1
Introdução

Com o passar dos anos, os sistemas de computação vêm sofrendo grandes alterações para atender as novas necessidades do mercado. Em sua maioria, os novos sistemas devem ser concorrentes e distribuídos, e interagir com componentes, explorando serviços que são dinamicamente encontrados na internet. Estes sistemas são normalmente entidades que funcionam continuamente, isto é, que não podem ser interrompidas [1]. Em virtude destas características, tem-se buscado sistemas de larga-escala, complexos, robustos e autônomos, capazes de operar em diferentes plataformas. Para atender a esta nova demanda surgem novas tecnologias que oferecem respostas dinâmicas para as rápidas mudanças de circunstâncias exigidas neste tipo de sistema.

Os sistemas multi-agentes (SMA) são utilizados para atender a esta nova necessidade procurando dar suporte à complexidade encontrada no desenvolvimento de sistemas distribuídos de grande porte, através do uso da tecnologia de agentes de software. SMA caracterizam-se pela existência de uma certa quantidade de agentes autônomos, heterogêneos e independentes, trabalhando em conjunto para resolver um problema. Esses agentes são aptos a se adaptarem ao ambiente em que atuam, reagir a ele e provocar mudanças neste meio. Pode-se dizer então que um agente de software funciona continuamente, é capaz de comunicar-se com outros agentes, cooperar com eles, e ainda mover-se de um ambiente para outro [2].

As características dos SMA se encaixam perfeitamente às necessidades encontradas no desenvolvimento de sistemas complexos. Primeiro, a autonomia dos componentes da aplicação (ou seja, a habilidade de um agente em decidir que ação irá tomar em determinado momento) reflete a natureza descentralizada dos sistemas distribuídos modernos. Segundo, o modo flexível com o qual os agentes operam (tendo um comportamento diferenciado a cada resposta do ambiente, por exemplo) se encaixa às situações dinâmicas e imprevisíveis no que se espera atualmente em um sistema. Finalmente, a natureza dinâmica presente nas interações multi-agentes é apropriada para sistemas abertos, onde seus componentes e suas interações estão em constante mudança [1].

Portanto, SMA estão emergindo como um poderoso paradigma para a modelagem e desenvolvimento de sistemas complexos. No entanto, como é comum acontecer com novos paradigmas na engenharia de software, para o sucesso e a disseminação do desenvolvimento de SMA tem-se a necessidade não somente de novos modelos e tecnologias, como também novas metodologias para dar suporte ao desenvolvimento deste tipo de sistema [3].

Embora sistemas multi-agentes sejam de certa forma similares a sistemas orientados a objetos (OO) (ambos aderem ao princípio do encapsulamento e reconhecem a importância de interação) [4], existem algumas diferenças entre eles que fazem com que novas abstrações necessitem ser criadas para atender a abordagem orientada a agentes. Uma das principais diferenças entre agentes e objetos é o fato de os objetos serem passivos por natureza, necessitando de algum impulso externo para que se tornem ativos. Outra diferença está relacionada a noção de comportamento autônomo flexível (reativo, pró-ativo e social), pois o modelo OO não contém artifícios para a construção de sistema com este tipo de comportamento [5]. Devido a essas diferenças, novas linguagens de modelagens para SMA foram criadas [6, 7, 8, 9] para representar as novas abstrações presentes neste tipo de sistema. Muitas delas, por exemplo [6, 8, 9], se baseiam em linguagens de modelagens para sistemas OO.

Atualmente já existem algumas linguagens de programação específicas para SMA [10, 11, 12]. No entanto, na maioria dos casos, esses sistemas são implementados utilizando-se linguagens de programação orientadas a objetos. Deste modo, faz se necessária a criação de um mapeamento da linguagem de modelagem especifica para SMA (seja ela qual for) para linguagens de programação OO.

