Modelagem de Requisitos Orientada a Agentes utilizando MaSE

Raquel C. S. Alex L. de Araújo Carlos A. Gomes Junior
Rosa Maria E. M. da Costa Vera Maria B. Werneck

Instituto de Matemática e Estatística - Universidade do Estado do Rio de Janeiro
Rua São Francisco Xavier, 524 / sala 6020-B, Maracanã, CEP: 20.550 – 013, Rio de Janeiro, RJ Brasil
{rcosta, vera}@ime.uerj.br

Resumo

Apesar de vários métodos de modelagem orientados a agentes terem sido propostos adotando os conceitos de agentes para o desenvolvimento de sistemas, esta é ainda uma abordagem recente. O paradigma Orientado a Agentes surgiu como solução à demanda de novas aplicações que consideram características de autonomia, pro-atividade e integração. Neste trabalho é apresentada a metodologia MaSE, exemplificada na modelagem de requisitos de um estudo de caso prático e complexo, denominado Guardian Angel, analisando características consideradas vantajosas e destacando suas deficiências.

Abstract

The study of the Agent-Oriented design approach is still recent, even though many systems development methodologies that adopt this concept have been proposed so far. Agent-Oriented paradigm emerges to meet the demand for new application modeling which should consider characteristics of autonomy, pro-activity and integration This paper presents the MaSE methodology and study case requirements modeling, exposing their strengths and disadvantages.

1. Introdução

Novas aplicações, que atendam aos requisitos e características das organizações sociais e de seus relacionamentos, de forma autônoma e integrada, têm impulsionado a pesquisa por novos paradigmas de desenvolvimento de software. A Modelagem Orientada a Agentes vem suprir algumas destas necessidades, na construção de sistemas que tenham na autonomia, na mobilidade e na capacidade de coordenação e adaptação, os aspectos fundamentais de seu funcionamento.

Alguns métodos orientados a agentes têm sido propostos na literatura, tais como, Gaia [1], [2], MESSAGE [3], [4], MAS-CommonKADS [5], Adelfe [6], Tropos [7], MaSE [8], Prometheus [9]. Algumas extensões de métodos ou linguagens de modelagens consagradas como CommonKADS [10] e UML [11] também têm sido estendidas para atender aos sistemas multi-agentes como é o caso do MAS-CommonKADS [5] e AUML[12].

Este trabalho está inserido num projeto de pesquisa de metodologias Orientadas a Agentes [13], [14], [15], [16], [17] com base no *framework* de avaliação proposto por Yu e Cysneiros [18]. Este artigo tem como objetivo apresentar uma avaliação do método MaSE (Multi-agent Systems Engineering).

Os conceitos da metodologia MaSE [8], [20], [21], [22], [23] são apresentados, ilustrados através da modelagem de um estudo de caso prático, baseado num sistema denominado "Guardian Angel" [24] para o acompanhamento integral de pacientes com doenças crônicas, tais como diabéticos e hipertensos, que deve ser utilizado, inclusive através de dispositivos móveis (PDAs).

Neste artigo apresentamos na seção 2, o *framework* de avaliação baseado num exemplar. Na Seção 3 são descritos os conceitos básicos, modelos e diagramas do método MaSE. Na seção 4 a modelagem dos requisitos do estudo de caso é descrita através de exemplos dos modelos e diagramas do método. Na seção 5 é elaborada uma avaliação geral abordando as vantagens e desvantagens dessa experiência obtida no uso do método MaSE. Na seção 6, descrevem-se as conclusões deste trabalho.

2. *Framework* de avaliação baseado em um exemplar

O framework de avaliação [18] utilizado neste trabalho é baseado no Sistema "Guardian Angel" [24] que provê o suporte automático para assessorar pacientes com doenças crônicas tais como diabetes e hipertensão, integrando aspectos relacionados ao sistema de saúde, incluindo informações legadas de remédios e informações financeiras.

Esse sistema auxilia na captura, gerenciamento e interpretação do histórico pessoal da saúde do paciente e sua rotina diária na condução do tratamento, aconselhando-o com as melhores opções, com base em suas preferências pessoais. Mantém, também, os registros médicos de forma compreensiva, acumulativa, correta, e coerente, acessível em qualquer tempo, em diferentes locais e para diferentes sistemas de saúde [14], [18].

O exemplar [18] é expresso em termos de um conjunto de cenários numerados (EA0.0 até EA9.0) como por exemplo: "EA2.0 - Uma vez que o paciente e o médico estabeleceram um plano de tratamento, o GA-PDA deverá auxiliar para que o paciente siga e monitore o andamento desta rotina".

O processo propõe, também, acoplado a esses cenários, um conjunto de questões de avaliação para apoiar a avaliação de como a metodologia suporta a modelagem desses cenários.

