto com condições de contorno de *von* Neumann que não permite a troca das populações com limites fora do volume modelado.

adicionar classes mais especializadas ao modelo de domínio para criar novos experimentos *in silico*, introduzindo mais complexidade ao modelo base. Para evitar a introdução de novos conceitos no modelo de domínio, os cenários da metamodelagem de Dinâmica de Sistemas foram utilizados.

Primeiramente, o cenário *Periodic Antigen Injection* foi criado para ser responsável por adicionar uma nova fonte de antígenos, injetando uma determinada quantidade após um número fixo de passos de simulação. O cenário é associado com uma classe *Section* no modelo de domínio, mas somente no modelo instância ele é conectado e tem efeito sobre uma instância em particular. Após a conexão com a instância de classe, esse cenário adiciona uma fonte de antígenos dependente do passo de simulação. A Equação 5.1 passa a ser reescrita pela Equação 5.7 nas instâncias afetadas pelo cenário.

$$\begin{cases} \frac{\partial A}{\partial t} = -\mu_A A - \lambda_{N|A} A.N + D_A \Delta A + ASource(t) \\ ASource(t) = \begin{cases} \text{if}(\frac{t}{dt} \bmod 1000 = 0.0) \\ \qquad \frac{8.0}{dt} \\ \text{else} \\ \qquad 0.0 \end{cases} \\ A(x,0) = A_0 \quad | \quad 0 \leq x < 1, \frac{\partial A(.,t)}{\partial n}|_{\partial\Omega} = 0 \end{cases} \tag{5.7}$$

A Equação 5.7 inclui o termo $ASource(t)$ para realizar a injeção de antígenos (8 células por mm^3) a cada 1000 passos de simulação.

Um segundo cenário, chamado *Inject On Lower Value*, considera a injeção de

antígenos depois que o valor atual de concentração no tecido cai abaixo de um pré-determinado limiar (0,15 células por mm^3). Este cenário também afeta a Equação 5.1 que é reescrita para ficar como na Equação 5.8:

$$\begin{cases} \frac{\partial A}{\partial t} = -\mu_A A - \lambda_{N|A} A.N + D_A \Delta A + ASource(A) \\ ASource(A) = \begin{cases} \text{if}(A < 0.15) \\ \qquad \frac{8.0}{dt} \\ \text{else} \\ \qquad 0.0 \end{cases} \\ A(x,0) = A_0 \quad | \quad 0 \leq x < 1, \frac{\partial A(.,t)}{\partial n}|_{\partial\Omega} = 0 \end{cases} \tag{5.8}$$

Na Equação 5.8 é possível ver a inclusão de $ASource(t)$ na equação antiga, mas com a injeção de um impulso de antígenos apenas quando a concentração for menor que o limiar (novamente com 8 células por mm^3).

Uma vez que também há o interesse em capturar as populações totais de leucócitos, antígenos e a quantidade de citocinas em todas as seções de tecido, é preciso acrescentar uma outra classe que se associa com todas as instâncias de seções do tecido, a classe `Profiler`. A Figura 65 apresenta o modelo proposto para esta nova classe e a Equação 5.9 mostra o uso da função que realiza o somatório dos respectivos auxiliares. Essa é uma classe abstrata usada para somar as populações em todas as seções de tecido. Ela usa relacionamentos múltiplos *Watch* para ler as concentrações de todas as seções conectadas às suas instâncias.

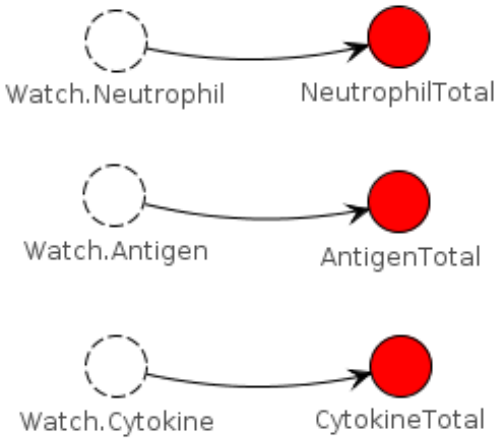


Figura 65 – Modelo de domínio para a classe *Profiler*.

$$\begin{cases} LeukociteTotal = SUM(Watch.Leukocite) \\ AntigenTotal = SUM(Watch.Antigen) \\ CytokineTotal = SUM(Watch.Cytokine) \end{cases} \tag{5.9}$$

A Figura 66 mostra o modelo de domínio com todas as classes de domínio e seus relacionamentos. A classe `Section` tem quatro auto-relacionamentos para modelar a distribuição espacial e condições de contorno no modelo de instância. O elemento `Watcher` é uma instância da classe `Profiler` que tem um relacionamento múltiplo com a classe `Section`. Dois cenários são associados à classe `Section` para serem utilizados nos modelos de instância para a injeção de antígenos periódica e por limiar.

Um novo modelo de instância usando auto-relacionamentos de instâncias de `Section` é criado para definir uma malha de 5 por 5 representando um tecido plano microscópico. Com uma instância `Watcher` da classe `Profiler`, é possível obter todas as concentrações conectando as instâncias de `Section` através do multi-relacionamento. Além disso, o cenário *Periodic Antigen Source* é conectado à instância *section[02,02]* para configurar a única fonte de antígenos no centro da malha para se observar a dispersão em todas as direções. Esse modelo de instância é apresentado na Figura 67 com destaque para seção *section[02,02]* no meio da malha e para a instância *Watcher*.

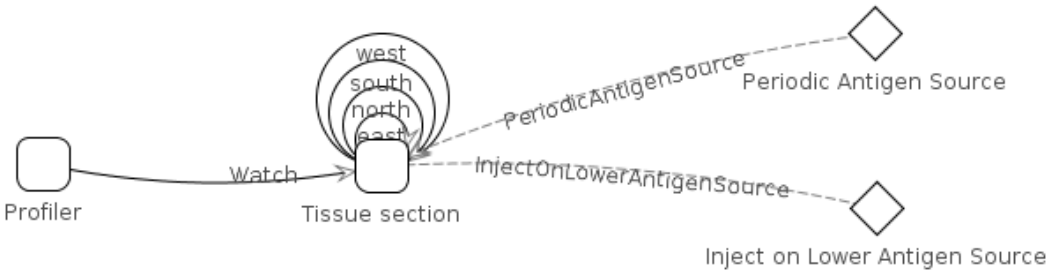


Figura 66 – Modelo de domínio para o sistema imune humano com cenário.

A simulação desse modelo permitiu obter os resultados apresentados na Figura 68 que mostra a dinâmica dos leucócitos, citocinas e antígenos no tecido para três instantes diferentes dada uma concentração inicial de antígenos na seção central. No início da simulação, na primeira linha, é possível ver a população inicial de leucócitos no tecido em estado permanente, não há citocinas e apenas a população inicial de antígenos. Durante a evolução do tempo, a população de antígenos diminui devido à fagocitose realizada pelos leucócitos. Durante a fagocitose há a liberação de citocinas pelos leucócitos, o que aumenta a permeabilidade endotelial, permitindo recrutar mais leucócitos ao ponto de infecção. A produção de citocinas cessa após o agente patogênico ser controlado e a sua concentração diminui rapidamente devido ao seu rápido decaimento natural.

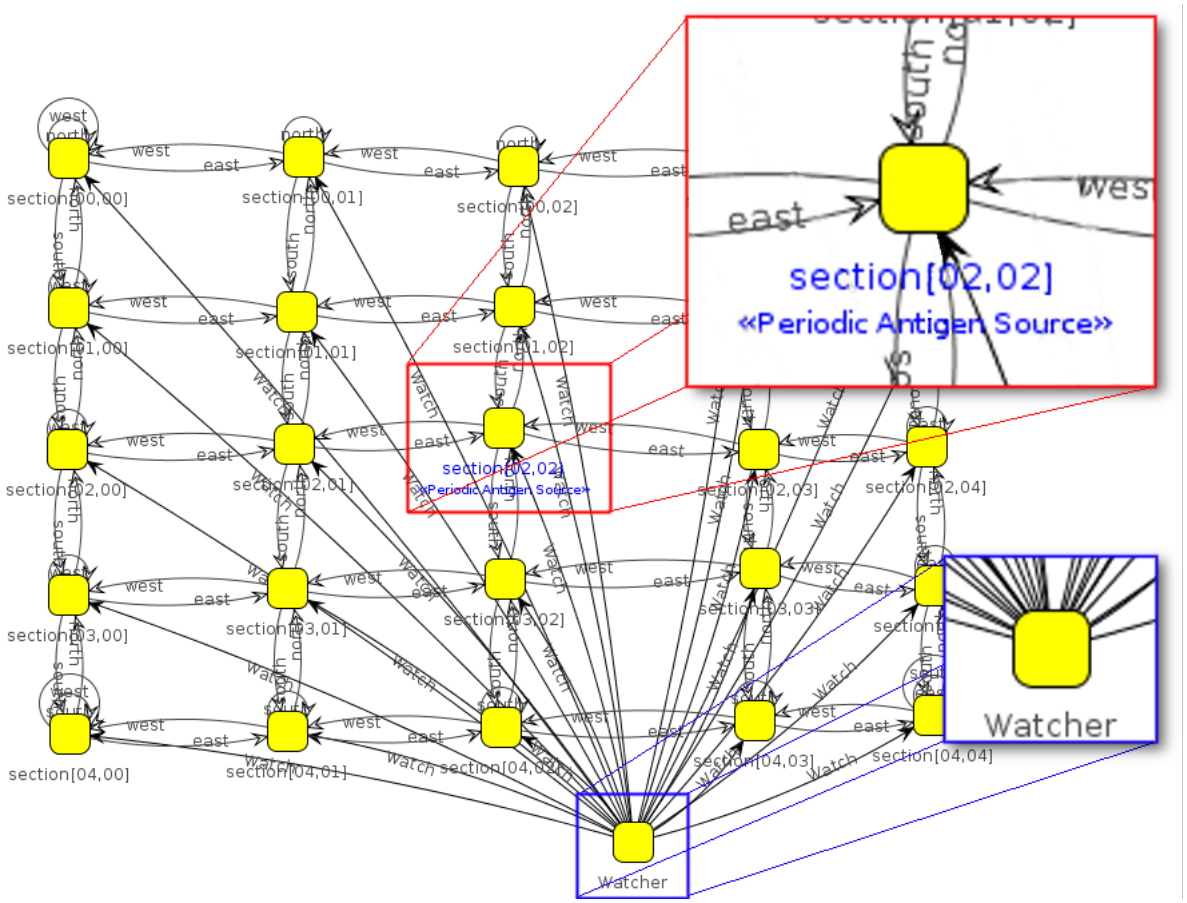


Figura 67 – Modelo de domínio para o sistema imune humano com cenários.