Como em qualquer outro processo de desenvolvimento de software, o desenvolvimento de SMA também está centrado no levantamento de requisitos da aplicação, na análise do domínio, no design, na implementação e nos testes do sistema. Para o correto desenvolvimento de SMA é imprescindível a utilização de metodologias que permitam que sua complexidade seja completamente gerenciada. Por este motivo um grande número de metodologias [1, 13, 14] são propostas na literatura de SMA para dar suporte à construção de SMA. Muitas dessas metodologias e suas ferramentas estão focadas em uma arquitetura específica de agentes.

Este trabalho tem por objetivo sugerir o uso da arquitetura Model Driven Architecture (MDA) [15], proposta pela OMG, no processo de desenvolvimento de SMA. MDA é uma arquitetura para desenvolvimento de software, que dá

suporte a todo o ciclo de desenvolvimento, ou seja, engloba todas as fases do desenvolvimento de um sistema. Sua principal característica é a importância dada aos modelos gerados nas diferentes etapas do processo de desenvolvimento e no rastreamento dos modelos de uma etapa para a etapa seguinte.

A arquitetura MDA é composta por quatro etapas principais, onde ao final de cada etapa tem-se como resultado modelos formais. A passagem de uma etapa para a etapa seguinte é realizada através de transformações de um modelo para outro. Estas transformações possibilitam o rastreamento dos modelos. Existem três tipos básicos de modelos definidos no processo de desenvolvimento de MDA: modelos independentes de computação (CIM), modelos independentes de plataforma (PIM) e modelos específicos de plataforma (PSM). Além da especificação dos modelos, MDA define ainda que um conjunto de transformações consecutivas deve ser aplicado aos modelos para permitir transformar modelos de mais alto nível de abstração em código.

O objetivo da proposta apresentada é oferecer um ambiente que permita o desenvolvimento de SMA utilizando a abordagem MDA. Neste trabalho, utilizamos a linguagem de modelagem MAS-ML [8] para modelar SMA em um alto nível de abstração. A modelagem de um SMA gerada utilizando-se MAS-ML corresponderá aos modelos PIM propostos em MDA. Tais modelos são independentes da plataforma de implementação, pois MAS-ML não restringe ou especifica a plataforma onde os modelos devem ser implementados. MAS-ML é uma linguagem de modelagem para SMA que estende UML incluindo abstrações relacionadas a agentes. Utilizando-se MAS-ML é possível modelar os aspectos estruturais (as abstrações, suas propriedades e relacionamentos entre as abstrações) e os aspectos dinâmicos (as interações entre as abstrações e as execuções internas das mesmas) de SMA.

De acordo com MDA, após a geração dos modelos PIM estes modelos são transformados em modelos PSM. Em nossa proposta, os modelos MAS-ML são transformados em modelos UML [16] utilizando-se o framework ASF (Agent Society Framework) [17]. O framework ASF possibilita a implementação de SMA utilizando a linguagem de programação orientada a objetos Java. Os modelos UML definidos nesta etapa incluem características dependentes do framework e da linguagem de programação onde estes serão implementados. Por fim, na última etapa do processo de desenvolvimento, os modelos UML específicos de plataforma são transformados em código. Utilizamos XMI [18] para representar

os modelos UML e possibilitar a geração de código da aplicação a partir destes modelos.

A utilização da abordagem de MDA no processo de desenvolvimento proposto traz uma série de benefícios para o desenvolvimento de SMA. Devido ao design de modelos independentes de plataforma através do uso de MAS-ML, os modelos PIM de descrição do problema são modelos portáveis que podem ser reutilizados para geração de diferentes modelos computacionais utilizando-se diferentes plataformas de implementação. Em nossa proposta utilizamos o framework ASF para geração de modelos PSM a partir de modelos MAS-ML, porém a plataforma Jadex [19, 20], por exemplo, também poderia ter sido utilizada. Outra vantagem no uso de MDA é a geração de diferentes modelos cujos focos estão em diferentes pontos de vista do sistema. Os modelos MAS-ML de alto nível de abstração focam na descrição do problema, descrevendo o sistema no nível de abstração de agentes. Já os modelos UML de mais baixo nível de abstração focam na descrição da solução do problema incluindo detalhes de implementação. Durante a manutenção dos modelos MAS-ML, o designer não se preocupa com os detalhes de implementação e se concentra na definição do problema. Se as alterações a serem feitas estiverem relacionadas restritamente à plataforma de implementação, os modelos UML serão modificados, porém os modelos MAS-ML não sofrerão alterações. O uso de MDA possibilita o baixo acoplamento entre os diferentes modelos gerados durante o processo de desenvolvimento.

