Portabilidade dos Modelos Epidemiológicos disponíveis na MDD4ABMS para a Plataforma Repast

Fernando Santos[1], **Jéssica B. Petersen**[1]

[1] Departamento de Engenharia de Software
Universidade do Estado de Santa Catarina (UDESC)
Ibirama – SC – Brasil

fernando.santos@udesc.br, jessica041ster@gmail.com

Abstract. *MDD4ABMS is a model-driven approach for developing agent-based simulations. It provides ready-to-use, platform-independent building blocks for modeling aspects that are frequently used in agent-based simulations. One of these blocks provides epidemiological models for modeling diseases that spread among agents. MDD4ABMS offers portability of these epidemiological models by means of automatic code generation, but only for the NetLogo platform. This paper presents an extension of the MDD4ABMS approach to offer portability of epidemiological models to the Repast platform. A case study was conducted. The results obtained with the execution of the Repast and NetLogo simulations were similar, showing evidence that the portability of the epidemiological models provided by the MDD4ABMS approach is feasible.*

Resumo. *A MDD4ABMS é uma abordagem de desenvolvimento dirigido a modelos de simulações baseadas em agentes. Ela oferece blocos de construção prontos para uso e independentes de plataforma, permitindo especificar aspectos frequentemente utilizados em simulações baseadas em agentes. Um destes blocos oferece modelos epidemiológicos, para modelar doenças que se propagam através dos agentes. A MDD4ABMS oferece portabilidade destes modelos epidemiológicos através de geração automática de código, mas apenas para a plataforma NetLogo. Este artigo apresenta uma extensão da MDD4ABMS para oferecer portabilidade dos modelos epidemiológicos para a plataforma Repast. Um estudo de caso foi realizado. Os resultados obtidos com a execução das simulações Repast e NetLogo foram similares, evidenciando a viabilidade da portabilidade dos modelos epidemiológicos disponíveis na abordagem MDD4ABMS.*

1. Introdução

Simulações baseadas em agentes têm sido utilizadas para reproduzir e estudar comportamentos emergentes de sistemas complexos. Uma simulação baseada em agentes é constituída por agentes, entidades autônomas que possuem atributos e comportamentos próprios, e interagem entre si e com o ambiente. Desenvolver tais simulações é uma tarefa desafiadora, que tem sido investigada no contexto da modelagem e simulação baseada em agentes (*agent-based modeling and simulation*—ABMS) [Klügl e Bazzan 2012].

Simulações baseadas em agentes têm demonstrado sua relevância no atual contexto da pandemia de Covid-19, pois foram utilizadas para estudar a propagação do vírus

e avaliar estratégias de mitigação de infecções e mortes. Ainda no estágio inicial da pandemia em março de 2020, uma simulação baseada em agentes desenvolvida pelo Imperial College de Londres previu milhares de mortes no Reino Unido caso medidas não fossem tomadas [Ferguson et al. 2020]. Tais previsões fundamentaram a imediata adoção de medidas de contenção [Adam 2020], o que potencialmente salvou inúmeras vidas.

Recentemente foi proposta uma abordagem dirigida a modelos para desenvolver simulações baseadas em agentes denominada MDD4ABMS [Santos et al. 2018]. Esta abordagem é fundamentada na técnica de *model-driven development* (MDD), permitindo modelar simulações utilizando blocos de construção prontos para uso e independentes de plataforma. Estes blocos abstraem aspectos frequentemente utilizados em simulações, permitindo ao desenvolvedor focar *no que* simular, em vez de *em como* implementar tais aspectos. Um dos blocos de construção disponíveis na MDD4ABMS oferece modelos epidemiológicos, para simular doenças que se propagam nos agentes [Santos et al. 2020].

Por meio de geração automática de código, a MDD4ABMS oferece portabilidade de seus blocos de construção para a plataforma de simulação baseada em agentes NetLogo. Isto permite que o desenvolvedor execute a simulação especificada com a MDD4ABMS nesta plataforma. Recentemente, [Santos e Tenfen 2019] ofereceram portabilidade para a plataforma de simulação Repast, porém apenas de blocos que abstraem aspectos elementares da simulação (e.g., movimentação e sobrevivência dos agentes). Este artigo apresenta uma extensão da MDD4ABMS para oferecer portabilidade dos modelos epidemiológicos para a plataforma Repast, permitindo que desenvolvedores com alguma experiência em Repast também possam executar as simulações especificadas na MDD4ABMS. Um estudo de caso foi realizado, e os resultados obtidos com a execução das simulações Repast e NetLogo foram similares, evidenciando a viabilidade da portabilidade dos modelos epidemiológicos especificados com a abordagem MDD4ABMS.

O restante deste artigo está organizado da seguinte forma. A seção 2 apresenta a fundamentação teórica sobre modelagem e simulação baseada em agentes e a abordagem MDD4ABMS. A seção 3 apresenta a extensão desenvolvida neste artigo para suportar a portabilidade dos modelos epidemiológicos. Em seguida, a seção 4 descreve o estudo de caso realizado. Por fim, a seção 5 apresenta as conclusões e trabalhos futuros.