Os três cenários de metamodelagem de Dinâmica de Sistemas foram simulados para explorar o comportamento do modelo quanto à injeção periódica de antígenos. A injeção periódica pode gerar a tolerância ou potencialização da resposta do sistema imune humano.

No primeiro cenário um impulso de 8 LPS é injetado na seção central a cada 21 horas. A Figura 69 apresenta os resultados obtidos quando o cenário de reinjeção periódica está habilitado e dá destaque para os dois primeiros impulsos. A reinjeção conduz a uma nova reação imune para combater a infecção, mas para uma mesma quantidade de antígenos a resposta é diferente devido à presença dos chamados leucócitos residentes, convocados para combater a infecção anterior. A segunda resposta é muito mais forte: a luta contra a população de antígenos é bem mais rápida do que a da primeira.

O mesmo cenário é explorado para um período de tempo maior, 5 dias após a primeira injeção e com o mesmo intervalo entre elas. Todas as injeções posteriores à primeira tiveram apenas uma pequena diminuição no tempo de combate porque a quantidade de leucócitos nos instantes seguintes é bem próxima, contudo apresentando um leve incremento. A Figura 70 mostra os resultados obtidos durante 5 dias de injeção periódica de antígenos.

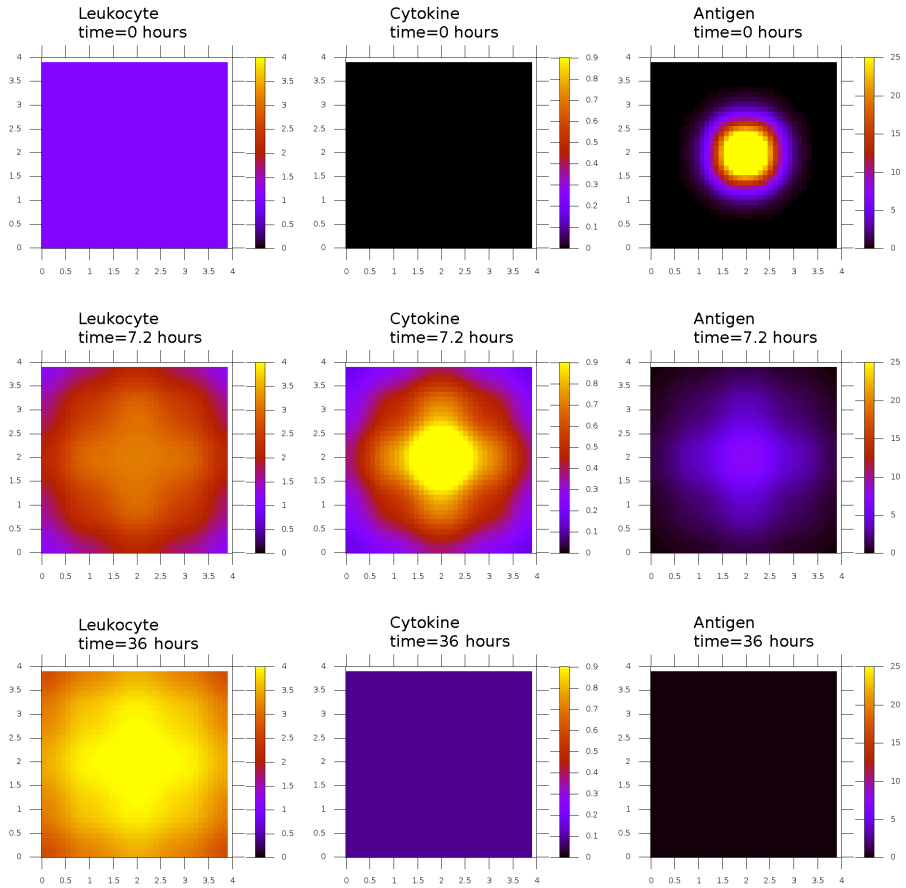


Figura 68 – A dinâmica dos leucócitos (esquerda), citocinas (centro) e antígenos (direita). Cada linha representa três distintos passos do tempo.

Em um segundo experimento foi utilizado o cenário que injeta um impulso de 200 antígenos quando a quantidade de antígenos é bem próxima a zero na seção de tecido. Sempre que a quantidade de antígenos ficar abaixo de $0,15$ $células/mm^3$, ou seja, um combate total à infecção, um novo impulso é gerado. A Figura 71 apresenta os resultados para uma injeção por limiar de antígenos. Nela observa-se que há um conjunto de diferentes intervalos de disparo devido à resposta do sistema. Como no cenário periódico, podemos ver que a segunda injeção é combatida muito mais rapidamente devido aos leucócitos residentes na região infectada. Entretanto, nas injeções seguintes, o tempo de combate à infecção fica bem mais curto, com aproximadamente a mesma duração, devido à quase estabilização da quantidade de leucócitos residentes no tecido.

5.2.4 Análise da reprodutibilidade

O *workflow* de um experimento com o JynaCore API apresenta uma peculiaridade na qual seus modelos podem ser compostos por dois ou mais artefatos. Esses artefatos são relacionados para expor as dependências entre eles, que não são relações de consumo e geração, mas condições fundamentais para a descrição e funcionamento do experimento.

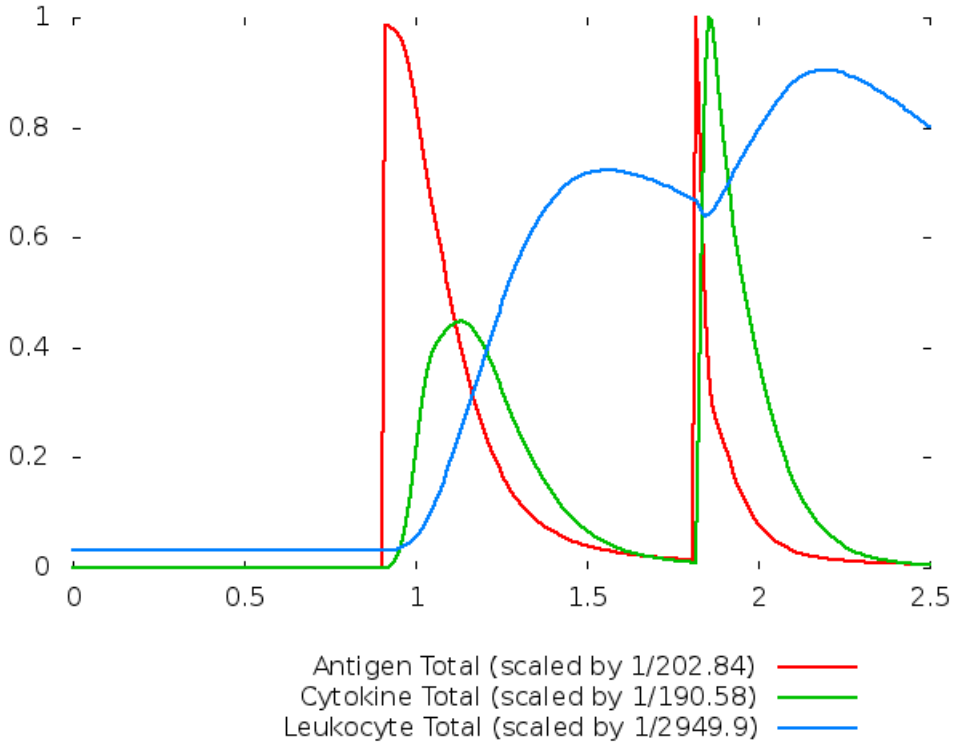


Figura 69 – Resposta do modelo de instância para dois impulsos de antígenos nos primeiros dois dias de infecção.

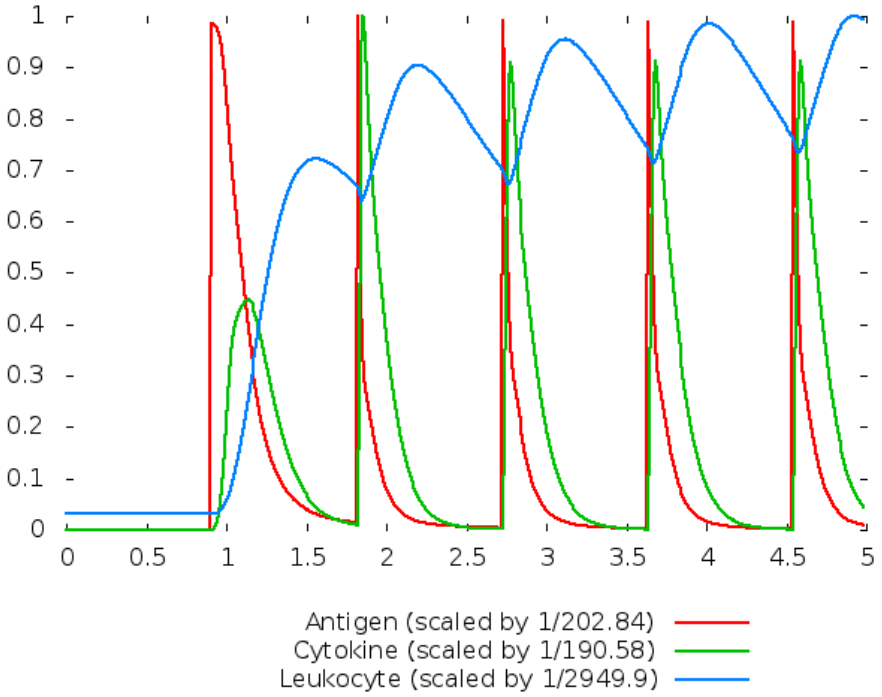


Figura 70 – Resposta de um modelo de instância para uma injeção periódica de antígenos em intervalos de 21 horas durante 5 dias.

Um exemplo de *workflow* para o JynaCore API pode ser visto na Figura 72.

