1
Introdução

Os Engenheiros de sistemas multi-agentes (SMA) além de lidarem com o projeto e a implementação de seus requisitos, constantemente lidam com características como autonomia, interação e aprendizagem que se encontram embuidas nos SMAs.

Um dos principais pontos no desenvolvimento de um projeto é a modelagem do sistema e, em se tratando de SMA, inclui-se também a modelagem dos agentes de software do sistema. A consagração da UML como linguagem "padrão" para a modelagem de sistemas orientados a objeto já é fato, porém ainda não há uma linguagem consagrada ou mais indicada para a modelagem dos agentes, até mesmo por ser bastante recente e ainda estar sofrendo mudanças.

A escolha de uma linguagem de modelagem torna-se ainda mais imprescindível quando se trata de agentes de software, já que está linguagem contribuirá de maneira a auxiliar e minimizar a complexidade do desenvolvimento de SMA.

As atuais linguagens de modelagem de agentes de software trazem consigo a possibilidade de descrever os agentes, seus papéis, planos, ações, comportamentos e objetivos dentro dos sistemas. O desenvolvimento do plano de um agente muitas vezes requer a implementação de ações internas ao mesmo, as quais são executadas a fim de alcançar o objetivo do agente no sistema.

Assim como *frameworks* orientados a objeto os agentes de software apresentam pontos de flexibilização em seu desenvolvimento.

Muitas vezes, em sistemas complexos, são detectados pontos de flexibilização em alguns agentes, porém não são representados na modelagem por limitações das linguagens atuais. Com isso percebeu-se que as modelagens de tais sistemas só representam aplicações inteiras, impossibilitando a representação de uma linha de produto construída através de *framework* o que exigiria, agentes adaptáveis por customização e extensão da estrutura que eles provêem assim como em um *framework*.

Caso sejam utilizadas as ferramentas atuais para obter uma maior aproximação do código com a modelagem provavelmente será necessário utilizar artifícios não definidos pela linguagem.

As atuais linguagens de modelagem de agentes trazem um modelo não preditivo (do ponto de vista de não adaptar as partes relevantes durante o processo de instanciação).

O trabalho aqui proposto visa representar pontos de flexibilização em planos e ações de agentes e com isso possibilitar que o sistema modelado com o método proposto possa apresentar uma estrutura de *framework* para agentes de software. Além disso, o método também poderá ser utilizado juntamente com as linguagens de modelagens de agentes.

Para alcançar este objetivo, a solução proposta utilizará diagramas para representar o design do agente, seus planos e instâncias, além de uma técnica de documentação baseada em RDL, com a qual podemos documentar todas as mudanças planejadas na construção do agente, (assim como em um *framework*), tornando a implementação do agente mais simples e servindo de guia para a futura instancias de aplicações.

O uso de técnicas como RDL, *hooks*, etc., é bastante útil, pois possibilita documentar a descrição dos problemas e requisitos (Froehlich et al., 1998) de construção do agente (*framework*). Com isto será possível antecipar o desenvolvimento de aplicações obtendo também um guia para futuras instanciações.

Desta forma, a dissertação visa um meio de modelar agente de forma que seja possível definir uma linha de produto para SMA através de uma visão de um *framework*.

1.1. Descrição do Problema

A abordagem multi-agentes tem sido apontada como adequada para o desenvolvimento de sistemas de software complexos (Almeida et al., 2005) de forma que tem havido grandes investimentos em engenharia de software para que esta possa dar suporte à nova abordagem. Segundo Jennings & Wooldridge (2001), “um agente de software é um sistema de computador encapsulado que está

situado em algum ambiente, e é capaz de ter uma ação autônoma flexível nesse ambiente para encontrar seus objetivos no projeto". A idéia principal de um sistema multi-agente é que o comportamento global do sistema seja alcançado através da cooperação e do comportamento individual de cada agente. Os SMA são sistemas que contém um grupo de agentes de software os quais, munidos de algumas das características como pró-atividade, re-atividade e autonomia executam suas tarefas a fim de cumprirem os objetivos a eles estabelecidos.