2. Fundamentação Teórica

2.1. Modelagem e Simulação baseada em Agentes

A modelagem e simulação baseada em agentes (ABMS) é um paradigma de simulação que utiliza agentes para analisar, reproduzir ou predizer fenômenos normalmente complexos e emergentes. Em ABMS, um modelo do sistema em estudo é construído em termos de agentes, que interagem com seus pares e também com o ambiente. A partir da simulação do modelo, através de repetida execução por determinado intervalo de tempo, pode-se obter informações significativas a respeito das propriedades do sistema ou fenômeno analisado, bem como sobre sua própria evolução [Garro e Russo 2010].

No paradigma ABMS, o desenvolvedor não possui restrições quanto a complexidade do agente que deseja especificar, sendo livre para definir técnicas sofisticadas para raciocínio e interações (por exemplo, técnicas de aprendizagem e redes neurais disponíveis na área de inteligência artificial), bem como para compor a população de agentes de forma

heterogênea. Desta forma, o pesquisador pode incorporar explicitamente na simulação a complexidade e diversidade de comportamento e interações dos indivíduos observada em cenários reais [Macal e North 2014].

De acordo com [Klügl e Bazzan 2012], uma simulação baseada em agentes é formada por três elementos: um conjunto de agentes autônomos; a especificação das interações entre os agentes e também com o ambiente, responsáveis por produzir a saída geral do sistema; e o ambiente simulado, que contém todos os demais elementos de simulação, como recursos e outros objetos sem comportamento ativo.

Simulações baseadas em agentes são frequentemente desenvolvidas em plataformas específicas, que disponibilizam recursos inerentes a agentes visando simplificar o desenvolvimento. De acordo com [Klügl e Bazzan 2012], entre as principais plataformas estão: NetLogo [Wilensky 1999], Repast [North et al. 2013] e MASON [Luke et al. 2005].

O NetLogo foi projetado para o usuário final, tendo uma interface para trabalhar com os parâmetros da simulação, e uma extensa biblioteca de modelos existentes. Além disto, NetLogo oferece uma linguagem de programação própria para desenvolver simulações. A plataforma Repast, por sua vez, permite desenvolver simulações utilizando programação Java. Através de bibliotecas Java existentes, o desenvolvedor pode incorporar na simulação quaisquer recursos utilizados de modo geral na indústria de software, como por exemplo integração com serviços web ou aprendizagem de máquina. Por fim, a MASON também é uma plataforma baseada em Java que visa facilitar a programação de simulações em larga escala para ganhar desempenho, sendo muito atrativo para simulações que demandam performance.

2.2. Abordagem MDD4ABMS para Desenvolvimento de Simulações com Agentes

A engenharia de software fornece suporte para o desenvolvimento de software profissional por meio de técnicas de especificação, projeto e evolução de sistemas. Uma destas técnicas é o desenvolvimento dirigido a modelos (*model-driven development*—MDD).

Em MDD são utilizados artefatos de modelagem que abstraem aspectros frequentemente presentes em softwares, para impulsionar o desenvolvimento de artefatos de baixo nível como códigos-fonte [Mernik et al. 2005]. O MDD facilita o desenvolvimento pois permite ao desenvolvedor focar em *quais* aspectos o software deve ter, em vez de focar em *como* desenvolver estes aspectos. A modelagem torna-se o papel chave no processo de desenvolvimento, pois é a partir do modelo que o código fonte será gerado automaticamente. Isto torna o desenvolvimento mais produtivo e eficiente, permitindo a portabilidade do software para diferentes plataformas [Mohagheghi et al. 2009].

Recentemente foi proposta uma abordagem MDD para desenvolver simulações baseadas em agentes chamada MDD4ABMS [Santos et al. 2018]. A MDD4ABMS foi elaborada a partir da análise de simulações existentes, onde foram identificados aspectos que são frequentemente utilizados em diferentes domínios. Estes aspectos foram abstraídos em um metamodelo de simulações com agentes.

A MDD4ABMS oferece a linguagem de modelagem ABStractLang, que fornece blocos de construção para os aspectos abstraídos no metamodelo de simulações. Além disso, a MDD4ABMS disponibiliza uma ferramenta de modelagem, cha-

mada ABStractme, que permite especificar simulações de acordo com a linguagem ABStractLang. A Figura 1 apresenta a ferramenta ABStractme. Na parte central está o diagrama com a especificação de uma simulação de propagação de doença, elaborado de acordo com a linguagem de modelagem ABStractLang. À direita observa-se uma paleta com os blocos de construção disponíveis para modelagem.

