FERRAMENTA MAS-ML: UM AMBIENTE DE MODELAGEM PARA SISTEMAS MULTIAGENTES

Abstract: Os Sistemas Multi-Agente (MAS) surgiram como uma abordagem promissora para o desenvolvimento de sistemas complexos e distribuídos. No entanto, as ferramentas que suportam o desenvolvimento de MASs são essenciais para que esta abordagem seja efetivamente explorada no contexto industrial. Portanto, há necessidade de ferramentas para a modelagem de SMA, pois criar e manipular modelos sem o suporte de um ambiente adequado são tarefas tediosas e sujeitas a erros que demandam tempo. Este artigo visa atender a essa necessidade construindo um domínio de ambiente de modelagem específico para MAS, implementado como um plugin para a plataforma Eclipse. O ambiente é baseado em MAS-ML, uma linguagem de modelagem para MAS. Este trabalho tem como foco a implementação de ferramenta para diagramas estáticos MAS-ML, conforme a versão 2.0 da linguagem.

1. Introdução

A indústria de software e o meio acadêmico pesquisam e fornecem tecnologia para atender a demanda de construção de sistemas de software cada vez mais complexos. Nesse cenário, os Sistemas Multi-Agente (MAS) surgiram como abordagem promissora na tentativa de melhor gerenciar essa complexidade. De acordo com Jennings e Wooldridge [Jennings e Wooldridge 2000], MAS podem ser entendidos como sociedades de agentes onde entidades heterogêneas e autônomas que podem trabalhar juntas para propósitos semelhantes ou totalmente diferentes. MAS se tornou um paradigma poderoso para engenharia de software [Mubarak 2008] e têm sido usados com sucesso para o desenvolvimento de diferentes tipos de sistemas [Lind 2001] [Wooldridge e Ciancarini 2001]. Nesse cenário, as linguagens e ferramentas de modelagem MAS têm um papel central no processo de desenvolvimento.

A possibilidade de um único SMA pode abranger vários tipos de agentes com diferentes arquiteturas internas (Weiss, 1999) justificam a existência de uma linguagem para apoiar a modelagem de diferentes agentes internos
arquiteturas. Neste contexto, o MAS-ML (Multi-Agent System Modeling Language) [Silva, Choren e Lucena 2007] foi atualizado para cumprir este requisito, resultando no MAS-ML 2.0 [Gonçalves et al. 2010].

Recomenda-se a utilização de ferramentas CASE para apoiar os processos de engenharia de software a fim de automatizar as atividades envolvidas. Em particular, a ferramenta de modelagem aumenta a produtividade e pode ser útil para garantir a boa construção dos modelos.

As ferramentas de modelagem são projetadas para suportar os recursos e a mecânica relacionados a uma linguagem de modelagem específica.

MAS-ML define três diagramas estruturais: diagrama de classes, diagrama de papéis e diagrama de organização; e dois diagramas dinâmicos: sequência e atividades [Silva, Choren e Lucena 2007]. Este trabalho visa a implementação de um ambiente de modelagem para suporte os diagramas estáticos MAS-ML com base no metamodelo MAS-ML 2.0.

Este artigo está organizado da seguinte maneira. A seção 2 apresenta uma teoria relacionada à linguagem MAS-ML. Na Seção 3 o ambiente é apresentado, descrevendo alguns de seus benefícios, limitações, mencionando algumas questões de implementação. Na Seção 4 é apresentado um estudo de caso. Na Seção 5, os trabalhos relacionados são confrontados com as contribuições desses artigos. Por fim, a Seção 6 apresenta as conclusões e trabalhos futuros.

2. MAS-ML 2.0

MAS-ML 2.0 [Gonçalves et al. 2010] é uma extensão da linguagem de modelagem MAS-ML [Silva, Choren e Lucena 2007] para apoiar a
modelagem de: (i) agentes reflexos simples, (ii) agentes reflexos baseados em modelos e (iii) agentes baseados em objetivos com o planejamento e (iv) agentes baseados na utilidade.

Em termos práticos, a extensão acima mencionada envolveu a criação de duas metaclasses: AgentPerceptionFunction, que representa as percepções do agente e AgentPlanningStrategy, que representa o agente de planejamento. Ambas as classes são especializações da metaclasse BehavioralFeature da UML. Além disso, quatro estereótipos foram criados: formular-objetivo-função que representa a formulação do objetivo do agente; formular-problema-função que representa a formulação do problema; função seguinte, que representa a atualização das crenças do agente; e função-utilidade que representa o grau de utilidade com base na ação atual [Gonçalves et al. 2010].