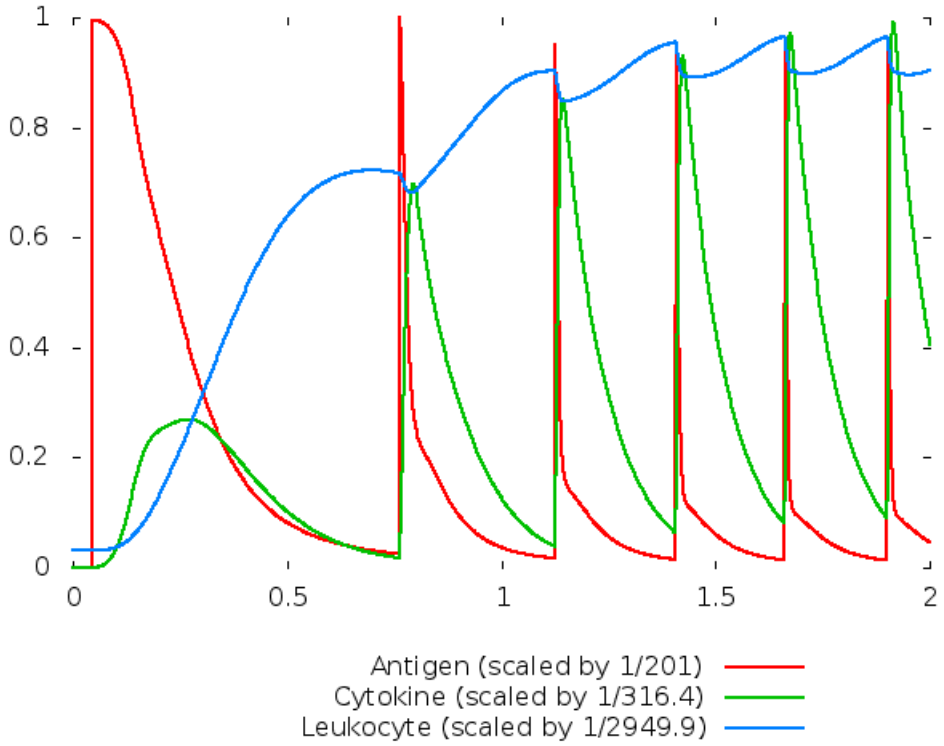


Figura 71 – Resposta do modelo de instância para uma reinjeção de antígenos por limiar: os intervalos de tempo ficam cada vez mais frequentes devido ao combate rápido.

O artefato de sistema de arquivo `ALC-planar-antigen-leucocyte-cytokine -chemiotaxis.jymm` representa o metamodelo de domínio onde as seções de tecido são descritas. Esse artefato é utilizado pelo modelo de instância `ALC-25x25-p2-jymmi` que monta uma malha de 25 por 25 seções de tecido através dos relacionamentos conforme visto na Seção 5.2.3. Adicionalmente, esse modelo de instância referencia dois outros arquivos de cenário, o `ALC-25x25-p2.jymms` e `ALC-25x25-p2b.jymms`. Apesar do nome dos arquivos utilizados, eles não estão relacionados com a dimensão da malha do modelo de instância; eles se conectam a uma seção de tecido definido no modelo de domínio. Portanto, eles também apresentam relações com o modelo de domínio, pois este deve ser fornecido junto dos modelos de cenário.

O artefato de instância é consumido pelo processo `jynacore-runner`, junto de um conjunto de artefatos que serão utilizados como parâmetros para sua execução. Os artefatos texto `variables` e `method` (representandos por ovais no diagrama) descrevem, respectivamente, uma lista de variáveis a serem registradas durante a execução do simulador e qual o método de simulação (atualmente assume apenas os valores `euler` e `rk4` que são os métodos Euler explícito e Runge-Kutta de quarta ordem, implementados nativamente no *jynacoreapi*). Os artefatos circulares `initialTime` e `finalTime` representam dados de ponto flutuante com os limites da simulação e os artefatos `steps` e `skip`, núme-

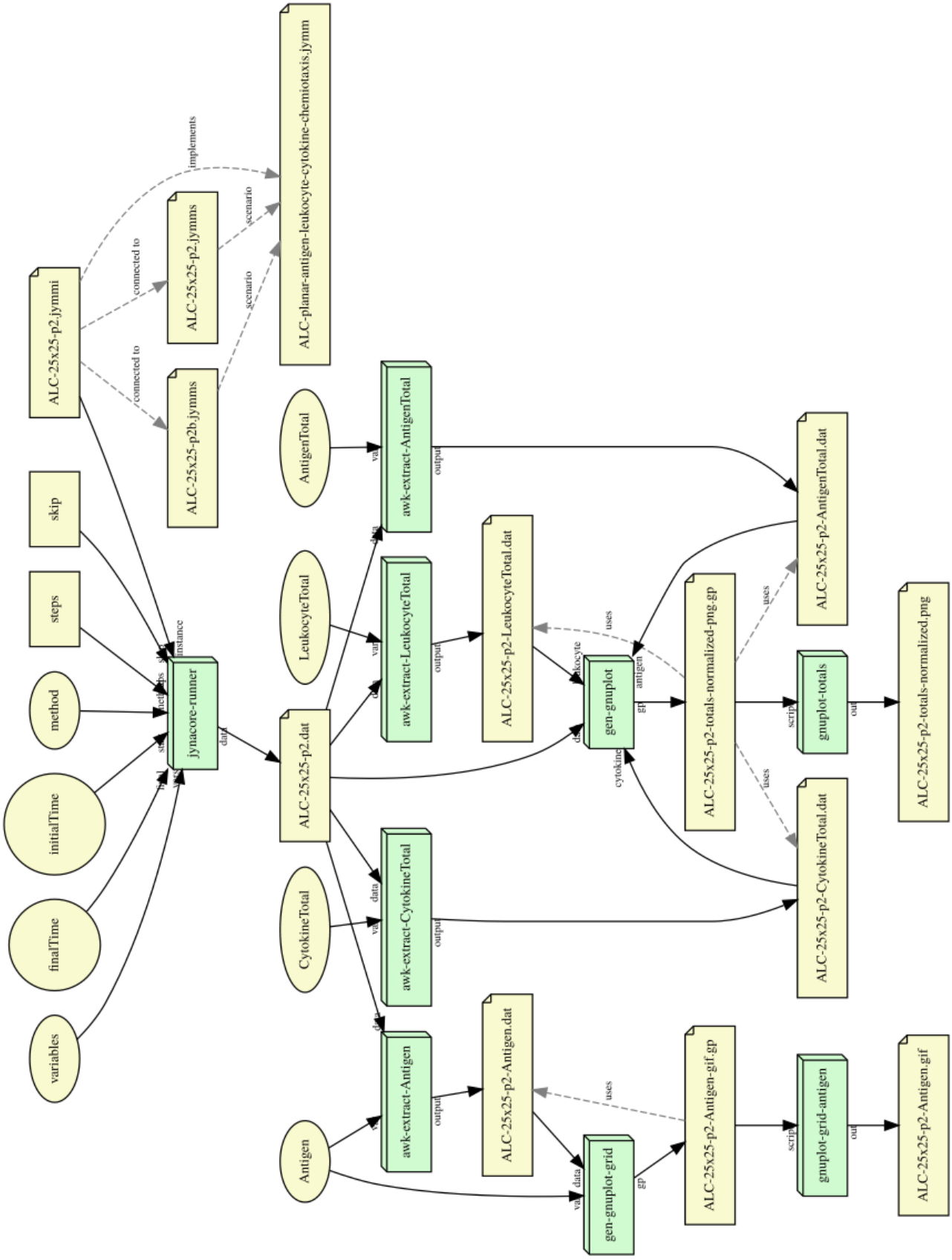


Figura 72 – Fluxo de trabalho para um experimento baseado em metamodelos de Dinâmica de Sistemas com o JynaCore API.

ros inteiros com o número de iterações para os métodos numéricos e o intervalo de registro dos dados de saída. O processo utiliza um programa em linha de comando, desenvolvido com o JynaCore API e implantado ao FISIOENV, que dá como saída um arquivo de dados com o resultado da simulação.

Os dados de saída são consumidos e processados para criar gráficos e análises dos experimentos. Os quatro processos `awk-extract-Antigen`, `awk-extract-AntigenTotal`, `awk-extract-LeucocyteTotal` e `awk-extract-CytokineTotal` são execuções de um mesmo *script* para o programa de linha de comando *Linguagem de* script *AWK* (AWK) integrado ao FISIOENV. Cada processo consume um parâmetro de configuração na forma de um artefato de texto para filtrar os dados de saída da simulação e gerar novos conjuntos de dados que podem ser analisados separadamente. Esses novos conjuntos de dados são consumidos por *scripts* que geram arquivos para o aplicativo GNUPLOT.

Os artefatos `ALC-25x25-p2-totals-normalized-png.gp` e `ALC-25x25-p2-totals-normalized-png.gp` referenciam os conjuntos de dados gerados anteriormente para serem utilizados pelos processos `gnuplot-totals` e `gnuplot-grid-antigen`. Ambos os processos utilizam o *software* GNUPLOT para gerar os artefatos de saída com imagens dos totais (artefato `ALC-25x25-p2-totals-normalized.png`) e uma imagem animada para uma única variável na forma de grade (artefato `ALC-25x25-p2-Antigen.gif`).

Uma análise da reprodutibilidade pode ser realizada para gerar um conjunto de anotações sobre o estado do fluxo de trabalho. A Figura 73 apresenta um relatório simples sobre o fluxo de trabalho quando o mesmo está completo e todos os seus artefatos disponíveis e prontos para uso.

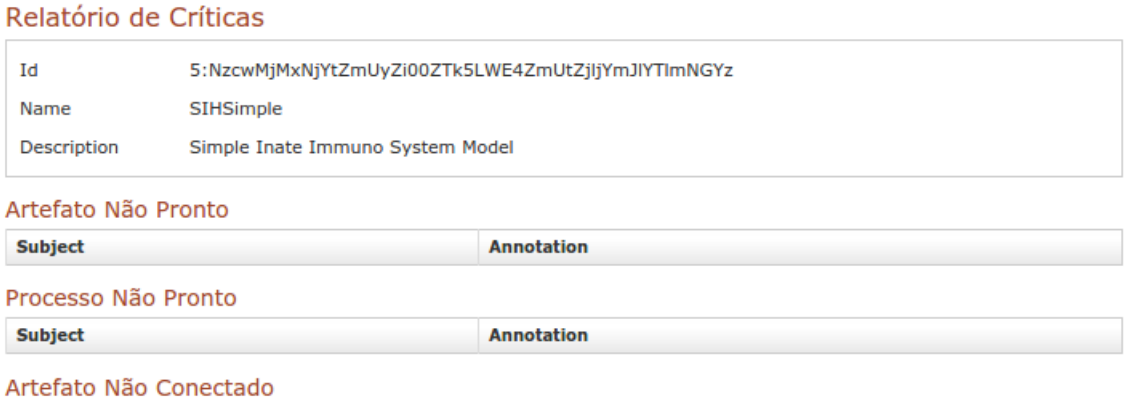


Figura 73 – Relatório de críticas de artefatos quando não há nenhum problema para reprodução do experimento.