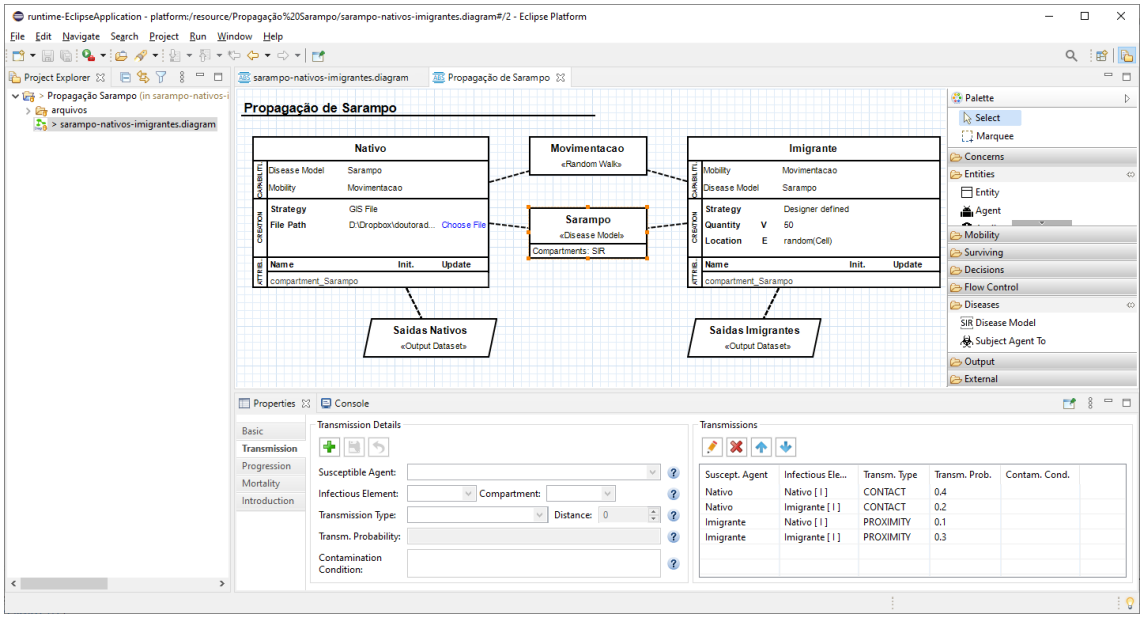

Figura 1. Simulação de propagação de doença especificada na MDD4ABMS

Os aspectos de simulações com agentes abstraídos no metamodelo da MDD4ABMS incluem: ambientes do tipo grade com inicialização a partir de arquivos georreferenciados; entidades e agentes com atributos, que podem ser inicializados a partir de dados fornecidos; coleta de dados para exibição e análise de resultados; habilidades de agentes, tais como mobilidade, sobrevivência, e aprendizagem por reforço; e modelos epidemiológicos de propagação de doenças [Santos et al. 2020].

Os modelos epidemiológicos permitem especificar a dinâmica de propagação de uma doença entre os agentes da simulação. Eles são fundamentados no modelo compartimental de [Kermack e McKendrick 1932], utilizado por epidemiologistas para prever e entender a propagação de doenças.

Estes modelos compartimentais classificam os indivíduos em compartimentos, de acordo com o estado de saúde. Por exemplo, o modelo epidemiológico elementar `SIR` considera os seguintes compartimentos: suscetível (`S`) para indivíduos ainda não infectados pela doença; infectado (`I`) para indivíduos contaminados pela doença e que podem manifestar sintomas e transmitir para outros agentes; e recuperado (`R`) para indivíduos curados e que eventualmente desenvolveram imunidade. A dinâmica da propagação da doença ocorre com a transição dos indivíduos por estes compartimentos. Essa transição é governada por parâmetros que caracterizam a doença, como por exemplo taxas de transmissão e de recuperação.

Variações do modelo `SIR` existem na literatura, sendo o modelo `SEIR` uma delas [Keeling e Rohani 2008]. No `SEIR` existe um compartimento adicional, exposto (`E`),

para indivíduos em que a doença está em período de incubação. Para doenças em que a imunidade é temporária, há modelos que especificam o retorno do indivíduo ao compartimento suscetível (S) após um período de tempo, como por exemplo os modelos SIRS e SEIRS. Os modelos SIR, SEIR, SIRS e SEIRS são contemplados pela abordagem MDD4ABMS.

A MDD4ABMS provê geração automática de código para simulações especificadas a partir de sua linguagem de modelagem, conferindo agilidade e produtividade ao desenvolvimento de simulações. Através da geração de código, os aspectos da simulação são portados para artefatos de plataformas de simulação baseada em agentes (e.g., códigos fonte e arquivos de configuração). É possível gerar código NetLogo para todos os aspectos contemplados pela abordagem MDD4ABMS [Santos et al. 2020]. Recentemente, [Santos e Tenfen 2019] ofereceram portabilidade (geração de código) para a plataforma Repast, mas apenas de aspectos elementares das simulações (ambientes do tipo grade, entidades, agentes, e as habilidades de mobilidade e sobrevivência). Este artigo apresenta uma extensão da abordagem MDD4ABMS para oferecer portabilidade dos modelos epidemiológicos para a plataforma Repast.

3. Portabilidade dos Modelos Epidemiológicos da MDD4ABMS

Para obter portabilidade dos modelos epidemiológicos para a plataforma Repast, este trabalho estendeu o gerador de código desenvolvido por [Santos e Tenfen 2019]. Este gerador é formado por regras de produção de código, que transformam os elementos do metamodelo MDD4ABMS (agentes, ambiente, etc) para elementos e blocos de comandos da plataforma Repast. As regras são descritas em Xpand,[1] uma linguagem para especificação de *templates* de geração de código. O restante desta seção descreve, de forma geral, as modificações realizadas no gerador de código Repast para contemplar os modelos epidemiológicos suportados pela MDD4ABMS.

