1
Introdução

Nada é mais poderoso do que uma idéia
cujo momento chegou.
Victor Hugo

O uso do paradigma de orientação a objetos (OO) no desenvolvimento de sistemas de software de grande porte está estabelecido, assim como o uso de linguagens de modelagem como UML (Booch et al,1999) durante as atividades de modelagem destes sistemas. Modelos descritos em UML são compostos por conjuntos de diagramas que capturam diferentes visões do sistema, focando em seus aspectos estruturais, funcionais e comportamentais. Assim, modelos que representam aplicações reais de grande porte, desenvolvidas usando o paradigma OO, costumam ser compostos por conjuntos complexos de diagramas. Esta complexidade torna difícil a análise da boa formação destes diagramas, mesmo quando considerados independentemente do domínio da aplicação.

A complexidade dos sistemas atuais tem aumentado, especialmente quando considerados os sistemas de natureza distribuída, heterogênea, autônoma e aberta. Esta natureza é demandada para disponibilizar uma quantidade cada vez maior de informações, a qualquer hora e em qualquer lugar, a fim de satisfazer o desejo de uma sociedade cada vez mais conectada. Para lidar com estas características, agentes de software passaram a coexistir com objetos em soluções para este tipo de sistemas, ou seja, a abordagem multiagentes tem sido adotada com freqüência como solução para aplicações distribuídas, heterogêneas e autônomas de grande porte, notadamente no setor de logística (Luck et al, 2005).

Neste contexto, a adoção de uma abordagem multiagente como solução para estes sistemas gerou demanda por extensões de linguagens de modelagem, estilo UML, para acomodar as abstrações relacionadas aos agentes de software, tais como papéis, organizações e ambientes (Zambonelli et al, 2001; Silva et al, 2003). Assim, se a análise da boa formação de diagramas que modelam aplicações OO de grande porte já é um problema complexo, cuja

sistematização tem sido motivo de pesquisa fecunda (pUML, 2005; Kim & Carrington, 2000; Berardi et al, 2001; Berardi, 2002; Straeten et al, 2003; Enkenberg & Johannesson, 2004), mais complexo este problema fica quando considerada uma linguagem de modelagem que estende UML com a introdução de meios para modelar as abstrações relacionadas às características de agência. A análise de modelos de sistemas multiagentes (SMAs) descritos numa linguagem de modelagem que estende UML é, de fato, bastante complexa, e pode comprometer a adoção da tecnologia de agentes. Consequentemente, a definição de métodos e metodologias que facilitem a análise destes modelos, a fim de auxiliar os desenvolvedores a detectar inconsistências nestes modelos de forma automática, é bem-vinda.

1.1. Descrição do problema

Linguagens de modelagem são aliadas importantes no desenvolvimento de sistemas de software. Sua característica predominantemente gráfica facilita a troca de informações e o entendimento da estrutura e do comportamento do sistema entre o grupo de desenvolvedores. Quando o sistema a ser desenvolvido envolve um grau de complexidade elevado, mais importante se faz a utilização de modelos de alto nível de abstração que descrevam visões do sistema.

Além disso, linguagens de modelagem fornecem modelos que representam visões distintas de partes do sistema, possibilitando a abstração do todo a fim de dirigir o foco a questões específicas e que demandem maior atenção do desenvolvedor. Assim, o uso de linguagens de modelagem no desenvolvimento de sistemas de software pode ser considerado um fator responsável pelo aumento da velocidade do desenvolvimento e da qualidade do produto final (Mills &Gomaa, 2002).

O desenvolvimento de SMAs envolve um grau de complexidade alto dadas as características específicas dos agentes de software que os compõem. Estas características podem ser explicadas pelas novas entidades adicionadas a estes sistemas, pelos novos relacionamentos entre estas entidades, e pelas novas propriedades inerentes a estas entidades. Desta forma, linguagens de modelagem que possibilitem a descrição das especificidades inerentes a SMAs beneficiam o desenvolvimento destes sistemas ao definir diagramas capazes de representar todas as suas características. Na última década várias linguagens de

modelagem para SMAs foram desenvolvidas, tais como AUML (Bauer et al, 2002), AORML (Wagner, 2002), MAS-ML (Silva & Lucena, 2004) e ANote (Choren & Lucena, 2004). Ao possibilitar a descrição de especificidades inerentes a SMAs, estas linguagens tornam-se mais complexas que as linguagens de modelagem orientadas a objetos. O aumento da complexidade atribuído às linguagens de modelagem para SMAs deve-se não só ao aumento do número de entidades e relacionamentos previstos na linguagem, mas também à necessidade de criação de novos diagramas que descrevam a estrutura e o comportamento das entidades que compõem os SMAs.