Uma falha pode ser inserida artificialmente antes da análise reprodutibilidade para que o relatório capture o problema. No caso da Figura 74, o artefato `ALC-25x25-p2b.jymms` contendo o modelo de cenário foi excluído, deixando, por sua vez, o `ALC-25x25-p2.jymmi` não pronto.

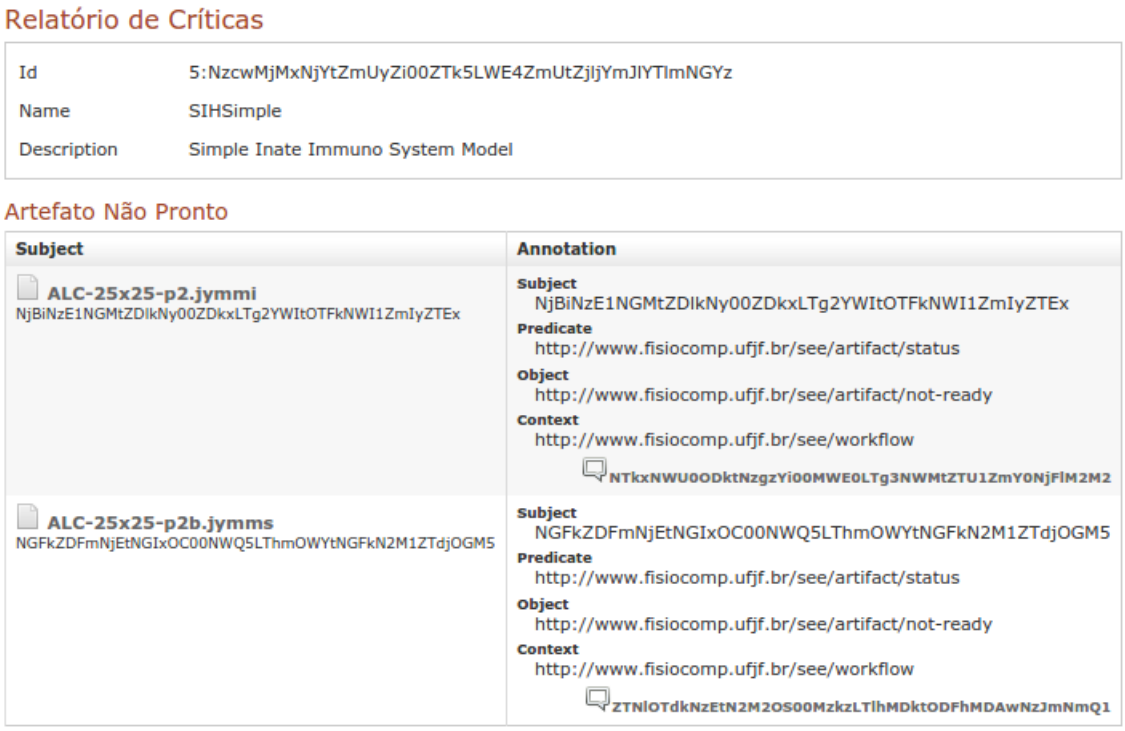


Figura 74 – Relatório de crítica de artefatos no modelo do sistema imune com problemas de reprodução de artefatos quando um artefato de um modelo de cenário necessário para um modelo de instância não está pronto para uso.

Uma falha ainda maior pode ser inserida artificialmente se o artefato de modelo de domínio estiver ausente. Isso inviabilizará os artefatos de cenários e de instância. Conforme é possível visualizar na Figura 75, se o artefato de modelo de domínio `ALC-planar-antigen-leukocite-cytokine-chemiotaxis.jymm` estive ausente, os artefatos dos modelos de cenários que dependem dele (`ALC-25x25-p2b.jymms` e `ALC-25x25-p2.jymms`) e o artefato de modelo de instância `ALC-25x25-p2.jymmi` também não estarão prontos para reprodução.

A análise de reprodutividade entre artefatos e processos continua funcionando como apresentado anteriormente. um processo que não possui os artefatos necessários para sua execução também torna-se não executável. A Figura 76 apresenta o relatório de reprodução de processos quando o artefato `ALC-planar-antigen-leukocite-cytokine-chemiotaxis.jymm` de modelo de domínio não está disponível, inviabilizando sua execução.

A Figura 77 apresenta o relatório de reprodutibilidade sumarizado com os totais de artefatos, processos e programas que impedem a resolução completa do experimento dentro do ambiente quando uma falha foi inserida manualmente. O ambiente só marca o experimento como reprodutível quando não houver nenhum artefato com problemas de reprodução.

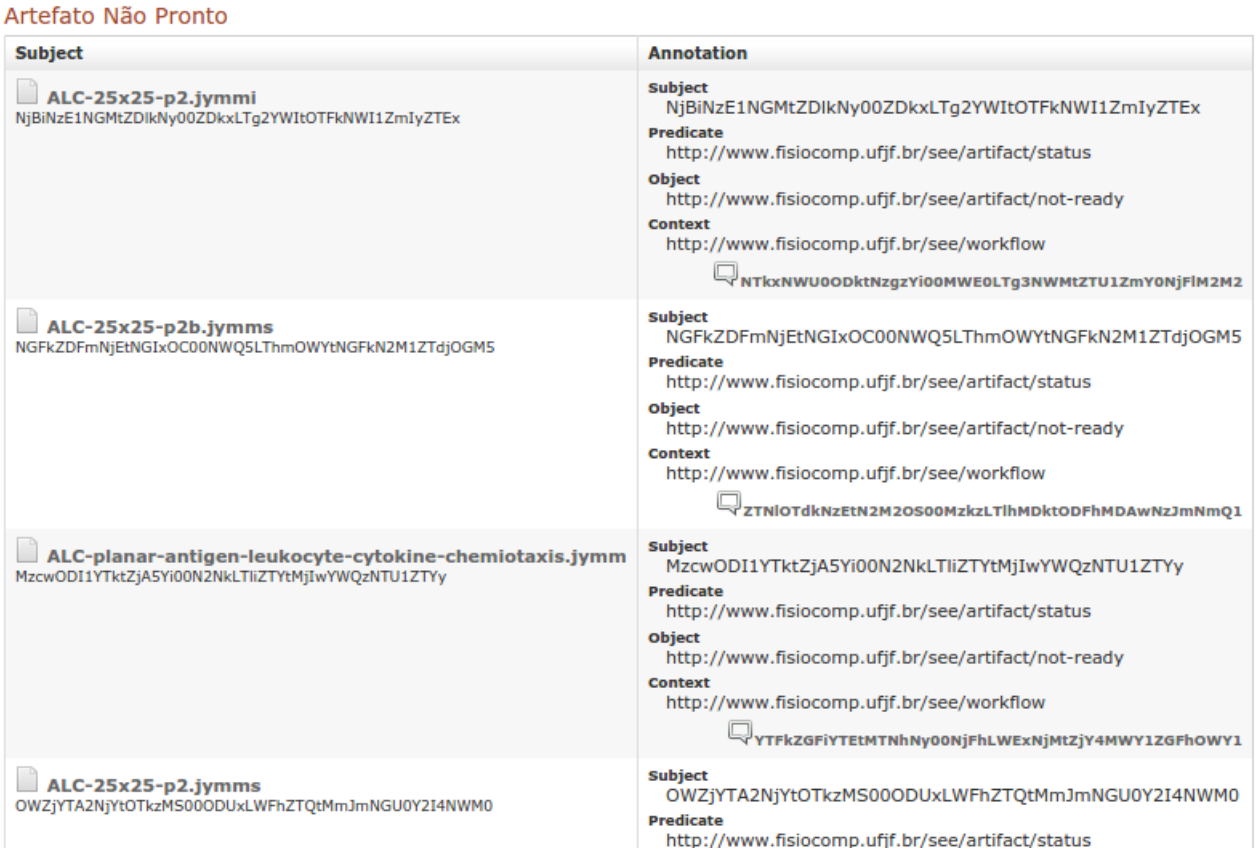


Figura 75 – Relatório de crítica de artefatos no modelo do sistema imune com problemas de reprodução de artefatos quando um artefato de um modelo de domínio necessário para modelos de instância e cenários não está pronto para uso.

Figura 76 – Relatório de crítica de processos no modelo do sistema imune com problemas de reprodução de processos quando um artefato de um modelo de domínio necessário para o processo não está pronto para uso.

5.2.5 Conclusões parciais de Imunologia

Nesta seção foi apresentada abordagem alternativa para a construção de experimentos para o estudo da resposta inata do sistema imune humano utilizando a metamodelagem de Dinâmica de Sistemas. A abordagem se aproveita da reutilização provida pelos dois níveis de abstração, ao separar a construção dos modelos em uma fase de modelagem

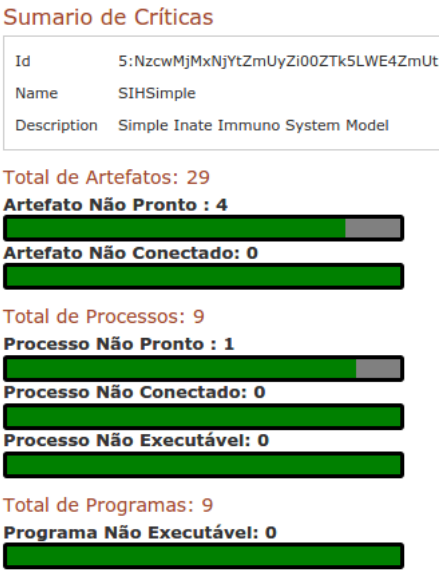


Figura 77 – Relatório de reprodutibilidade compacto exibindo o percentual de artefatos e processos que apresentam problemas para a reprodução do experimento.

do domínio e outra de modelagem de instância. Através dessa abordagem, foi possível descrever um comportamento comum para seções de tecido e, através dos modelos de instância, diversas configurações espaciais puderam ser construídas isoladamente. Alterações no comportamento das seções, como as condições especiais de injeção de antígenos, foram isoladas através do uso de cenários providos pela metamodelagem. A Dinâmica de Sistemas não apresenta construtores próprios para distribuições espaciais e a abordagem utilizando os metamodelos permitiu capturar formalmente o modelo de discretização entre as seções de tecido e explorar os efeitos de difusão e quimiotaxia.

As ferramentas baseadas na JynaCore API apresentam um segundo conjunto restrições para a reprodutibilidade ao serem integradas ao ambiente de simulação: a divisão dos modelos em vários arquivos. Um modelo simulável é uma junção dos modelos de domínio, instância e cenário, e a reprodução dos experimentos deve levar em conta a relação entre os artefatos ao realizar a descrição de experimentos.