Inicialmente, foi necessário projetar como os modelos epidemiológicos poderiam ser implementados na plataforma Repast. Tendo em vista que na Repast as simulações são desenvolvidas em linguagem Java, optou-se por adotar um projeto orientado a objetos. A Figura 2 apresenta o diagrama de classes do projeto de implementação dos modelos epidemiológicos. Na Repast, por definição, o desenvolvedor precisa implementar uma classe Java para cada tipo de agente existente na simulação. No diagrama, isto está representado pela classe AgentX (na implementação concreta seria chamada por exemplo de Humano).

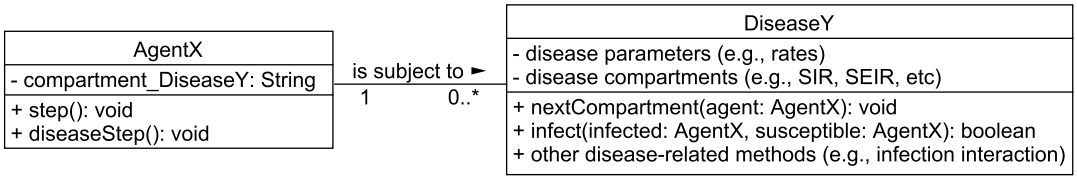

Figura 2. Projeto de implementação dos modelos epidemiológicos em Repast

Assumimos que, ao utilizar um modelo epidemiológico na simulação, o desenvolvedor está especificando uma doença que acometerá um ou vários tipo(s) de

[1] http://www.eclipse.org/modeling/m2t/?project=xpand

agente(s). Neste sentido, a doença é representada por uma nova classe, `DiseaseY`. Na implementação concreta, o nome da classe é substituido pelo nome da doença, por exemplo `Sarampo`. Durante a execução, um ou mais objetos de `DiseaseY` estarão associados a cada agente que estiver sujeito à(s) doença(s). O comportamento do agente será afetado pela dinâmica de propagação e parâmetros desta(s) doença(s).

O gerador de [Santos e Tenfen 2019] já estava gerando o código Java da classe `AgentX`. Modificações foram realizadas para gerar atributos relacionados à doença, como por exemplo o compartimento no qual o agente se encontra atualmente, e também para gerar um novo método `diseaseStep()` no agente, para ativar a dinâmica de propagação da doença no objeto `DiseaseY` associado. Além disso, a geração do método `step()` foi estendida. Este método, que implementa o comportamento do agente durante um ciclo de execução da simulação, agora ativa o método `diseaseStep()` caso o agente esteja sujeito a alguma doença.

A classe `DiseaseY` encapsula os atributos e métodos que gerenciam a dinâmica da doença. Para os parâmetros que caracterizam a doença (e.g, taxas de transmissão, recuperação, e mortalidade) são gerados atributos. Também são gerados atributos para identificar os compartimentos do modelo compartimental adotado na doença.

Dentre os métodos gerados para implementar a dinâmica da doença na classe `DiseaseY`, destacam-se: `nextCompartment()`, que determina o próximo compartimento do agente de acordo com a evolução da doença; e `infect()`, responsável por fazer com que um agente infectado contamine um suscetível quando ocorrer interação entre eles.

Outros métodos são gerados para implementar aspectos relacionados à dinâmica da doença, como por exemplo identificar as interações entre os agentes onde poderá ocorrer a infecção. Estes aspectos relacionados à dinâmica dos modelos compartimentais estão documentados em [Santos et al. 2020].

Para gerar o código da classe `DiseaseY` foi desenvolvido um novo módulo Xpand, com a implementação de regras para gerar código Java dos atributos e métodos que realizam a dinâmica da doença. A Figura 3 apresenta a principal regra Xpand, que define e gera a estrutura da classe da doença. Inicialmente a regra gera os elementos Java *package* e *imports* nas linhas 2 e 3, respectivamente. Na linha 4 é gerada a declaração da classe Java da doença. Conforme mencionado anteriormente, o nome `DiseaseY` apresentado na Figura 2 é substituído pelo nome da doença especificada no diagrama da simulação. Em seguida, são invocadas diversas regras Xpand responsáveis por gerar código para os outros aspectos da doença. Nas linhas 5 a 7 são invocadas regras para gerar os atributos, o construtor, e os métodos *getters*/*setters*. Os métodos `nextCompartment()` e `infect()` são gerados pelas regras invocadas nas linhas 8 e 11, respectivamente.