O desenvolvimento deste tipo de sistema envolve uma complexidade não encontrada no desenvolvimento de softwares comuns, já que os agentes de software possuem comportamentos imprevisíveis, o que torna a abordagem mais complexa (Chavez, 2004). Além disso, podem ser abertos, ou seja, comunicar-se com sistemas externos. O desenvolvimento de agentes de software traz consigo algumas facilidades como a possibilidade de descrever os planos dos agentes e implementar dentro destes planos às ações que devem ser efetuadas pelos agentes. Muitas vezes as ações dos agentes de software determinam uma enorme variabilidade de opções.

A variabilidade ocorre quando, dentro de um objetivo impar obtém-se caminhos distintos e através da execução destes, o agente alcança o seu objetivo e conseqüentemente coopera para atingir o objetivo do sistema.

Atualmente as variabilidades não são representadas pelas linguagens de modelagens de agentes atuais, ou seja, só são modelados sistemas de uma única aplicação o que é uma grande falha do ponto de vista de reutilização.

Por meio da solução proposta será possível obter uma maior riqueza no que diz respeito à modelagem da variabilidade interna dos agentes de software. O método proposto foi concebido de forma a não estar atrelado a nenhum meta-modelo[1] de linguagem de modelagem, o que traz a possibilidade de utilização em conjunto com uma linguagem de modelagem de agente facilitando a abordagem do ponto de vista do responsável pela modelagem que poderá utilizar a sua linguagem preferencial.

Através do método proposto será possível dar suporte a construção de *framework* em um sistema multi-agente visando o desenvolvimento de uma linha

[1] O nível de meta-modelo provê construções para modelar o nível de conhecimento de entidades e conceitos.

de produto, para isto será necessária à representação das variabilidades internas de um agente em função dos pontos de flexibilização do agente.

A solução tornará o desenvolvimento de SMA mais simplificados e coesos do ponto de vista de deixar claro quando o sistema poderá ter uma linha de produto e quando não.

A solução proposta trará vantagens como à definição de linha de produto e *framework* para o desenvolvimento de SMA e com isto será possível obter vantagens como a redução no custo de desenvolvimento.

1.2. Motivação

O primeiro ponto de motivação para este trabalho veio da vontade de modelar o sistema CQPM (um sistema médico que será discutido em detalhes mais a frente) de modo que este pudesse vir a ter duas instâncias, uma para clínicas e outra para uso em CTI.

O sistema CQPM tem uma característica que foi fundamental para motivar a pesquisa que é o sistema possui um ponto de flexibilização num de seus agentes, onde tal agente para cumprir seu objetivo (em ambos os escopos clínica e CTI) precisaria implementar ações completamente distintas. Pensando que não são todos os locais (hospitais) têm setores de CTI e clínica, seria um desperdício não utilizar a idéia de *framework*, já que todo o sistema é igual, sofrendo mudanças apenas no plano do agente e as ações executadas em cada setor (clínica ou CTI).

O segundo ponto de motivação é que nenhuma linguagem de modelagem traz uma representação para sistemas com a característica de variabilidade, ou seja, tais linguagens não trazem uma maneira de representar a situação descrita.

O terceiro ponto de motivação para o trabalho é a diminuição do custo do ponto de vista do desenvolvedor já que este terá toda a arquitetura do sistema que foi prevista poderá ser modelada.

Além disso, algo que não é feito em SMA e será provido pelo método é uma documentação para guiar as instâncias das futuras aplicações, o que facilitará ainda mais do ponto de vista do desenvolvedor.

O paradigma de agentes traz muitas linguagens para modelagem de SMA e diferente do paradigma de orientação a objeto não existe uma linguagem mais

utilizada como é a UML em OO. Em função disto a solução proposta visa ser utilizável por todas as linguagens de modelagem.