Do ponto de vista biológico, é importante destacar que todos os modelos e simulações foram construídos baseados no modelo matemático reduzido de Pigozzo (2011) e apresentaram o mesmo comportamento. Como foram utilizadas apenas três células do SIH, o modelo não apresenta alguns dos efeitos das citocinas anti-inflamatórias, fator de crescimento transformador $\beta 1$ ($TGF - \beta 1$). Além disso, não é feita a distinção entre os leucócitos mais especializados, como macrófagos e neutrófilos que apresentam dinâmicas e comportamentos diferentes durante a resposta imune. Também não existem os efeitos do sistema complemento e do sistema imune adaptativo. Esses modelos mais complexos poderiam lançar mão do mesmo processo de modelagem e respectivas ferramentas, mas ficam fora do escopo desse trabalho.

5.3 Considerações parciais sobre os Estudos de Caso

Este capítulo apresentou estudos de caso onde dois experimentos diferentes foram capturados pelo SEE. O primeiro estudo captura um experimento na área de eletrofisiologia cardíaca, que engloba o uso de modelos e compiladores para se gerar um potencial de ação simulado. O fluxo de trabalho do experimento é repetido e reproduzido dentro do ambiente. Adicionalmente, esse estudo ainda explora as capacidades do ambiente para readequar o fluxo de trabalho previamente definido em uma nova aplicação *web* que permite variar o valor de um parâmetro do modelo e observar seu efeito no potencial de ação. Já o segundo estudo de caso captura um outro experimento, no domínio da imunologia, que utiliza um segundo grupo de ferramentas. Esse experimento explora o uso de metamodelos de dinâmica de sistemas para realizar a simulação da resposta do sistema imune inato. O ambiente, além de replicar seus resultados, demonstrou que ferramentas já implantadas no sistema podem ser reaproveitadas e que podem ser definidas relações complexas entre os artefatos do experimento.

O próximo capítulo apresentada uma discussão sobre as contribuições deste trabalho e como elas se relacionam com outros trabalhos e abordagens.

6 Trabalhos Relacionados

Entre as abordagens para descrição dos experimentos, o MIASE (WALTEMATH et al., 2011a) é um trabalho conceitual que mais se aproxima da construção do modelo de dados do SEEM. O seu foco, assim como o SEEM, também é na definição de um conjunto mínimo de informações necessárias para a reprodução do experimento. Entretanto, enquanto o MIASE foca na descrição passada do experimento, ou seja, na proveniência retrospectiva, o SEEM introduz a ideia de acompanhar ativamente o estado dos elementos do *workflow*. Assim, o SEEM inova ao fornecer uma descrição que associa o estado de um experimento atual com um experimento já publicado, sendo possível fornecer indicações para o pesquisador sobre o que falta ou o que foi alterado em relação à uma execução anterior.

O intercâmbio de experimentos de forma que seu *workflow* completo seja trocado entre programas e ambientes diferentes levou à formação de formatos como o SED-ML (WALTEMATH et al., 2011b). Esse formato codifica a descrição de um experimento em XML e reutiliza outros padrões de modelos já estabelecidos. O SEE previa o uso do SED-ML como formato de persistência e intercâmbio dos modelos. Entretanto, devido à grande variedade de formas que as ferramentas de modelagem utilizam o padrão, optou-se por postergar o seu uso. Este trabalho deu ênfase à reprodução dentro de uma mesma instância do ambiente e entre instâncias diferentes do mesmo, no qual uma interface *web* é disponibilizada aos pesquisadores e estes utilizam as ferramentas integradas ao ambiente. Ainda é de interesse futuro a integração com outras ferramentas de reprodução através da adoção do SED-ML.

O SBSI (ADAMS et al., 2013) é um conjunto de aplicações e bibliotecas na área de biologia sistêmica e sintética que objetiva prover acesso a algoritmos paralelizados e servir de gestor para submissão de trabalhos para conjuntos de máquinas de alto desempenho. Apesar do SBSI e uma instância do SEE se apresentarem como uma infraestrutura para experimento, o SBSI tem o foco em modelos descritos exclusivamente em SBML e para fluxos de trabalho voltados a ajuste de parâmetros de modelos. O SEE não impõe nenhuma restrição ao pesquisador, nem quanto à natureza do experimento, nem quanto à ao uso de modelos ou ferramentas.

Para a descrição dos experimentos, o Repose (GURAVAGE; MERKS, 2011) funciona como um repositório com interface *web* criado à partir das definições do MIASE e SED-ML. Nele os pesquisadores podem criar, editar e compartilhar os detalhes dos experimentos. O Repose tem o mesmo objetivo do SEE de expor os detalhes dos experimentos em um ambiente centralizado. Entretanto, foi idealizado para servir como um “caderno

de bancada" virtual e suas atribuições se restringem a documentar e controlar do fluxo de publicação através de um sistema de gerenciamento de conteúdo. O SEEM prevê que a verificação do estado dos artefatos e programas utilizados na simulação dos modelos façam parte ativa do processo de publicação do experimento, o que difere do Repose, pois ele não realiza a execução de experimentos.

O WOODSS (MEDEIROS et al., 2005) é uma infraestrutura criada inicialmente para planejamento ambiental que foi estendida para descrever e armazenar *workflows* científicos. Nela é possível decompor um *workflow* em elementos reutilizáveis através de descrições semânticas e descrever *workflows* abstratos para experimentos. Através de buscas semânticas *workflows* concretos são criados a partir do repositório e seu fluxo é serializado em WS-BPEL para ser utilizado em alguma ferramenta de orquestração de serviços *web* compatível com o padrão. O WOODSS possui objetivos muito próximos do SEEM, mas foca na composição automática de serviços *web*, ou seja, na descoberta de elementos para a criação dos experimentos. O SEEM, por outro lado, foca do SEEM no acompanhamento da reprodutibilidade de experimentos através do registro e monitoramento do estado dos componentes do experimento. O SEEM também não restringe o pesquisador a utilizar processos baseados em serviços *web*.

O SED-ML *Web Tools* (BERGMANN, 2011) se relaciona com a abordagem aqui proposta por permitir que os pesquisadores submetam arquivos SED-ML contendo experimentos ou que novos experimentos sejam criados utilizando uma interface *web*. O ambiente fornece suporte a modelos SBML ou CELL-ML. Ele difere do trabalho aqui apresentado por não permitir aos pesquisadores compor novos fluxos de trabalho, sendo menos flexível ao exigir que alguns protocolos de experimentação sejam seguidos. O SED-ML *Web Tools* se restringe aos fluxos de trabalho associados às ferramentas previamente integradas.

O projeto Reprozip (CHIRIGATI; SHASHA; FREIRE, 2013) realiza a repetição de experimentos capturados pelo VisTrails. A abordagem permite gerar arquivos compactados com todos os executáveis utilizados durante a execução do experimento e comparar lado a lado os resultados de execuções subsequentes. A abordagem difere da adotada pelo SEEM por exigir que o experimento seja desde o início desenvolvido dentro do Vistrails e em um sistema operacional idêntico ao original. Apesar da implementação do SEEM também assumir que experimentos são fortemente ligados ao sistema operacional no qual foram desenvolvidos, há o destaque dos elementos que devem ser substituídos, caso surja alguma incompatibilidade quando trocados entre instâncias de ambiente diferentes ou caso não estejam mais disponíveis. Novamente deve-se destacar que pelo SEEM, o ambiente que tem a responsabilidade de prover a infraestrutura a um grupo de experimentos, devendo, inclusive, fornecer diferentes instâncias para grupos muito heterogêneos entre si.

O projeto ResearchCompedia.org (STODDEN; MIGUEZ; SEILER, 2015) é uma

infraestrutura que permite a pesquisadores publicar seus artigos, dados e *software* necessários para reproduzir os resultados de seus experimentos. O objetivo final de se obter a reprodução dos experimentos é equiparável ao do SEEM. Entretanto, é necessário que uma equipe conduza uma execução do código para realizar a certificação do experimento junto ao artigo. Isso dá ao projeto uma ênfase na publicação dos chamados "artigos executáveis". Apesar dos autores proporem a execução através de virtualização na nuvem, o enfoque é mais na repetição dos experimentos. O SEEM prevê que a captura do fluxo de trabalho deve ser exposta formalmente para permitir a posterior reprodução e readaptação dos experimentos. Portanto, isso vai mais além do que prover a repetição e validação dos resultados com as publicações.

A Tabela 5 sumariza as características das abordagens quanto a, respectivamente, se estão disponíveis em uma versão *web*, se podem ser utilizadas como um método isolado, se são de uso geral, se permitem a execução do experimento e qual o formato de reprodução que apresentam.

Tabela 5 – Comparação entre as características das abordagens com o SEEM e SEE.

Projeto	Web	Método	Uso Geral	Execução	Reprodutibilidade
MIASE (2011)	-	Sim	Sim	-	Descritiva
SED-ML (2011)	-	Não	Sim	-	Descritiva
SBSI (2013)	Não	Não	Não	Sim	Ativa
WOODS (2005)	Sim	Sim	Parcial	Não	Descritiva
REPOSE (2011)	Sim	Não	-	Não	Descritiva
SED-ML Web Tools (2011)	Sim	Não	Parcial	Sim	Ativa
Reprozip (2013)	Não	Não	Não	Sim	Ativa
ResearchCompendia.org (2015)	Sim	Não	Sim	Certificada	Descritiva/Assistida
SEEM+SEE	Sim	Sim	Sim	Sim	Ativa

Fonte: Produzido pelos autores

7 Considerações Finais

A biologia sistêmica busca aproveitar os últimos resultados da genômica funcional para criar modelos dinâmicos de sistemas biológicos e estudar os inter-relacionamentos e comunicações de sinais entre esses para melhorar os tratamentos e descoberta de novas drogas para combate à doenças. Entretanto, a grande produção de modelos e *software* com o propósito de dar suporte a esses estudos acaba por se perder após a publicação pela falta ou dificuldade de se documentar e repetir experimentos *in silico*.