As regras ativadas nas demais linhas são responsáveis por gerar códigos que implementam os outros aspectos relacionados à doença, como por exemplo a transição entre os compartimentos do modelo epidemiológico adotado (na MDD4ABMS estes compartimentos são chamados de estados) e a identificação das interações entre agentes onde ocorre a infecção (e.g., por contato físico ou proximidade entre agentes).

```
1  «DEFINE DiseaseClasses (mm::DiseaseModel diseaseModel) FOR mm::AgentBasedSimulation»
2  package «GET_PACKAGE_NAME(this)»;
3  «EXPAND importsDiseaseClass ((DiseaseModel)diseaseModel) FOR this»
4  public class «EXPAND getClassName ((DiseaseModel)diseaseModel)  FOR this» {
5    «EXPAND diseaseAttributes ((DiseaseModel)diseaseModel) FOR this»
6    «EXPAND diseaseConstructor ((DiseaseModel)diseaseModel) FOR this»
7    «EXPAND diseaseGettersSetters ((DiseaseModel)diseaseModel) FOR this»
8    «EXPAND methodNext ((DiseaseModel)diseaseModel) FOR this»
9    «EXPAND resetDurations ((DiseaseModel)diseaseModel) FOR this»
10   «EXPAND transitionBetweenStates ((DiseaseModel)diseaseModel) FOR this»
11   «EXPAND infect ((DiseaseModel)diseaseModel) FOR this»
12   «EXPAND contactTransmission ((DiseaseModel)diseaseModel)  FOR this»
13   «EXPAND getTransmissionByInfected  FOR this»
14   «EXPAND returnAgents ((DiseaseModel)diseaseModel) FOR this»
15   «EXPAND proximityTransmission ((DiseaseModel)diseaseModel) FOR this»
16   «EXPAND returnValueOfEdgeDimension ((DiseaseModel)diseaseModel) FOR this»
17   «EXPAND getTypeTransmission FOR this»
18   «EXPAND transitionValue ((DiseaseModel)diseaseModel) FOR this»
19 }
20 «ENDDEFINE»
```

Figura 3. Regra Xpand para Gerar a Classe Java do Agente Repast

4. Estudo de Caso

Esta seção descreve o estudo de caso conduzido para demonstrar a viabilidade da portabilidade dos modelos epidemiológicos e os resultados obtidos.

4.1. Materiais e Métodos

Duas simulações de propagação de doenças foram especificadas para o estudo de caso. Uma delas utilizou o modelo epidemiológico `SIR`, e a outra o modelo `SEIR`. A partir do modelo da simulação, especificado a partir da ferramenta ABStractme da MDD4ABMS, foram gerados código fonte para as plataformas Repast e NetLogo. As simulações foram executadas, e os resultados obtidos nestas plataformas foram comparados. O resultado obtido com a NetLogo é considerado *baseline*, tendo em vista que a consistência da sua geração de código já foi abordada por [Santos et al. 2020]. A portabilidade será obtida com sucesso caso os resultados das simulações Repast e NetLogo sejam similares quando executados com os mesmos valores de parâmetros.

As simulações desenvolvidas consideram a propagação de Sarampo entre dois tipos de agentes, arbitrariamente denominados de Nativos e Imigrantes. Estes agentes podem se movimentar em um ambiente no formato de grade. A modelagem desta simulação está retratada na Figura 1, apresentada como exemplo de modelagem na seção 2.2.

A Tabela 1 apresenta a quantidade de agentes por tipo, e também os valores dos parâmetros utilizados nas simulações. Valores arbitrários foram utilizados para as quantidades de agentes e os parâmetros que caracterizam a disseminação, recuperação, e mortalidade da doença simulada, pois o objetivo do estudo não é analisar a propagação da doença em uma situação real de epidemia, mas sim avaliar a consistência do gerador de código desenvolvido e consequentemente a viabilidade da portabilidade para a plaforma Repast. Neste sentido, optou-se por introduzir a doença e estabelecer uma taxa de mortalidade apenas para um tipo de agente.

Com relação às taxas de transmissão, também foram utilizados diferentes valores para cada possível interação entre os agentes. Na Tabela 1, estas interações e taxas

Agente	Qtd.	Infectados*	Taxas de transmissão	Taxa mortalidade	Modelo	Durações
Nativo	600	nenhum	Nativo: 0.4 (contato)	0.2	SIR	I: 20
			Imigrante: 0.3 (proximidade: 2)		SEIR	I: 20 E: 3 R: 25
Imigrante	100	5	Imigrante: 0.2 (proximidade: 1)	nenhuma	SIR	I: 15
			Nativo: 0.1 (contato)		SEIR	I: 15 E: 5 R: 30

* Infectados no início da simulação

Tabela 1. Parâmetros utilizados nas simulações

devem ser interpretadas da seguinte maneira: o agente da coluna *Agente* transmite para os agentes (interage) da coluna *Taxas de transmissão* de acordo com a taxa informada nesta coluna. Na MDD4ABMS esta interação pode ser por contato, quando os agentes ocupam a mesma posição no ambiente, ou por proximidade, quando estão situados até determinada distância. Por fim, a tabela apresenta as durações dos compartimentos para os modelos SIR e SEIR utilizados nas simulações. Estas durações correspondem à quantidade de passos da simulação que o agente permanecerá nos compartimentos.

As simulações Repast e NetLogo geradas a partir da especificação em MDD4ABMS foram executadas 20 vezes cada. As quantidades de agentes suscetíveis, infectados, expostos e recuperados, bem como a quantidade total de agentes, foram coletados a cada passo da simulação, e por fim foram calculadas a média e desvio padrão destas quantidades. A simulação com o modelo SIR foi executada por 90 passos. Já a simulação com o modelo SEIR foi executada por 210 passos para verificar a reincidência da doença nos agentes, pois nesta simulação foi especificada uma duração para o compartimento R, tornando a imunidade temporária.

