1
Introdução

Resumo

Esta introdução descreve a motivação para a elaboração de um ambiente que dê suporte computacional ao desenvolvimento de sistemas multi-agentes auxiliando o desenvolvedor durante o ciclo de vida de construção (da modelagem à implementação) destes sistemas e apresenta uma breve visão da abordagem da solução proposta. Finalmente, é apresentada uma lista com as contribuições deste trabalho e um guia do leitor, com a estrutura deste documento.

Atualmente, a rápida expansão da Internet traz desafios constantes que incluem o acesso a informações relevantes, a identificação de oportunidades, a ação no momento preciso, a manipulação de grandes volumes de informações e etc. Tudo isso vem contribuindo bastante para a tendência de desenvolvimento de sistemas de larga escala, complexos e autônomos, capazes de operar em diferentes plataformas. Para atender às necessidades destes novos sistemas, o conceito de agentes de software está sendo introduzido como novo paradigma de desenvolvimento de software.

Os Sistemas Multi-Agentes (SMA) são propícios para dar suporte à complexidade encontrada no desenvolvimento de aplicações distribuídas (abertas) de grande porte. Estes sistemas são caracterizados pela existência de agentes autônomos, heterogêneos e independentes, permitindo que sistemas possam ser implementados através do comportamento mais simples de diversos agentes. Na prática, um SMA é composto de múltiplos tipos de agentes, cada um destes com propriedades e capacidades distintas.

Um agente de software é capaz de se adaptar ao ambiente em que atua, reagir a ele e provocar mudanças neste meio. Pode também se comunicar com outros agentes que estejam presentes no meio, movendo-se de um ambiente para outro. Um bom exemplo do uso da tecnologia de agentes de software é o comércio

eletrônico, onde os agentes podem tanto procurar por fornecedores, clientes e produtos, bem como negociar um determinado produto.

Assim, para impulsionar o uso de SMA, a criação de um Ambiente de Desenvolvimento de Software (ADS) é de extrema importância. Ele é responsável por organizar, padronizar, simplificar e acelerar a construção de sistemas de software. Um ADS pode ser visto como um conjunto de ferramentas integradas que facilitam a realização de atividades da engenharia de software, apoiando todo o ciclo de vida do produto de software desenvolvido [12]. A idéia de um ADS como um conjunto integrado de ferramentas que suportam o processo completo de desenvolvimento do software tem evoluído bastante nas últimas duas décadas. Apoiar o processo do desenvolvimento significa apoiar o desenvolvimento e a manutenção de todos os tipos de documentos como especificações de requerimentos, listas de códigos, etc. O objetivo final de tal conjunto de ferramentas é melhorar a qualidade do produto final, para apoiar o reuso nos projetos de software, livrando os desenvolvedores do trabalho rotineiro.

Um ambiente inclui editores que apoiam todos os tipos de entrada gráfica e textual, analisadores, compiladores incrementais, browsers, debuggers, etc. Mesmo assim, ferramentas que informam sobre interdependências de documentos e controle de consistência sempre que um dos documentos é modificado são uma parte intrínseca de um ambiente integrado [12]. Ambientes integrados permitem o desenvolvimento e a manutenção incremental dos documentos. Apoiam uma aproximação evolucionária do desenvolvimento do software.

Este trabalho procura ajudar o desenvolvimento de sistemas multi-agentes, por meio de um ambiente que auxilie o desenvolvedor, oferecendo uma ferramenta visual de apoio à modelagem, de transformação e geração parcial de código.

1.1. Motivação

À medida que sistemas crescem, também cresce a complexidade e torna-se mais difícil satisfazer a um número cada vez maior de requisitos, bem como atender às restrições de orçamento e cronograma. Um sistema de software complexo se caracteriza por um conjunto de componentes abstratos de software (estruturas de dados e algoritmos) encapsulados na forma de procedimentos,

funções, módulos, objetos ou agentes interconectados entre si, compondo a arquitetura do software, que deverão ser executados em sistemas computacionais.

Os fundamentos científicos para a engenharia de software envolvem o uso de modelos abstratos e precisos que permitem ao engenheiro especificar, projetar, implementar e manter sistemas de software. Além disto, a engenharia de software deve oferecer mecanismos para se planejar e gerenciar o processo de desenvolvimento.

A modelagem e a especificação de software são atividades fundamentais para fazer do desenvolvimento de software uma atividade de engenharia. O uso de modelos nas diversas áreas da engenharia sempre foi fundamental para o desenvolvimento de bons produtos. Os modelos permitem uma visão antecipada do produto a ser desenvolvido e permitem análise e avaliação prévias, antes mesmo dos modelos serem construídos. O design do projeto é a atividade na qual a estrutura e o comportamento do software são elaborados. Durante esta fase, diversos modelos são desenvolvidos, antecipando diversas visões do produto final e possibilitando a avaliação do software antes mesmo que ele seja implementado. Diversas técnicas de modelagem e especificação de software que são utilizadas no design de software têm por objetivo permitir transformar requisitos em software de maneira a diminuir a distância entre a definição dos requisitos e a implementação do software.