Neste trabalho foi investigado o estado da arte das técnicas, ferramentas e padrões de intercâmbio utilizados na formulação de experimentos de modelagem computacional da biologia sistêmica. Foi proposto um novo método para integrar ferramentas e descrever experimentos *in silico*, de forma a permitir a sua repetição e reprodução. Essa nova abordagem acolhe ferramentas legadas e experimentais, sem exigir que se utilizem modelos e ferramentas diferentes das utilizadas pela pesquisa original. O fluxo de trabalho de um experimento *ad hoc* é capturado, anotado e compartilhado através de um ambiente que se adapta para integrar as ferramentas e expõe um série de problemas de funcionamento. Posteriormente, novas instâncias de ambientes podem ser criadas e os fluxos de trabalho podem ser trocados entre elas, dando prioridade para a adoção de padrões bem estabelecidos pela comunidade de modelagem.

Para evidenciar a abordagem, foi desenvolvido um ambiente computacional, o FISIOENV, que funciona como ambiente de descrição e simulação de *workflows* de experimentos *in silico* e, sobre ele, o SEE, um ambiente computacional compatível com o método proposto, que incentiva o registro e anotações dos experimentos. Um experimento bem documentado tem todos os seus programas implantados, todos os processos associados a artefatos de entrada, saída e respectivos programas, gerando a composição dos *workflows*. Os experimentos ficam registrados de sua concepção até o compartilhamento resultados encontrados.

Dois estudos de caso no domínio de biologia sistêmica foram desenvolvidos como aplicação para a abordagem proposta. Cada um utilizou um conjunto de modelos e ferramentas distintos e teve seus *workflows* integrados ao ambiente. O AGOS realiza a importação de modelos no formato CELL-ML e Mathematics Markup Language (MathML) na área de eletrofisiologia cardíaca e tem seu funcionamento altamente dependente do ambiente. Sua integração ao ambiente permitiu ainda mostrar o uso da abordagem para o reuso de fluxos de trabalho para a construção de aplicações para o usuário final, o AGOSWeb. O JynaCore API, que lida com modelos descritos por metamodelos de Dinâmica de Sistemas, foi empregado para criar experimentos na área de modelagem do sistema imune

humano. Esse último trabalho aplica os metamodelos de dinâmica de sistemas para problemas de diferenças finitas e serviu como exemplo de captura de fluxos de trabalho em que há uma composição de modelos em diversos arquivos diferentes.

A abordagem utilizou uma abstração para os experimentos na forma de grafos direcionados compostos por processos que consomem e geram artefatos. Ao organizar esse fluxo de consumo e geração de artefatos, define-se um fluxo de trabalho que pode ser executado pelo usuário através de uma interface *web*. A infraestrutura se mostrou capaz de encapsular ferramentas computacionais existentes, como compiladores, simuladores e ferramentas de pós-processamento de dados. A abordagem é genérica e não restringe o seu uso a uma implementação específica, podendo ser aplicada a outros motores de composição e execução de *workflows*.

7.1 Limitações da Pesquisa

Este trabalho procurou apresentar uma abordagem que englobasse todas as fases do ciclo de vida de um experimento *in silico*. Entretanto, devido a limitações de tempo e recursos, foi necessário dar maior ênfase a algumas fases em detrimento de outras. Algumas das limitações impostas nesta pesquisa dão subsídios para a formulação de sugestões para trabalhos futuros.

O trabalho se limitou ao estudo de fluxos de trabalho de experimentos computacionais *ad hoc*, que podem ser realizados como processos isolados em um computador. O uso de grades computacionais, virtualização e computação em nuvem são de grande interesse para área de pesquisa de *workflows* científicos e a ciência em larga escala mas não foram abordados neste trabalho. Entretanto, a abordagem aqui proposta permite que novos motores de execução e acompanhamento de serviços remotos possam ter seu funcionamento registrado estendendo os construtores básicos e anotações.

Várias das anotações utilizadas no trabalho possuem identificadores que fornecem um contexto semântico dentro do método e são utilizadas para análise e decisões dentro do ambiente computacional implementado. Apesar de ter sido definida uma taxonomia inicial para guiar a definição das anotações do método, não foi criada uma ontologia para os conceitos dos métodos. Atualmente, as críticas estão sendo realizadas por inferências inseridas no código do ambiente.

7.2 Contribuições Alcançadas

As contribuições deste trabalho estão distribuídas em vários contextos diferentes mas podem ser resumidas em:

- um novo método para descrição de experimentos *in silico* e armazenamento de seus resultados, o SEEM;, com o objetivo de guiar sua reprodutibilidade;
- uma implementação prova de conceito de arquitetura que realiza a composição e execução de *workflows* científicos que atende aos requisitos do método proposto;
- um estudo de caso na área de eletrofisiologia, onde uma ferramenta tem seu fluxo de trabalho capturado de forma reprodutível;
- a readequação de um fluxo de trabalho capturado pelo ambiente como uma aplicação *web* para construção de novos experimentos na área de eletrofisiologia cardíaca;
- um estudo de caso na área de imunologia onde uma ferramenta computacional tem seu fluxo de trabalho capturado de forma reprodutível;
- um estudo no qual o fluxo de trabalho capturado pelo ambiente é utilizado para criar uma nova aplicação dos metamodelos de dinâmica de sistemas para a modelagem do sistema imune.

7.3 Trabalhos Futuros

Pela característica interdisciplinar, este trabalho abre várias oportunidades para estudos subsequentes em mais de um domínio de conhecimento. O ciclo de vida completo de um experimento é capturado pelo método, e respectivo ambiente computacional, de forma que em cada fase é possível listar pontos onde podem ser realizados mais trabalhos.

O motor de execução de *workflows* do FISIOENV assume que o usuário está no completo controle da escolha da ordem de execução dos processos. Apesar de fornecer meios para que ele automatize sua execução, através do uso de composição de processos, mais estudos podem ser realizados para a execução automática, escalonamento e paralelismo do *workflow*. A ordem e priorização da execução pode ser obtida pela análise do grafo do *workflow* e anotações extras podem auxiliar o processo.

As anotações desenvolvidas neste trabalho são flexíveis a ponto de permitir desde uma entrada informal, fornecida pelo pesquisador, até anotações semânticas formais. Apesar da semântica ser utilizada para inferências internas na execução do ambiente, a definição de uma ontologia para capturar os conceitos do método aliada a uma maior integração com motores de inferência pode contribuir para uma maior flexibilização para os resultados de reprodução de experimentos.

Os modelos utilizados são tratados pelo SEEM como artefatos “opacos” e o fluxo do trabalho é gerido de forma transparente para permitir maior flexibilidade do ponto de vista de reprodução do experimento como um todo. Outras abordagens, em contrapartida, invertem essa visibilidade, se baseando fortemente nas estruturas dos modelos

mas ofuscam o uso das ferramentas do fluxo de trabalho dos experimentos. Uma possível continuação para melhoria do método é a criação de mais uma camada de extração ou tradução dos conceitos de reprodução contidos nos modelos, na forma de metadados e anotações, para auxiliar na avaliação da reprodução do experimento. Assim seria possível explicitar e analisar o efeito da propagação de parâmetros dos modelos entre experimentos derivados.

O método propõe uma série de relatórios e métricas para para o acompanhamento da reprodutibilidade dos *workflows* dentro do ambiente, mas novas podem ser criadas. Adicionalmente, a análise dos inter-relacionamentos entre métricas de diferentes contextos podem levar a um maior entendimento sobre os experimentos. Novas métricas podem capturar o impacto do uso de ferramentas mais restritivas, seja em arquitetura, seja por licenciamento de *software* em experimentos científicos. Uma análise das restrições pode criar vários pontos de tomada de decisão de como conduzir trabalhos de pesquisa subsequentes ao criar uma estimativa de esforço necessário para se chegar nos mesmos resultados anteriores.

A coleta e análise dos dados de proveniência também podem fornecer pontos importantes sobre os métodos utilizados pelos pesquisadores. Experimentos com mesmo objetivo, mas fluxos de trabalhos diferentes podem ser comparados para se avaliar a eficácia de métodos e ferramentas. O grupo de pesquisa pode conseguir medir, avaliar e isolar pontos onde esforços devem ser concentrados a fim de melhorar a produtividade acadêmica ou técnica.

Referências

AALST, W. M. Van der. The application of Petri nets to workflow management. *Journal of circuits, systems, and computers*, World Scientific, v. 8, n. 01, p. 21–66, 1998. 42

ABDEL-HAMID, T.; MADNICK, S. E. *Software project dynamics: an integrated approach*. [S.l.]: Prentice-Hall, Inc., 1991. ISBN 0-13-822040-9. 35

ADAMS, R. et al. SBSI: an extensible distributed software infrastructure for parameter estimation in systems biology. *Bioinformatics*, Oxford Univ Press, v. 29, n. 5, p. 664–665, 2013. 22, 113

AKRAM, A.; MEREDITH, D.; ALLAN, R. Evaluation of BPEL to scientific workflows. v. 1, p. 269–274, 2006. 43

ALBERTS, B. et al. *Essential cell biology*. [S.l.]: Garland Science, 2013. 27

APACHE SOFTWARE FOUNDATION. *Apache Tomcat*. 2016. Disponível em: <http://tomcat.apache.org/>. Acesso em: 12 jan. 2013. 71

AT&T. *Graphvis - Graph Visualization Software*. 2016. Disponível em: <http://http://www.graphviz.org//>. Acesso em: 15 de jan. 2012. 72

BAETEN, J. C. A brief history of process algebra. *Theoretical Computer Science*, Elsevier, v. 335, n. 2, p. 131–146, 2005. 42

BARBOSA, C. B. et al. A transformation tool for ode based models. In: *Computational Science-ICCS 2006*. [S.l.]: Springer, 2006. p. 68–75. 81

BARGA, R.; GANNON, D. *Scientific versus Business Workflows*. [S.l.]: M. Shields, 2007. ISBN 978-1-84628-519-6. 41

BARLAS, Y. System dynamics: systemic feedback modeling for policy analysis. *SYSTEM*, v. 1, p. 59, 2007. 35

BARROS, B. G. d. *Simulações computacionais de arritmias cardíacas em ambientes de computação de alto desempenho do tipo Multi-GPU*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, 2013. 29

BARROS, M. d. O. *Gerenciamento de Projetos Baseado em Cenários: Uma Abordagem de Modelagem Dinâmica e Simulação*. Tese (Doutorado) — COPPE/UFRJ, Rio de Janeiro, 2001. 19, 37, 81

BARSEGHIAN, D. et al. Workflows and extensions to the kepler scientific workflow system to support environmental sensor data access and analysis. *Ecological Informatics*, Elsevier, v. 5, n. 1, p. 42–50, 2010. 43

BAUER, C.; KING, G. *Java Persistance with Hibernate*. [S.l.]: Dreamtech Press, 2006. 71

BERGMANN, F. *JLibSEDML .NET Resources for SED-ML*. 2014. Disponível em: <https://sourceforge.net/projects/jlibsedml/>. Acesso em: 10 nov. 2015. 22