4.2. Resultados com o Modelo Epidemiológico SIR

As Figuras 5 e 4 apresentam a média e desvio padrão dos resultados obtidos para a simulação SIR para cada tipo de agente. Os resultados obtidos na Repast são apresentados em linhas com cores intensas, e os obtidos na NetLogo em linhas com cores claras (e.g., verde intenso e verde claro).

Nos agentes Imigrantes observa-se um pico de infecções por volta do passo 20, que decai a medida que os agentes se recuperam da doença. A quantidade total de agentes Imigrantes permanece constante, pois para estes agentes não há taxa de mortalidade.

Por outro lado, no caso dos agentes Nativos, o pico de infecções ocorre por volta do passo 37 em razão da maior quantidade de agentes deste tipo, o que faz com que a doença circule e infecte maior quantidade de agentes. Como agentes Nativos estão sujeitos a uma taxa de mortalidade, sua quantidade decai ao longo da simulação.

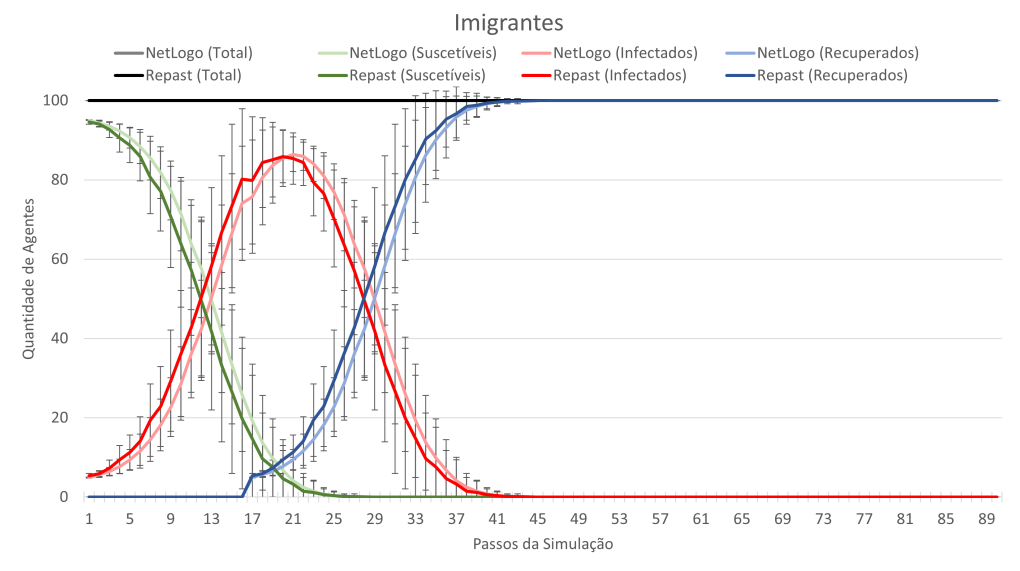

Figura 4. Resultados com o modelo epidemiológico SIR e agentes Imigrantes

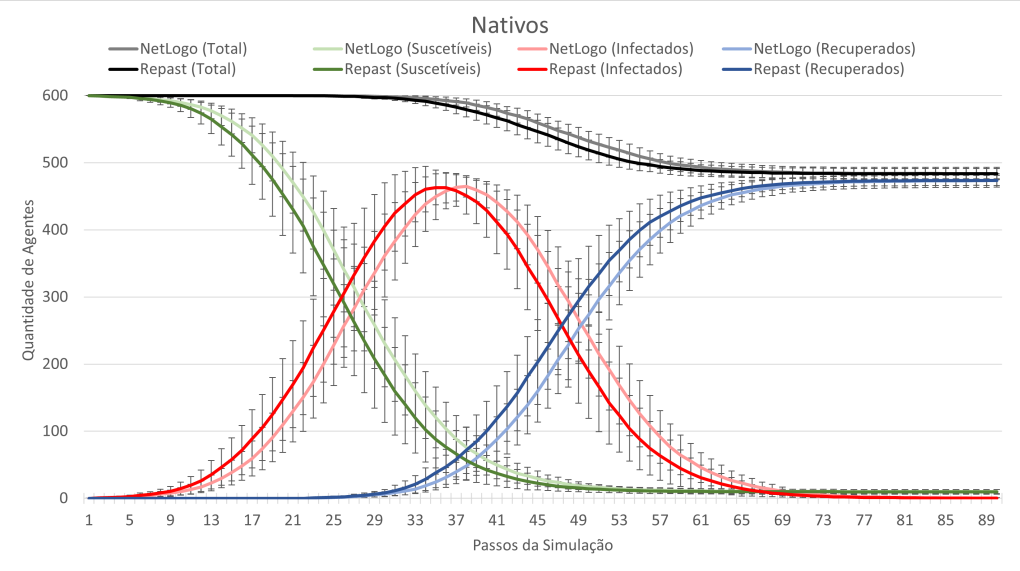

Figura 5. Resultados com o modelo epidemiológico SIR e agentes Nativos

4.3. Resultados com o Modelo Epidemiológico SEIR

As Figuras 7 e 6 apresentam a média e desvio padrão dos resultados obtidos para a simulação `SEIR`, utilizando também a representação de cores intensas e claras para Repast e NetLogo, respectivamente.