1.3. Abordagem

Esta dissertação propõe um método para suporte ao desenvolvimento de SMA onde um agente ao menos tenha um ponto de flexibilização.

O primeiro passo para isto é que após a divisão dos objetivos do sistema delegando para cada agente os seus objetivos, feito isto estes agentes devem ter suas atuações explicitadas em função do objetivo a ser alcançado (Choren, 2002). A partir da descrição de atuação do sistema será possível descrever os planos de ação dos agentes e as ações que estes deverão executar (Choren, 2002).

Finalizado tais passos a solução proposta visa dar instrumentos ao responsável pela modelagem para que através de diagramas este possa representar as instâncias possíveis de um novo sistema, que gira entorno da possibilidade de ação dos agentes segundo seus planos e ações.

Além de representar através de diagramas o responsável pela modelagem poderá utilizar uma técnica de documentação baseada em RDL a fim de guiar a instanciação de novas aplicações segundo seus planos e/ou ações e com isto será possível definir uma interface de comunicação textual, completando o método proposto. A solução também se preocupa com a clareza na forma com que os modelos serão representados.

1.4. Contribuições

A dissertação apresenta uma abordagem para a modelagem de agentes de software, podendo ser também utilizada juntamente com as linguagens de modelagem.

Como contribuição acredita-se que a dissertação venha trazer uma forma de nova de modelar agente de software que possibilitará a representação de pontos de flexibilização de um agente num dado sistema que conseqüentemente serão pontos de flexibilização do próprio sistema. Os pontos de flexibilização de um agente de software poderão ser seus planos e as ações, os quais dependerão de

características do sistema para serem utilizados como tal. Assim os SMA que podem ser utilizados em vários escopos distintos, ou seja, podem ter instancias distintas, terão suas representações através de seus pontos de flexibilização, ou seja, através de seus planos e ações. Com o método proposto será possível representar todos os possíveis escopos para uma aplicação baseada em SMA.

A modelagem de tais pontos traz a possibilidade de ter um agente com um *framework* onde os planos e as ações são utilizados como *hotspots* o que resultaria na instância de uma aplicação. Desta forma, cada aplicação teria a sua implementação de planos e ações.

A dissertação contribui com os seguintes pontos:

- O método proposto possibilita que o sistema multi-agente tenha uma flexibilidade documentada o que facilitará as futuras modificações;
- Com o método proposto o sistema terá uma melhor adaptação ao processo incremental;
- A proposta dá ao sistema uma possibilidade de reutilização do design (facilidade de manutenção);
- A proposta permite que seja dada uma maior atenção na área específica;
- O método proposto pode ser facilmente utilizado de modo concomitante com outras linguagens de modelagem de agentes;
- A proposta traz uma técnica de documentação que permitirá que o desenvolvedor tenha um guia para novas instâncias;
- O uso da idéia de *framework* para agentes de software, trazendo características de re-uso.

1.5. Estrutura da Dissertação

O capítulo 2 apresenta conceitos básicos de um agente de software e definição de um sistema multi-agente além de uma pequena discussão do uso de agentes de software.

O capítulo 3 apresenta os trabalhos relacionados, segundo a visão da modelagem e da implementação e uma comparação entre técnicas como *Cookbook*, *Hook* e RDL.

O capítulo 4 apresenta a solução proposta, que resulta em um método para representar as variabilidades de um agente de software, além de apresentar dois estudos de casos, o sistema médico CQPM e um sistema de compra e venda.

O capítulo 5 apresenta uma breve introdução ao sistema *LearnAgents*, a modelagem do *LearnAgents* com o ANote e a modelagem de um dos agentes de tal sistema com o método proposto, com o objetivo de mostrar como a solução seria aplicada a um sistema mais complexo. O capítulo 6 apresenta a conclusão e trabalhos futuros. O capítulo 7 traz as referências utilizadas na dissertação.