BERGMANN, F. *LibSEDML Java Resources for SED-ML*. 2014. Disponível em: <https://sourceforge.net/projects/jlibsedml/>. Acesso em: 10 nov. 2015. 22

BERGMANN, F. T. SED-ML Web Tools. Nature Publishing Group, 2011. 22, 114

BONDARENKO, V. E. A compartmentalized mathematical model of the β1-adrenergic signaling system in mouse ventricular myocytes. *PloS one*, Public Library of Science, v. 9, n. 2, 2014. 30

BONDARENKO, V. E. et al. Computer model of action potential of mouse ventricular myocytes. *American Journal of Physiology-Heart and Circulatory Physiology*, Am Physiological Soc, v. 287, n. 3, p. H1378–H1403, 2004. 30, 35, 85, 87

BOX, D. et al. *Simple Object Access Protocol (SOAP) 1.1*. 2000. 72

BOYCE, W. E.; DIPRIMA, R. C.; HAINES, C. W. *Elementary differential equations and boundary value problems*. [S.l.]: Wiley New York, 1992. v. 9. 34, 35

BRAY, T. et al. Extensible markup language (XML) 1.1 (Second Edition). *World Wide Web Journal*, 2006. 21

CALLAHAN, S. P. et al. VisTrails: visualization meets data management. In: ACM. *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. [S.l.], 2006. p. 745–747. 23

CAMPAGNE, F. et al. Quantitative information management for the biochemical computation of cellular networks. *Science Signaling*, AAAS, v. 2004, n. 248, p. pl11, 2004. 17

CAMPOS, D. R. *Alterações eletromecânicas do miócito cardíaco na fase aguda da doença de Chagas em modelo murino: papel do óxido nítrico, intereferon-gama e ânion superóxido*. Tese (Doutorado) — UFMG, 2011. 30, 31, 91

CAMPOS, F. O. *Modelagem computacional da eletrofisiologia cardíaca: O desenvolvimento de um novo modelo para células de camundongos e avaliação de novos esquemas numéricos*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, 2008. 28, 30

CHIRIGATI, F.; SHASHA, D.; FREIRE, J. Reprozip: Using provenance to support computational reproducibility. In: *Presented as part of the 5th USENIX Workshop on the Theory and Practice of Provenance*. [S.l.: s.n.], 2013. 23, 114

DAY, J. et al. A reduced mathematical model of the acute inflammatory response ii. capturing scenarios of repeated endotoxin administration. *Journal of theoretical biology*, Elsevier, v. 242, n. 1, p. 237–256, 2006. 90

DEELMAN, E. et al. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming*, Hindawi Publishing Corporation, v. 13, n. 3, p. 219–237, 2005. 43

DIFRANCESCO, D.; NOBLE, D. A model of cardiac electrical activity incorporating ionic pumps and concentration changes. *Philosophical Transactions of the Royal Society B: Biological Sciences*, The Royal Society, v. 307, n. 1133, p. 353–398, 1985. 30

DRUMMOND, C. Replicability is not reproducibility: nor is it good science. 2009. 44

DUMAS, M.; HOFSTEDE, A. H. T. UML activity diagrams as a workflow specification language. In: *UML 2001 - The Unified Modeling Language. Modeling Languages, Concepts, and Tools*. [S.l.]: Springer, 2001. p. 76–90. 42

FAHRINGER, T. et al. *Workflows for eScience, Scientific Workflows for Grids, chapter ASKALON: A Development and Grid Computing Environment for Scientific Workflows*. [S.l.]: Springer Verlag, 2007. 42

FAWCETT, D. W.; MCNUTT, N. S. The ultrastructure of the cat myocardium i. ventricular papillary muscle. *The Journal of Cell Biology*, Rockefeller Univ Press, v. 42, n. 1, p. 1–45, 1969. 26, 27

FITZHUGH, R. Mathematical models of threshold phenomena in the nerve membrane. *The bulletin of mathematical biophysics*, Springer, v. 17, n. 4, p. 257–278, 1955. 82

FOMEL, S.; HENNENFENT, G. Reproducible computational experiments using scons. In: *ICASSP (4)*. [S.l.: s.n.], 2007. p. 1257–1260. 42

FOMEL, S. et al. Madagascar: open-source software project for multidimensional data analysis and reproducible computational experiments. *Journal of Open Research Software*, v. 1, n. 1, p. e8, 2013. 42

FORD, F. A. *Modeling the environment: an introduction to system dynamics models of environmental systems*. [S.l.]: Island Press, 1999. 35

FORRESTER, J. W. *Industrial dynamics*. [S.l.]: The M.I.T. Press, 1961. ISBN 0-262-56001-1. 35

FOSTER, I. Service-oriented science. *Science*, American Association for the Advancement of Science, v. 308, n. 5723, p. 814–817, 2005. 17

GARNY, A. et al. *CellML and associated tools and techniques*. 2008. 19, 21, 81

GLEESON, P. et al. NeuroML: a language for describing data driven models of neurons and networks with a high degree of biological detail. *PLoS computational biology*, Public Library of Science, v. 6, n. 6, p. e1000815, 2010. 17

GRAILS PROJECT. *Grails Framework*. 2016. Disponível em: <http://grails.org>. Acesso em: 02 fev. 2016. 71

GROOVY PROJECT. *Groovy Language*. 2016. Disponível em: <http://www.groovy-lang.org/>. Acesso em: 02 fev. 2016. 71

GURAVAGE, M. A.; MERKS, R. A web-based repository of reproducible simulation experiments for systems biology. In: *SIMULTECH*. [S.l.: s.n.], 2011. p. 134–141. 22, 113

HAGIWARA, N.; IRISAWA, H.; KAMEYAMA, M. Contribution of two types of calcium currents to the pacemaker potentials of rabbit sino-atrial node cells. *The Journal of physiology*, Wiley Online Library, v. 395, n. 1, p. 233–253, 1988. 26

HEATH, M. T. *Scientific computing*. [S.l.]: McGraw-Hill New York, 1997. 98

HIBERNATE COMMUNITY. *Hibernate Object/Relational Mapping framework*. 2016. Diponível em: <http://hibernate.org/orm/>. Acesso em: 20 dez. 2015. 71

HODGKIN, A. L.; HUXLEY, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, Wiley Online Library, v. 117, n. 4, p. 500–544, 1952. 29, 34, 35

HSQL DEVELOPMENT GROUP. *Hyper SQL Database*. 2016. Disponivel em: <http://www.hslqdb.org/>. Acesso em: 20 dez. 2015. 71

HUCKA, M. et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, Oxford Univ Press, v. 19, n. 4, p. 524–531, 2003. 17, 22, 23

HULL, D. et al. Taverna: a tool for the composition and enactment of bioinformatics workflow. *Bioinformatics*, v. 20, n. 3045, p. C3054, 2004. 42

INGALLS, B. P. *Mathematical modeling in systems biology*. [S.l.]: MIT Press Harvard, MA, 2013. 34

JANEWAY, C. A. et al. *Immunobiology: The Immune System (Janeway)*. 7ed.. ed. [S.l.]: Garland Science, 2010. ISBN 0815341237. 31

JARRARD, R. D. *Scientific methods*. [S.l.: s.n.], 2001. Disponível em: <http://emotionalcompetency.com/sci/booktoc.html>. Acesso em: 09 jun. 2014. 34

KAIL, E.; KACSUK, P.; KOZLOVSZKY, M. A novel approach to user-steering in scientific workflows. In: IEEE. *Applied Computational Intelligence and Informatics (SACI), 2015 IEEE 10th Jubilee International Symposium on*. [S.l.], 2015. p. 233–236. 56

KEENER, J. P.; SNEYD, J. *Mathematical physiology*. [S.l.]: Springer, 1998. v. 8. 82

KELLNER, M.; R, M.; D, R. Software process simulation modeling: Why? what? how? *Journal of Systems and Software*, 1999. 35

KEPLER PROJECT. *KEPLER - Scientific Workflows and Process Networks*. 2004. Disponível em: <http://kepler-project.org>. Acesso: 24 Mar 2009. 42, 43

KESELER, I. M. et al. Ecocyc: a comprehensive database resource for escherichia coli. *Nucleic acids research*, Oxford Univ Press, v. 33, n. suppl 1, p. D334–D337, 2005. 17

KITANO, H. Computational systems biology. *Nature*, Nature Publishing Group, v. 420, n. 6912, p. 206–210, 2002. 32, 33

KNOP, I. *Infraestrutura para Simulação de Processos de Software Baseada em Metamodelos de Dinâmica de Sistemas*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, Juiz de Fora, 2011. 21, 81, 89, 94

KNOP, I. et al. A New Web-Based Integration Tool for the Development of in Silico Experiments of Cardiac Electrophysiology. In: SPRINGER. *5th European Conference of the International Federation for Medical and Biological Engineering*. [S.l.], 2012. p. 355–358. 21, 24, 71, 81

KNOP, I. et al. Modeling Human Immune System Using a System Dynamics Approach. In: SPRINGER. *5th European Conference of the International Federation for Medical and Biological Engineering*. [S.l.], 2012. p. 363–366. 21, 32, 81

KNOP, I. et al. System dynamics metamodels supporting the development of computational models of the human innate immune system. In: *Computational Science and Its Applications-ICCSA 2012*. [S.l.]: Springer, 2012. p. 707–722. 21, 32, 81, 90

KOJIRI, T. et al. World continental modeling for water resources using system dynamics. *Physics and Chemistry of the Earth, Parts A/B/C*, Elsevier, v. 33, n. 5, p. 304–311, 2008. 35

KOOP, D. et al. A provenance-based infrastructure to support the life cycle of executable papers. *Procedia Computer Science*, Elsevier, v. 4, p. 648–657, 2011. 42

LASZEWSKI, G. von; HATEGAN, M.; KODEBOYINA, D. *Workflows for e-Science Scientific Workflows for Grids*. [S.l.]: Springer, 2007. 17

Workflow handbook 1997. In: LAWRENCE, P. (Ed.). New York, NY, USA: John Wiley & Sons, Inc., 1997. cap. The WfMC Glossary, p. 385–421. ISBN 0-471-96947-8. 18, 41

LEVEQUE, R. J. Wave propagation software, computational science, and reproducible research. In: *Proc. Int. Congr. of Mathematicians*. [S.l.: s.n.], 2006. 44

LLOYD, C. M.; HALSTEAD, M. D.; NIELSEN, P. F. Cellml: its future, present and past. *Progress in biophysics and molecular biology*, Elsevier, v. 85, n. 2, p. 433–450, 2004. 17