Em relação à simulação anterior, notam-se duas principais diferenças. A primeira é a existência de agentes expostos, que se encontram no compartimento `E` (linhas amarelas). Os agentes contaminados pela doença permanecem neste compartimento por um período de tempo até evoluirem para o compartimento `I`. A segunda difença é a oscilação nas quantidades de agentes em cada compartimento. Isto ocorre pois nesta simulação

ambos agentes possuem imunidade temporária, retornando ao compartimento S após permanecer certo tempo no compartimento R. Estas oscilações representam as *ondas* de contaminação por uma doença. Nos agentes Imigrantes, as ondas possuem amplitude similar ao longo da simulação pois não há mortalidade de agentes. Já no caso dos agentes Nativos, as amplitudes são diferentes pois há mortalidade de agentes.

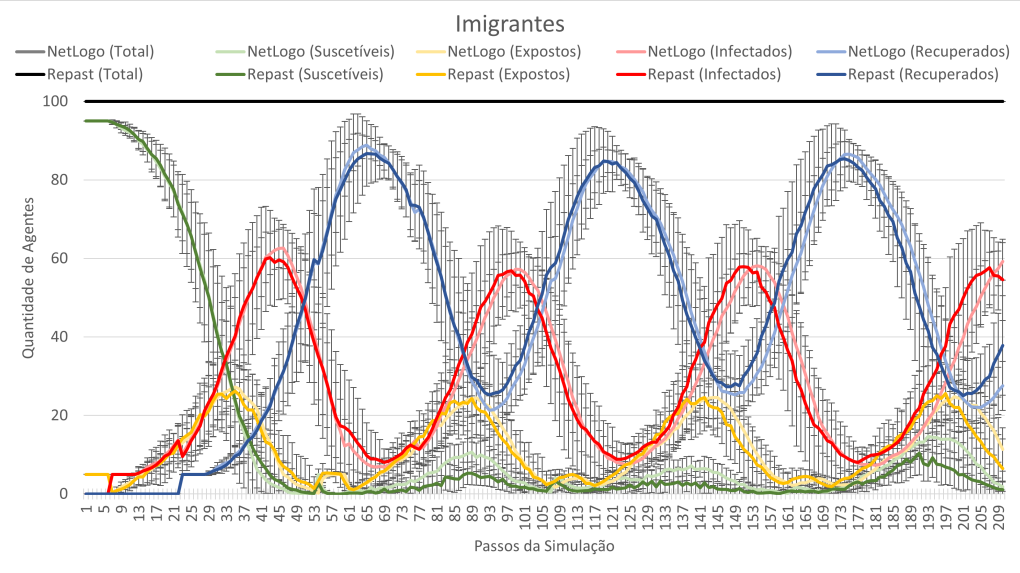

Figura 6. Resultados com o modelo epidemiológico SEIR e agentes Imigrantes

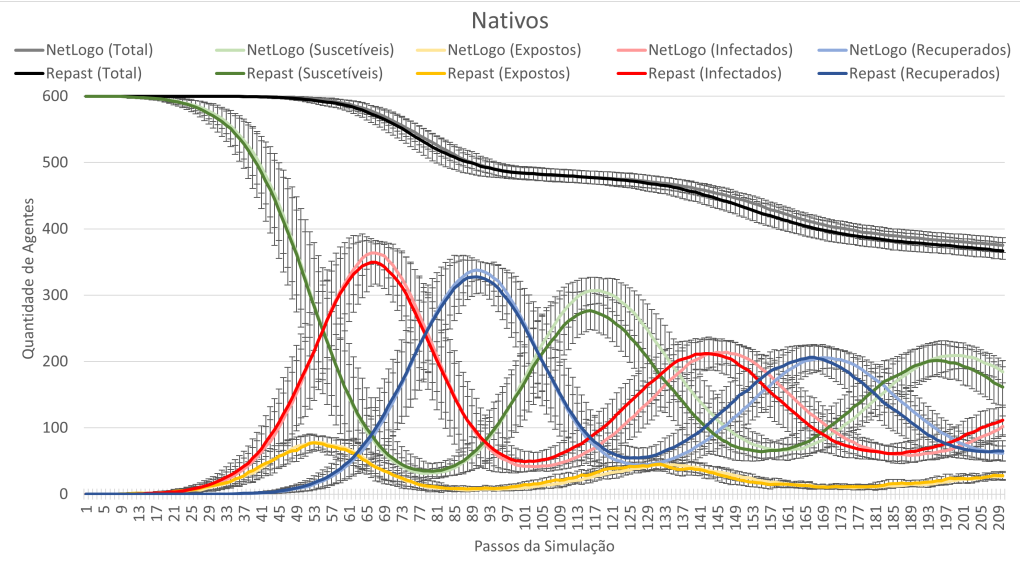

Figura 7. Resultados com o modelo epidemiológico SEIR e agentes Nativos

Em ambas as simulações, observa-se que os resultados obtidos na Repast coincidem, dentro das margens indicadas pelos desvios padrão, com os resultados obtidos na NetLogo (*baseline*). Isto evidencia sucesso na portabilidade dos modelos epidemiológicos disponíveis na plataforma MDD4ABMS para a plataforma Repast.