O número de entidades e relacionamentos contidos em modelos descritos em linguagens de modelagem, acrescido das relações de interdependência entre os vários diagramas, podem ser considerados catalisadores do surgimento de inconsistências intra-diagramas e inter-diagramas. Define-se inconsistências intra-diagramas como aquelas relacionadas às violações a propriedades internas dos diagramas. Inconsistências inter-diagramas são definidas como inconsistências resultantes de violações às propriedades de interdependência entre os diagramas.

As diferentes visões do sistema definidas utilizando-se os diagramas propostos pela linguagem de modelagem são, em geral, interdependentes. Algumas entidades do sistema participam de várias visões aumentando a probabilidade de surgirem inconsistências inter-diagramas. Por outro lado, o acréscimo de novas entidades e relacionamentos na descrição de diagramas que modelam um SMA também aumenta a probabilidade de surgirem inconsistências intra-diagramas.

A fim de evitar estes tipos de problemas torna-se importante o desenvolvimento de métodos de análise formal de modelos definidos nas linguagens de modelagem. No caso de linguagens de modelagens para SMAs em particular, o desenvolvimento destes métodos se faz importante seja devido à maior complexidade das próprias linguagens, seja devido ao suporte que estas técnicas podem prover para ferramentas que apóiem o desenvolvimento de modelos usando tais linguagens. O uso de ferramentas de suporte ao desenvolvimento de modelos usando linguagens de modelagem para SMAs deve permitir a checagem automática de algumas propriedades estruturais dos modelos ainda durante a sua construção.

A definição de técnicas e métodos de análise formal de modelos descritos em linguagens de modelagem pressupõe uma descrição formal destes modelos, o que viabiliza a automatização do processo de análise, permitindo maior

velocidade na geração dos modelos, assim como maior qualidade e padronização dos modelos gerados (Mills & Gomaa, 2002).

Neste contexto, considere uma linguagem de modelagem orientada a agentes, que estende uma linguagem de modelagem orientada a objetos, estilo UML, e os diagramas que ela define. Considere, ainda, o fato de que linguagens de modelagens são baseadas em metamodelos (Henderson-Sellers, 2002; Evermann & Wand, 2005), que especificam as propriedades internas (ou estruturais) dos diagramas, assim como as propriedades de interdependência entre estes diagramas.

O problema tratado nesta tese consiste em, dada uma linguagem de modelagem orientada a agentes, juntamente com o metamodelo que a originou, definir meios que possibilitem a descoberta e indicação automática de inconsistências intra-diagramas e inter-diagramas descritos na linguagem de modelagem para SMAs.

1.2. Solução proposta

Como solução para o problema citado, foram propostas ontologias que descrevem o domínio de SMAs e um método para estruturação e análise de modelos de SMAs baseado nestas ontologias, chamado Observed-MAS (*Ontology-Based method for Structuring and analysing the Design of MASs models*).

As ontologias são baseadas num arcabouço conceitual definido para o domínio de SMAs (Silva et al, 2003). Desenvolveu-se duas ontologias: a primeira considerando-se apenas as principais entidades e relacionamentos que ocorrem entre estas entidades; e a segunda como extensão da primeira, considerando-se as restrições inerentes a cada entidade e relacionamento, acrescentando-se à primeira ontologia axiomas que descrevem tais restrições. A partir da instanciação destas ontologias é possível analisar se a instância satisfaz determinadas propriedades que um SMA deve satisfazer. Esta análise pode ser feita através de consultas feitas na base de conhecimento gerada pela instância estruturada pela primeira ou segunda ontologia, dependendo da propriedade analisada.