A partir dos novos elementos no metamodelo, a representação do agente nos diagramas MAS-ML aumentou quatro variantes gráficas, onde cada uma representa cada uma das arquiteturas internas mencionadas acima. Em consistência com as novas representações do agente, a representação do papel do agente foi associada a três representações: (i) a representação MAS-ML original, (ii) uma representação sem objetivos, relacionada a agentes reflexivos baseados em modelos, e (iii) uma representação sem objetivos e crenças, relacionados a agentes reflexos simples. o
Os diagramas MAS-ML foram modificados relacionados aos novos recursos para a modelagem das arquiteturas de agentes internos.

3. DESENVOLVIMENTO DA FERRAMENTA MAS-ML

Esta seção apresenta as funções, tecnologias e detalhes relacionados ao desenvolvimento do ambiente de modelagem de domínio específico, ferramenta MAS-ML.

A abordagem orientada por modelo foi usada, onde o modelo central e a abstração maior é o metamodelo auto MAS-ML. O metamodelo representa o ponto inicial do processo de derivação que ocorre ao longo de um conjunto de transformações. Cinco etapas realizadas durante o desenvolvimento são descritas a seguir:

Modelo de Domínio - primeiro, o metamodelo MAS-ML foi especificado usando o EMOF (Essential Meta-Object Facility), uma linguagem de definição de metamodelo. Os estereótipos foram adicionados ao ActionClass pelo recurso ActionSemantics, esta semântica apresenta as opções: 0- sem estereótipo, 1- próxima função, 2 - função utilidade, 3- formular função-problema e 4- formular função objetivo.

Modelo de Definição Gráfica - Nesta etapa são definidas as entidades e suas propriedades, e relacionamentos. As entidades e relacionamentos do metamodelo foram usados.

Modelo de definição de ferramentas - nesta etapa são especificados quais elementos existirão na paleta de ferramentas. Esta etapa recebe o modelo de domínio e o modelo de definição citados anteriormente.

Modelo de mapeamento - nesta etapa, um mapeamento entre o modelo de domínio, o modelo gráfico e o modelo de ferramentas é construído :. O mapeamento gerado foi utilizado como entrada do processo de transformação, que objetivou criar um modelo específico de plataforma. Um conjunto de seis regras de validação definidos usando OCL (Object Constraint Language) são usados para verificar se o modelo foi formado corretamente (Tabela 1).

Geração de Tooling - A próxima etapa, segundo a abordagem generativa [Czarnecki e Eisenecker, 2000], é a geração do código de acordo com o modelo criado na última etapa. O GMF (Graphical Eclipse Framework) [GMF, 2011] é usado, o que fornece um gerador componente e uma infraestrutura de tempo de execução para desenvolver editores gráficos. Siga cada desenvolvimento do diagrama será descrito.

3.1. Desenvolvimento de diagrama de classes

A ferramenta MAS-ML Class Diagram resultante disponibiliza os seguintes elementos: 1) Nós: Classe, AgentClass, OrganizationClass, EnvironmentClass, ActionClass, PlanClass, Property, Operation, goal,

crença, percepção e planejamento; 2) Relacionamentos: Associação, Habitar, Dependência, Generalização, Agregação e Composto 3) Notas.

Além disso, a ferramenta pode validar os diagramas de acordo com as regras de geração. Essas regras validam a representação das arquiteturas internas.

3.2. Desenvolvimento do diagrama organizacional

Os resultados do desenvolvimento do diagrama de classes foram usados para criar o diagrama da organização. Além disso, as regras do agente e as regras do objeto foram representadas de acordo com MAS-ML 2.0 e os relacionamentos propriedade e jogo, parte do diagrama de organização, foram adicionados. O habitar relacionamento tem a semântica alterada para permitir que agentes, regras de agente e organização habitem o ambiente. A associação, dependência, generalização, agregação e composição foram removidas. Esses novos elementos estavam no modelo de domínio e no modelo gráfico, mas não eram usados no diagrama de classes (Seção 3.1).

3.3. Desenvolvimento de diagrama de papéis

Os resultados do diagrama de classes e do desenvolvimento do diagrama de organização foram usados para criar o diagrama de funções, uma vez que algumas entidades são iguais em ambos os diagramas. Assim, também foi utilizado o metamodelo MAS-ML 2.0.

Alguns elementos foram preservados: Classe, Função do Agente e Função do Objeto. Da mesma forma, os relacionamentos de Associação, Controle, Dependência, Generalização e Agregação. A representação gráfica dos elementos, relacionamentos e diagramas da ferramenta MAS-ML são apresentados na próxima seção por meio de um estudo de caso.

4. Estudo de Caso

Esta seção apresenta um MAS para Moodle usando a ferramenta MAS-ML. Inicialmente o Moodle será descrito e após a modelagem estará presente.

4.1. Moodle

O uso de ferramentas computacionais tem um impacto positivo nas atividades educacionais. Professores, alunos e o sistema interagem por meio de recursos tecnológicos, compartilhando um

mesmo espaço de trabalho e resolvendo problemas de forma conjunta, apoiados em tecnologias de comunicação à distância. Normalmente, os ambientes de aprendizagem colaborativa enfatizam a Comunicação Mediada por Computador (CMC), com ferramentas que possibilitam síncrona (salas de chat, videoconferência) e assíncrona (e-correio, quadro branco).