5. Conclusão

Este trabalho estendeu a abordagem MDD4ABMS para suportar a portabilidade dos modelos epidemiológicos para a plataforma Repast. A portabilidade foi obtida por meio de modificações no gerador de código Repast existente, além do desenvolvimento de um novo módulo para gerar código dos aspectos relacionados à dinâmica dos modelos epidemiológicos. Essa portabilidade pode ser útil em meios acadêmicos e profissionais, pois oferece maior liberdade aos interessados em simulações com agentes, fomentando a adoção do paradigma de modelagem e simulação baseada em agentes.

Um estudo de caso foi realizado, considerando a modelagem de duas simulações de propagação de doenças. O gerador de código desenvolvido foi utilizado para gerar código fonte Repast destas simulações. O resultado da execução destas simulações foi similar ao resultado obtido na plataforma NetLogo, evidenciando a viabilidade da portabilidade dos modelos epidemiológicos especificados com a abordagem MDD4ABMS para a plataforma Repast.

Como trabalhos futuros, sugere-se melhorias no gerador de código para suportar modelos epidemiológicos customizados. Nestes modelos customizados o desenvolvedor pode definir novos compartimentos para customizar a dinâmica da propagação da doença, para atender casos em que seja necessário a adoção de modelos compartimentais ainda não contemplados pela MDD4ABMS. Ainda que a abordagem MDD4ABMS suporte a especificação de modelos epidemiológicos customizados, tanto o gerador de código Repast quanto NetLogo não suportam a sua geração de código. Além disso, outra sugestão de trabalho futuro é evoluir a própria abordagem MDD4ABMS para suportar a especificação de vetores de transmissão de doenças. Vetores de transmissão são agentes que não estão sujeitos aos sintomas ou mortalidade, apenas estão infectados e transmitem o patógeno causador da doença. Exemplos de vetores são mosquitos (dengue) e carrapatos (febre maculosa).

Referências

Adam, D. (2020). Special report: The simulations driving the world's response to COVID-19. *Nature*, 580(7803):316–318.

Ferguson, N. M., Laydon, D., Nedjati-Gilani, G., Imai, N., Ainslie, K., Baguelin, M., Bhatia, S., Boonyasiri, A., Cucunubá, Z., Cuomo-Dannenburg, G., Dighe, A., Dorigatti, I., Fu, H., Gaythorpe, K., Green, W., e Hamlet, A. (2020). Report 9: Impact of non-pharmaceutical interventions (NPIs) to reduce COVID19 mortality and healthcare demand. Technical report, MRC Centre for Global Infectious Disease Analysis, London.

Garro, A. e Russo, W. (2010). easyABMS: A domain-expert oriented methodology for agent-based modeling and simulation. *Simulation Modelling Practice and Theory*, 18(10):1453–1467.

Keeling, M. J. e Rohani, P. (2008). *Modeling infectious diseases in humans and animals*. Princeton University Press.

Kermack, W. O. e McKendrick, A. G. (1932). Contributions to the mathematical theory of epidemics. ii.—the problem of endemicity. *Proc. R. Soc. Lond. A*, 138(834):55–83.

Klügl, F. e Bazzan, A. L. C. (2012). Agent-based modeling and simulation. *AI Magazine*, 33(3):29–40.

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., e Balan, G. (2005). Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527.

Macal, C. e North, M. (2014). Introductory tutorial: Agent-based modeling and simulation. In *Proceedings of the 2014 Winter Simulation Conference*, WSC '14, pages 6–20, Piscataway, NJ, USA. IEEE Press.

Mernik, M., Heering, J., e Sloane, A. M. (2005). When and how to develop domain-specific languages. *ACM computing surveys (CSUR)*, 37(4):316–344.

Mohagheghi, P., Dehlen, V., e Neple, T. (2009). Definitions and approaches to model quality in model-based software development – a review of literature. *Information and Software Technology*, 51(12):1646–1669.

North, M. J., Collier, N. T., Ozik, J., Tatara, E. R., Macal, C. M., Bragen, M., e Sydelko, P. (2013). Complex adaptive systems modeling with repast simphony. *Complex Adaptive Systems Modeling*, 1(1):1–26.

Santos, F., Nunes, I., e Bazzan, A. L. (2020). Quantitatively assessing the benefits of model-driven development in agent-based modeling and simulation. *Simulation Modelling Practice and Theory*, 104:102–126.

Santos, F., Nunes, I., e Bazzan, A. L. C. (2018). Model-driven agent-based simulation development: a modeling language and empirical evaluation in the adaptive traffic signal control domain. *Simulation Modelling Practice and Theory*, 83:162–187.

Santos, F. e Tenfen, R. (2019). Extensão da abordagem de desenvolvimento dirigido a modelos de simulações com agentes (MDD4ABMS) para suportar portabilidade de simulações. In *Anais da III Escola Regional de Engenharia de Software*, pages 33–40, Porto Alegre, RS, Brasil. SBC.

Wilensky, U. (1999). NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.