Partindo-se do pressuposto que SMAs possuem propriedades específicas distintas de outras abordagens usadas no desenvolvimento de sistemas, como, por exemplo, orientação a objetos ou aspectos, é esperado que linguagens de

modelagens para SMAs sejam capazes de descrever as propriedades de SMAs, além das propriedades comuns a qualquer sistema de software. Assim, as ontologias desenvolvidas para descrever o domínio de SMAs foram usadas para embasar a definição do método Observed-MAS. O método faz uso das ontologias de SMAs e da forma como se define as consultas na base de conhecimento para análise das propriedades de SMAs para, integradas a ontologias que descrevem o metamodelo que originou a linguagem de modelagem, permitir a análise de modelos descritos na linguagem. Esta análise é relativa à conformidade com as propriedades do domínio de SMAs e com as regras de boa formação dos diagramas da linguagem. Diz-se que o método também serve para a estruturação dos modelos pois as ontologias formalizam e validam a estruturação proposta pelo metamodelo da linguagem de modelagem.

O Observed-MAS é composto por duas fases: uma destinada à análise das propriedades internas dos diagramas e a outra destinada à análise das propriedades de interdependência entre os diagramas. Além disso, ambas as fases são dotadas de mecanismos que permitem a sugestão de boas práticas de modelagem de SMAs usando a linguagem na qual os diagramas foram descritos.

Para dar suporte formal à análise, cada fase conta com uma ontologia (Ont1 e Ont2) e um conjunto pré-definido de consultas (QV1 e QV2). Na primeira fase a ontologia (Ont1) é resultante da integração da primeira ontologia definida para o domínio de SMAs com o núcleo do metamodelo da linguagem. As consultas (QV1) são definidas para retornar como resposta indivíduos que violem as propriedades de SMAs e as propriedades intra-diagramas. Os diagramas da linguagem são mapeados para instâncias da ontologia e, então, analisados por consultas previamente definidas quanto à sua consistência. Ainda na primeira fase também é definido outro conjunto de consultas (QD1). Estas consultas são definidas para sugerir ao desenvolvedor boas práticas de modelagem, baseadas nas caracterísiticas específicas de SMAs e na forma como elas são especificadas nos diagramas da linguagem de modelagem, quando considerados individualmente.

A segunda fase define uma ontologia (Ont2) que estende a ontologia da primeira fase através da adição de axiomas que descrevem as propriedades intra-diagramas. Estas propriedades devem refletir a análise obtida a partir das consultas (QV1) definidas na primeira fase. Observa-se que Ont2 também é resultado da integração da segunda ontologia definida para o domínio de SMAs os axiomas que especificam as propriedades internas dos diagramas da linguagem de modelagem, de acordo com seu metamodelo. As consultas (QV2)

definidas na segunda fase são especificadas para devolver como resposta indivíduos que violem as propriedades de interdependência entre os diagramas, considerando-se a modelagem do SMA como um todo. De forma análoga à primeira fase também é definido na segunda fase outro conjunto de consultas (QD2), a fim de sugerir boas práticas de modelagem ao desenvolvedor que faz uso do método instanciado. Estas consultas são baseadas nas especificidades de SMAs e na forma como elas são descritas pela linguagem de modelagem que instancia o método, quando considerados todos os diagramas que compõem a modelagem de um SMA.

O uso de ontologias como formalismo de descrição dos modelos de SMAs justifica-se pela possibilidade de emprestar, da engenharia de conhecimento, o suporte necessário para análise automática de consistência das ontologias e das bases de conhecimento estruturadas pelas ontologias. Além disso, os serviços de inferência/consultas disponibilizados pelos sistemas de representação de conhecimento agregam maior poder de análise, possibilitando a descoberta de propriedades implícitas ou explícitas dos indivíduos que populam a base de conhecimento. Uma propriedade é dita implícita se ela não foi expressamente descrita na ontologia. Outra justificativa para o uso de ontologias é a possibilidade de descrição de quaisquer tipos de relacionamentos entre quaisquer entidades.

A existência de ferramentas que possibilitam a automatização do processo de análise de modelos é um fator importante, pois aumenta a probabilidade de adoção do método pela comunidade de desenvolvedores de SMAs. A falta de suporte automático foi motivo de interrupção de uma pesquisa anterior para análise de modelos MAS-ML, quando foi definida uma notação formal chamada AgentZ (Brandão et al, 2005c), que estende a linguagem Object-Z (Smith, 1999) para o domínio de SMAs. No caso de ontologias, não só existem diversas ferramentas gratuitas disponíveis para suporte à construção, edição, manutenção e classificação de ontologias (Protege, 2004; OilEd, 2002), como também uma série de metodologias de desenvolvimento de ontologias, para auxiliar não especialistas em engenharia de conhecimento na tarefa de construção de ontologias. As ferramentas citadas são, em geral, conhecidas como sistemas de representação de conhecimento e utilizam tecnologias estabelecidas para criação, armazenamento e manipulação do conhecimento estruturado pelas ontologias.