No que se refere à definição e análise de um sistema multi-agentes, já existem diversas linguagens de modelagens, como: Gaia [15], MaSE [24], AUML [2], Message/UML [5], Tropos [20] e vários outros. Tais metodologias propõem diferentes formas no uso dos conceitos de agentes e suas tecnologias, em vários estágios do ciclo de vida do desenvolvimento. O laboratório LES da PUC-Rio possui algumas iniciativas no sentido de linguagens de modelagem para SMA, como MAS-ML [27] e o ANote [23].

No que se refere ao projeto e implementação de um sistema multi-agentes, também já existem diversas arquiteturas, como: ZEUS [3], JADE [26] e várias outras. A arquitetura de um agente mostra como ele está implementado em relação às suas propriedades, sua estrutura, e como os módulos que o compõem podem interagir, garantindo sua funcionalidade. Em suma, a arquitetura de um agente especifica sua estrutura e funcionamento. O laboratório LES da PUC-Rio também

possui algumas iniciativas para arquiteturas de desenvolvimento de sistemas visado para SMA, como o: ASYNC [12].

A necessidade de diminuir a distância entre a definição dos requisitos e a implementação do software gera muitas incompatibilidades. Tem se notado um esforço em iniciativas isoladas pra modelagem e implementação. É preciso diminuir esta distância com o uso de um Ambiente de Desenvolvimento de Software Multi-Agente (ADSMA), criando ferramentas integradas, de modo que a informação criada por uma ferramenta possa ser usada por outra.

Este trabalho adota o ANote como linguagem de modelagem para especificação de sistemas multi-agentes. Esta escolha foi feita por dois motivos: por ser uma linguagem de modelagem simples orientada a agentes e por possuir um meta-modelo e um conjunto de regras de boa formação que permitem tanto a verificação da corretude da modelagem quanto a geração parcial de código a partir de seus diagramas.

O ASYNC será adotado nesta dissertação como arquitetura de desenvolvimento para sistemas multi-agentes. Esta escolha também foi feita por dois motivos: por ser um framework orientado a objetos simples, implementado em JAVA, o que, para o propósito desta dissertação, facilita seu uso, e por já ter sido usado para o desenvolvimento de SMAs, como é o caso do sistema criado para o Trading Agent Competition (TAC) [25] que será apresentado como estudo de caso.

1.2. Objetivo

O objetivo desta dissertação é elaborar um ambiente de desenvolvimento de software que dê suporte computacional ao desenvolvimento de sistemas multi-agentes auxiliando o desenvolvedor durante o ciclo de vida de construção (da modelagem à implementação) destes sistemas.

Este ambiente é composto por 3 ferramentas que trabalham de forma independente, mas que juntas formam o ambiente esperado. Esta flexibilidade torna o ambiente mais simples, leve e dinâmico. Estas ferramentas são integradas, de modo que a informação criada por uma ferramenta pode ser usada por outra.

A primeira ferramenta corresponde a ferramenta para o apoio à modelagem visual com o ANote. Esta ferramenta representa graficamente todos os elementos

relacionados aos diagramas do ANote, incluindo as entidades (como agentes, objetivos, etc) e os relacionamentos entre estas entidades.

Além do apoio visual, esta ferramenta implementa o meta-modelo do ANote, responsável pela consistência da modelagem entre os diagramas durante a especificação do sistema. Desta forma, os tipos de entidades que podem existir em um modelo, suas relações e restrições de aplicação são verificados através de um conjunto de regras de boa formação.

A ferramenta permite a gravação e o armazenamento de modelos gerados. Estes modelos são armazenados segundo uma estrutura intermediária baseada no meta-modelo ANote, utilizando a tecnologia XML Metadata Interchange (XMI) [29]. Esta estrutura permite a interoperabilidade (exportação) de modelos, sendo o ponto de partida para a geração de código.

A segunda ferramenta dá apoio à transformação da estrutura intermediária dos diagramas para a arquitetura de desenvolvimento de sistemas multi-agentes ASYNC. Esta transformação ocorre através da configuração de um arquivo de extensão XML, responsável pelo mapeamento dos modelos (as estruturas intermediárias) envolvidos. Como resultado desta ferramenta é gerada uma nova estrutura intermediária baseada na arquitetura.

E, para finalizar, uma terceira ferramenta é responsável pela geração do código parcial da estrutura intermediária baseada na arquitetura, acelerando assim consideravelmente o processo de implementação, dado que a geração, embora crie código compilável, não cria código completo. Esta ferramenta possui marcadores no código gerado indicando o que falta ser implementado.

1.3. Contribuições

As contribuições principais esperadas para este trabalho são:

• Ferramenta visual para o apoio à utilização da linguagem de modelagem do ANote.

• Estrutura intermediária para a representação e armazenamento dos diagramas do ANote.

• Transformação dos diagramas do ANote para a arquitetura de desenvolvimento ASYNC.

1.4. Guia do Leitor

A seção 2 apresenta uma caracterização de Sistemas Multi-Agentes, uma breve introdução a Ambientes de Desenvolvimento de Software e trabalhos relacionados; na seção 3 apresenta a linguagem de modelagem ANote, o ASYNC e as tecnologias utilizadas para o desenvolvimento do ambiente; na seção 4 apresenta o processo de desenvolvimento e cada ferramenta do ambiente Albatroz detalhadamente; na seção 5 apresenta passo a passo o estudo de caso baseado no sistema Learn Agent; finalmente, a seção 6 apresenta a conclusão do trabalho.