Nesse contexto destaca-se o Ambiente Virtual de Aprendizagem MOODLE [MOODLE, 2011]. Baseia-se no construcionismo social e assume que as pessoas aprendem melhor quando estão engajadas de forma colaborativa em um processo social de construção do conhecimento.

4.2. Modelando um MAS para Moodle com a ferramenta MAS-Ml

Seis agentes foram propostos ao Moodle:

LearningPartnerAgent, SearcherInformationAgent, PedagogicAgent, UsageHelperAgent, TeamMakerAgent e CoordinatorAgent. A seguir, esses agentes são descritos e cada modelo de ferramenta MAS-ML é apresentado.

LearningPartnerAgent (Figura 1): Modelado como um agente de reflexo baseado em modelo. Este agente seleciona mensagens de apoio e reforço para os alunos apresentarem com base nas dificuldades e acertos que tem nas discussões e / ou nas tarefas e / ou conteúdos propostos.

PedagogicAgent (Figura 2): É um agente baseado em objetivos com planejamento. Sua função é auxiliar o aluno por meio de mensagens relacionadas ao tema em que está envolvido em diversos cursos e disciplinas que participam. Também sugeriu cursos e assuntos relacionados aos interesses do aluno.

UsageHelperAgent (Figura 3): Modelado como um agente de reflexo simples. É responsável por fornecer dicas de como fazer melhor uso de ferramentas específicas.

TeamMakerAgent (Figura 4): Modelado como um agente baseado em utilidade, sua função é formar ou ingressar em grupos de acordo com o assunto proposto ou perfil de aprendizagem sugerido pelo treinador.

SearcherInformationAgent (Figura 5): Objetivo baseado em agente com plano. Este agente é responsável por localizar pessoas dentro do ambiente Moodle que estão envolvidas em disciplinas e grupos relacionados ao mesmo

tópico do aluno. Além disso, este agente busca documentos (páginas, projetos e outros objetos digitais) que se relacionem com o tema de interesse.

CoordinatorAgent (Figura 6): Modelado como um agente baseado em metas com planejamento, este agente deve ser o responsável por ordenar as ações dos demais agentes, mediando assim a mesma conversa.

Seis funções de agente associadas a um agente apresentadas foram propostas ao Moodle: LearningPartner (Figura 7), Pedagogic (Figura 8), TeamMaker (Figura 9), SearcherInformation (Figura 10), UsageHelper (Figura 11) e Coordenador (Figura 12). A seguir, o modelo de papel do agente na ferramenta MAS-ML é apresentado.

Finalmente, a Figura 13 mostra o Diagrama de Funções para Moodle MAS e a Figura 14 mostra o Diagrama de Organização para Moodle MAS.

5. Trabalhos Relacionados

Considerando um amplo escopo em relação às ferramentas de suporte à modelagem de SMAs [AgentTool 2011] [Padgham, Winikoff Thangarajah e 2008]. No entanto, uma questão chave é que as ferramentas são projetadas para apoiar a construção do diagrama em uma linguagem de modelagem específica. Assim, as vantagens e limitações dessas linguagens são propagadas para as ferramentas que as implementam.

Considerando as ferramentas de modelagem já existentes relacionadas ao MAS-ML, VisualAgent [De Maria et al. 2005] é um ambiente de desenvolvimento de software que visa auxiliar os desenvolvedores na especificação, projeto e implementação de MASs.

VisualAgent é baseado no metamodelo MAS-ML original. Consequentemente, o suporte à modelagem para agentes com diferentes arquiteturas internas pode ser limitado. VisualAgent nenhum mecanismo de verificação de modelo é fornecido. A ausência desse recurso em VisualAgent pode comprometer a qualidade dos modelos e do código gerado. Além disso, a falta de documentação e acesso ao código fonte dificulta a continuidade do projeto.

6. Conclusões

Este artigo apresenta uma ferramenta que representa uma prova de conceito da linguagem de modelagem MAS-ML, com foco em diagramas estáticos. Em sua versão atual, o MAS-ML 2.0 incorpora recursos para modelar agentes racionais de maneira adequada, proporcionando um melhor

nível de abstração para representar as características internas dos SMAs. Com o ambiente é possível, apoiar a atividade de modelagem, verificar a veracidade da construção dos modelos e manter sua persistência. No contexto do desenvolvimento orientado ao modelo, os diagramas podem ser usado no processo de transformação para geração de código em plataformas de agentes específicos.

Como trabalhos futuros, algumas melhorias podem ser feitas na representação gráfica dos construtores para representar com mais fidelidade a representação proposta no metamodelo MAS-ML. Além disso, suporte para modelagem de diagramas dinâmicos do MAS-ML 2.0.