Com o intuito de viabilizar a metodologia proposta, foi desenvolvida uma ferramenta que segue todas as etapas propostas no processo de desenvolvimento, desde a modelagem até a implementação. Esta ferramenta, similar às ferramentas CASE já existentes, oferece ao usuário a facilidade de modelar graficamente seu sistema de forma rápida e simples, através da linguagem de modelagem MAS-ML. Após a modelagem gráfica, ela é capaz de realizar as transformações existentes no processo de desenvolvimento proposto de forma a gerar como produto final o código referente a aplicação modelada anteriormente.

1.1. Principais Contribuições

As principais contribuições deste trabalho são:

1. Criação de um processo de desenvolvimento de sistemas multi-agentes, com base na abordagem MDA e utilizando a linguagem de modelagem MAS-ML, framework ASF e a tecnologia XMI neste processo. Através deste processo temos como benefícios:

- A utilização de MDA permitindo a geração de diferentes modelos para cada etapa de transformação, desde a modelagem até a implementação, apresentando os diferentes pontos de vista do sistema, isto é, diferentes níveis de abstração. Com a utilização de MDA, o trabalho proposto se beneficia de todas as vantagens decorrentes do uso de MDA: portabilidade, reusabilidade, interoperabilidade e baixo acoplamento;

- A utilização da tecnologia XMI permite representar de forma padronizada e textual os diferentes modelos gerados durante cada etapa presente no processo de desenvolvimento. Além disso, possibilita a troca de modelos entre diferentes ferramentas (interoperabilidade de modelos);

- O emprego da linguagem MAS-ML específica para SMA foi utilizada como linguagem de modelagem no processo de desenvolvimento proposto por atender as principais características desses sistemas e ser independente de plataforma, estando de acordo portanto com o modelo PIM de MDA;

- O ASF framework foi utilizado na transformação dos modelos MAS-ML em modelos UML específicos de plataforma, sendo uma etapa importante na transformação entre modelos, característica predominante em MDA.

2. Criação de uma ferramenta que apóia o processo de desenvolvimento proposto, viabilizando a modelagem e implementação de SMA através do uso do processo de desenvolvimento proposto neste trabalho, ou seja, passando por todas as etapas propostas.

1.2. Organização

O trabalho está organizado da seguinte forma:

O Capítulo 2 apresenta alguns conceitos importantes para o entendimento do trabalho, como a definição de sistemas multi-agentes e uma breve explicação sobre alguns conceitos e tecnologias utilizados como: a linguagem de modelagem MAS-ML, o framework ASF, a arquitetura MDA e a tecnologia XMI.

O Capítulo 3 apresenta alguns trabalhos relacionados ao tema.

O Capítulo 4 propõe um processo de desenvolvimento para sistemas multi-agentes, utilizando a abordagem MDA.

O Capítulo 5 apresenta a ferramenta criada com o objetivo de implementar o processo de desenvolvimento proposto no capitulo anterior.

O Capítulo 6 apresenta dois exemplos de aplicações de sistemas multi-agentes, onde o processo de desenvolvimento proposto foi aplicado.

O Capítulo 7 tece considerações finais sobre o trabalho, conclusões e trabalhos futuros.

O Anexo I apresenta o MAS-ML DTD gerado para atender ao processo proposto.

O Anexo II apresenta o exemplo do MAS-ML XMI e do UML XMI para o exemplo de aplicação *Virtual MarketPlace* demonstrado neste trabalho.

O Anexo III apresenta o exemplo do MAS-ML XMI para o exemplo de aplicação Expert Committee também demonstrado neste trabalho.