Apesar de plenamente justificável, o uso de ontologias também pode ser um fator limitante do método, pois a eficiência computacional dos sistemas de

representação de conhecimento é fortemente dependente da expressividade da linguagem de descrição usada para formalizar as ontologias e também ao tamanho das bases de conhecimento geradas a partir de uma ontologia. Assim, para contornar a questão da expressividade foi usada, na especificação das ontologias definidas pelo método, uma descrição formal parcial, ou seja, os axiomas considerados como restrições da ontologia usam inclusão simples na definição de conceitos, no lugar de equivalências entre conceitos.

1.3. Principais Contribuições

Dentre as principais contribuições cita-se:

- a descrição do domínio de SMAs através de ontologias que formalizam o arcabouço conceitual TAO (Silva et al, 2003);
- a definição do método Observed-MAS;
- a descrição formal parcial do metamodelo de MAS-ML;
- a implementação de uma ferramenta de suporte à análise automática de modelos MAS-ML usando o Observed-MAS;
- a definição de propriedades de boas práticas de modelagem de SMAs.

As ontologias que descrevem o domínio de SMAs proporciona a formalização do arcabouço conceitual TAO. O TAO serviu de suporte para a definição de uma linguagem de modelagem para SMAs (Silva, 2004) e para a definição de um método arquitetural orientado a aspectos e de uma linguagem de padrões associados, por exemplo, às propriedades de autonomia e mobilidade de agentes (Garcia, 2004). Tanto a linguagem de modelagem, quanto o método arquitetural e a linguagem de padrões podem se beneficiar desta formalização.

O método Observed-MAS permite a estruturação e análise de modelos de SMAs descritos em linguagens de modelagem, a partir: (i) da formalização destes modelos como instâncias de ontologias baseadas no domínio de SMAs e no metamodelo da linguagem de modelagem; e (ii) do uso de sistemas de representação de conhecimento para proceder à análise destes modelos.

A efetiva aplicação do método para a linguagem MAS-ML proporciona a descrição formal parcial do metamodelo de MAS-ML, pois ela pressupõe sua descrição através da integração das ontologias que formalizam o TAO com a

ontologia que descreve o metamodelo da linguagem na qual o método é aplicado.

A implementação de uma ferramenta de suporte à utilização do Observed-MAS para a linguagem MAS-ML propicia maior agilidade durante a atividade de análise dos modelos, estimulando o uso da linguagem e diminuindo o tempo de desenvolvimento destes modelos.

A definição de propriedades de boas práticas de modelagem de SMAs é baseada em propriedades específicas a este tipo de sistemas, como por exemplo, interação entre agentes. Estas propriedades permitem a definição, para cada linguagem de modelagem na qual o método for aplicado, de consultas cujas respostas auxiliem o desenvolvedor durante a construção de modelos complexos. Além disso, estas propriedades podem servir como indicativos para a definição de padrões de projeto que descrevam características específicas a SMAs.

1.4. Organização da tese

Esta tese está organizada da seguinte maneira. O capítulo 2 contém os fundamentos teóricos e as tecnologias utilizadas, apresentando noções sobre ontologias, lógica de descrição, sistemas multiagentes e linguagens de modelagem orientadas a agentes. No capítulo 3 encontram-se alguns trabalhos relacionados, divididos em duas áreas de conhecimento. O capítulo 4 apresenta as ontologias que descrevem o domínio de SMAs através da formalização de TAO e, como aplicação destas ontologias, apresenta a descrição geral do método. Uma prova de conceito para aplicabilidade do método é apresentada no capítulo 5 para a linguagem MAS-ML, assim como uma ferramenta de suporte para o método aplicado. Estudos de caso realizados para a prova de conceito usando MAS-ML são descritos no capítulo 6. No capítulo 7 é feita uma aplicação do método para outra linguagem de modelagem orientada a agentes (AUML), com um pequeno estudo de caso. Finalmente, o capítulo 8 apresenta as conclusões, principais contribuições da tese e trabalhos futuros.