LUDÄSCHER, B. et al. *Scientific Workflows: More e-Science Mileage from Cyberinfrastructure.* 2006. 17, 41, 42

MARWICK, B. Computational reproducibility in archaeological research: Basic principles and a case study of their implementation. *Journal of Archaeological Method and Theory*, Springer, p. 1–27, 2016. 45

MATTOSO, M. et al. User-steering of HPC workflows: state-of-the-art and future directions. In: ACM. *Proceedings of the 2nd ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*. [S.l.], 2013. p. 4. 56

MATTOSO, M. et al. Towards supporting the life cycle of large scale scientific experiments. *International Journal of Business Process Integration and Management*, Inderscience Publishers, v. 5, n. 1, p. 79–92, 2010. 18, 42

MCGOUGH, S. et al. Workflow enactment in ICENI. In: *UK e-Science All Hands Meeting*. [S.l.: s.n.], 2004. p. 894–900. 42

MEDEIROS, C. B. et al. WOODSS and the Web: annotating and reusing scientific workflows. *ACM SIGMOD Record*, ACM, v. 34, n. 3, p. 18–23, 2005. 22, 114

MEDEIROS, C. B.; VOSSEN, G.; WESKE, M. Wasa: A workflow-based architecture to support scientific database applications. In: SPRINGER. *Database and Expert Systems Applications*. [S.l.], 1995. p. 574–583. 33

MEDINA, M. Á. Systems biology for molecular life sciences and its impact in biomedicine. *Cellular and Molecular Life Sciences*, Springer, p. 1–19, 2013. 17

MESIROV, J. P. Computer science. accessible reproducible research. *Science (New York, NY)*, NIH Public Access, v. 327, n. 5964, 2010. 43

MUETZELFELDT, R. Extended system dynamics modelling of the impacts of food system drivers on food security, livelihoods and the environment. *Report for the CGIAR Research Program on Climate Change, Agriculture and Food Security (CCAFS)*, 2010. 35

NICKERSON, D.; YOU, D. A. Extending libSedML to support CellML models. Nature Publishing Group, 2011. 23

NIEDERER, S.; SMITH, N. At the heart of computational modelling. *The Journal of physiology*, Wiley Online Library, v. 590, n. 6, p. 1331–1338, 2012. 30

NOBLE, D. The surprising heart: a review of recent progress in cardiac electrophysiology. *The Journal of Physiology*, Wiley Online Library, v. 353, n. 1, p. 1–50, 1984. 26

NOVERE, N. L. et al. Biomodels database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. *Nucleic acids research*, Oxford Univ Press, v. 34, n. suppl 1, p. D689–D691, 2006. 17

NOVERE, N. L. et al. Minimum information requested in the annotation of biochemical models (MIRIAM). *Nature biotechnology*, Nature Publishing Group, v. 23, n. 12, p. 1509–1515, 2005. 17

OASIS. *WS-BPEL: Web Services Business Process Execution Language Version 2.0*. 2007. Disponível em: <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>. Acesso em: 14 abr. 2015. 22

OGASAWARA, E. et al. Exploring many task computing in scientific workflows. In: ACM. *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*. [S.l.], 2009. p. 2. 42

OGASAWARA, E. S. *Uma Abordagem Algébrica para Workflows Científicos com Dados em Larga Escala*. Tese (Doutorado) — COPPE/UFRJ, Rio de Janeiro, 2011. 56

OLIVEIRA, A. H. M. de et al. Reprodução de experimentos científicos usando nuvens. In: *Brazilian Symposium on Databases (27th: 2012)*. [S.l.: s.n.], 2012. 45

OPENSTAX CNX. *Anatomy & Physiology*. [S.l.], 2013. Disponível em: <http://cnx.org/contents/FPtK1zmh@7.30:q2X995E3@7/The-Cell-Membrane>. Acesso em: 02 jan. 2015. 28

OREN, E. et al. What are Semantic Annotations. *Artificial Intelligence*, v. 8, 2006. 59

ÖZBAŞ, B.; ÖZGÜN, O.; BARLAS, Y. Modeling and simulation of the endogenous dynamics of housing market cycles. *Journal of Artificial Societies and Social Simulation*, v. 17, n. 1, p. 19, 2014. 35

PACIORETTY, L. M. et al. Reduction of the transient outward potassium current in a canine model of chagas' disease. *American Journal of Physiology-Heart and Circulatory Physiology*, Am Physiological Soc, v. 268, n. 3, p. H1258–H1264, 1995. 30, 85

PAN AMERICAN ORGANIZATION; WORLD HEALTH ORGANIZATION. *7th World Health Day: Chagas*. 2014. Disponível em: <http://www.paho.org/world-health-day-2014/wp-content/uploads/2014/04/Chagas.pdf>. Acesso em: 03 jan. 2015. 30

PAUL, W. E. *Fundamental immunology*. 6th edition. ed. [S.l.]: Lippincott Williams & Wilkins, Philadelphia., 2008. 31

PENG, R. D. Reproducible research in computational science. *Science*, v. 334, n. 6060, p. 1226–1227, 2011. 45

PERELSON, A. S.; WEISBUCH, G. Immunology for physicists. *Reviews of modern physics*, APS, v. 69, n. 4, p. 1219, 1997. 34

PIGOZZO, A. *Implementação computacional de um modelo matemático do sistema imune*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, 2011. 9, 35, 91, 92, 93, 94, 111

PIGOZZO, A. B. et al. Implementation of a computational model of the innate immune system. In: *Artificial Immune Systems*. [S.l.]: Springer, 2011. p. 95–107. 35, 95

QUINTELA, B. d. M.; SANTOS, R. W. dos; LOBOSCO, M. On the coupling of two models of the human immune response to an antigen. *BioMed research international*, Hindawi Publishing Corporation, v. 2014, 2014. 35

REED, K. et al. *General model of the innate immune response*. [S.l.], 2011. 35

RICHARDSON, L.; RUBY, S. *RESTful web services*. [S.l.]: O'Reilly, 2008. 72

ROCHA, P. A. F. et al. A three-dimensional computational model of the innate immune system. In: *Computational Science and Its Applications-ICCSA 2012*. [S.l.]: Springer, 2012. p. 691–706. 35

S MAURIN M, R. F. e. a. G. *The Hill equation: a review of its capabilities in pharmacological modeling*. 2008. 633-648 p. 93

SANTANA-PEREZ, I.; PÉREZ-HERNÁNDEZ, M. S. Towards reproducibility in scientific workflows: An infrastructure-based approach. *Scientific Programming*, Hindawi Publishing Corporation, v. 2015, 2015. 45

SANTO, D. P. M. E. *Simulação de Potencial de Ação Espontâneo em Miócitos Cardíacos do Ventrículo Esquerdo de Camundongos*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, 2014. 30

SCHWAB, M.; KARRENBACH, M.; CLAERBOUT, J. Making scientific computations reproducible. *Computing in Science & Engineering*, AIP Publishing, v. 2, n. 6, p. 61–67, 2000. 43

SENALDI, G. et al. Protection against the mortality associated with disease models mediated by tnf and ifn-γ in mice lacking ifn regulatory factor-1. *The Journal of Immunology*, Am Assoc Immnol, v. 163, n. 12, p. 6820–6826, 1999. 90

SHIELDS, M.; TAYLOR, I. Programming scientific and distributed workflow with triana services. In: *Proceedings of Workflow in Grid Systems Workshop in GGF10*. [S.l.: s.n.], 2004. 43

SMITH, G.; LEDBROOK, P. *Grails in Action*. [S.l.]: Manning, 2009. 71

STALLMAN, R. M.; MCGRATH, R. *GNU Make: A Program for Directed Recompilation, Version 3. 79. 1*. [S.l.]: Free Software Foundation, 2002. 43

STODDEN, V.; MIGUEZ, S.; SEILER, J. ResearchCompendia.org: Cyberinfrastructure for reproducibility and collaboration in computational science. *Computing in Science & Engineering*, AIP Publishing, v. 17, n. 1, p. 12–19, 2015. 23, 45, 114

SUŠNIK, J. et al. Integrated system dynamics modelling for water scarcity assessment: Case study of the kairouan region. *Science of the Total Environment*, Elsevier, v. 440, p. 290–306, 2012. 35

TAVERNA PROJECT. *Taverna: Workflow Authoring Environment*. 2007. Disponível em: <http://taverna.sourceforge.net>. Acesso em: 10 mai. 2013. 43

TUSSCHER, K. T. et al. A model for human ventricular tissue. *American Journal of Physiology-Heart and Circulatory Physiology*, Am Physiological Soc, v. 286, n. 4, p. H1573–H1589, 2004. 34, 35

VISTRAILS PROJECT. *VisTrails: A new scientific workflow and provenance management system*. 2015. Disponível em: <http://www.vistrails.org/>. Acesso em: 10 mai. 2015. 43

WAGNER, J. Kinetics of pharmacologic response I. Proposed relationships between response and drug concentration in the intact animal and man. *Journal of Theoretical Biology*, Elsevier, v. 20, n. 2, p. 173–201, 1968. 93

WAKELAND, W.; MACOVSKY, L.; AN, G. A hybrid simulation model for studying acute inflammatory response. In: SOCIETY FOR COMPUTER SIMULATION INTERNATIONAL. *Proceedings of the 2007 spring simulation multiconference-Volume 2*. [S.l.], 2007. p. 39–46. 35

WALTEMATH, D. et al. Minimum Information About a Simulation Experiment (MIASE). *PLoS Computational Biology*, Public Library of Science, v. 7, n. 4, 2011. 18, 22, 44, 45, 113

WALTEMATH, D. et al. Reproducible computational biology experiments with SED-ML - The Simulation Experiment Description Markup Language. *BMC Systems Biology*, Springer (Biomed Central Ltd.), v. 5, n. 1, p. 198, 2011. 18, 22, 113

WILLIAMS, T.; KELLEY, C. et al. *Gnuplot 4.4: an interactive plotting program*. 2010. Disponível em: <http://sourceforge.net/projects/gnuplot>. Acesso em: 28 mar. 2015. 71

WORLD HEALTH ORGANIZATION. *The top 10 causes of death: Fact Sheet 310*. 2014. Disponível em: <http://www.who.int/mediacentre/factsheets/fs310/en/>. Acesso em: 28 mar. 2016. 20

ZHAO, Y. et al. Swift: Fast, reliable, loosely coupled parallel computation. In: IEEE. *Services, 2007 IEEE Congress on*. [S.l.], 2007. p. 199–206. 42