Neste trabalho, porém, utilizamos principalmente as perguntas metodológicas apresentadas no exemplar relacionadas às fases de requisitos. O uso das perguntas metodológicas apresentadas em Yu e Cysneiros [18] justifica-se por já termos usado-as em diversos estudos anteriores [13], [14], [15]. [16], [17] durante os quais fomos capazes de avaliar o elevado grau de utilidade destas perguntas. Este aspecto será mais detalhado na seção 5.

3. A metodologia MaSE

A MaSE tem sido utilizado para projetar diversos tipos de sistemas, desde sistemas de integração de dados heterogêneos até sistemas biológicos, sistemas de computador vírus-imune e sistemas robóticos cooperativos. Os sistemas multi-agentes projetados pelo MaSE são tipicamente fechados, isto é, o número e tipo de todos os agentes são conhecidos a princípio.

A metodologia MaSE faz uso da abstração proveniente dos sistemas multi-agentes para colaborar com projetistas no desenvolvimento de sistemas inteligentes e distribuídos. Os agentes são considerados uma abstração além do paradigma orientado a objetos, sendo uma especialização dos objetos. No lugar de simples objetos, com métodos que podem ser chamados por outros objetos, os agentes se comunicam via mensagens e agem pro-ativamente para concluir metas individuais e globais. Os agentes são abstrações convenientes que permitem aos projetistas modelarem componentes de sistemas inteligentes e não inteligentes igualmente, dentro do mesmo ambiente.

Os modelos propostos na MaSE são similares aos modelos definidos na UML [11] mas freqüentemente, têm uma semântica especializada para definições de multi-agentes.

Este método abrange as fases de análise, projeto, e implementação de sistemas multi-agentes por proceder, de forma ordenada, todo o ciclo de vida do desenvolvimento [22], [23]. A ferramenta AgentTool oferece suporte ao desenvolvedor nos vários modelos, desde a definição de metas de alto nível até a verificação automática, geração de projetos semi-automáticos e geração de código.

A duas fases principais da metodologia MaSE são a fase de análise e a de projeto, que resultam na criação de uma série de modelos complementares. As atividades e os respectivos modelos de cada fase são apresentados no diagrama da figura 1. Esta metodologia é na prática iterativa, embora seja apresentada seqüencialmente. A intenção é deixar o analista livre para se movimentar entre os passos e fases de tal forma, que a cada passagem sucessiva, um detalhe é adicionado, gerando um sistema completo e consistente.

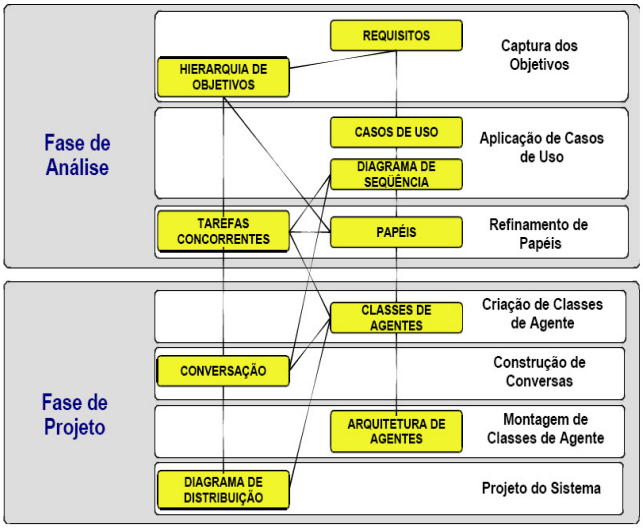

Figura 1. Fases de modelagem da MaSE

3.1. Fase de Análise

Na fase de análise é definida uma série de papéis que podem ser usados para alcançar metas do sistema. Estes são definidos explicitamente através de tarefas que são descritas pelos modelos de estado finito. Este processo é realizado em três passos: capturar as metas, aplicar os casos de uso e definir os papéis.

Na captura das metas do sistema são extraídos os requisitos iniciais encontrados. Os mesmos podem existir de várias formas, incluindo texto informal, e mostram como o sistema deve funcionar baseado em entradas específicas e nos estados do sistema.

Os requisitos são utilizados para definir metas em dois subpassos específicos: identificar e estruturar as metas.

O objetivo da identificação das metas é derivar a meta global do sistema e suas submetas em uma lista inicial de requisitos. Primeiramente os cenários são extraídos dos requisitos e então, as metas destes cenários são identificadas. Estes cenários iniciais são normalmente resumidos e são críticos para todo o sistema. Portanto, as metas identificadas nestes cenários estão em um alto nível de abstração. Estas metas servem como base para todo o sistema. Os papéis definidos posteriormente devem suportar cada uma destas metas. Se um papel não suporta uma destas metas, ou o papel não é necessário, ou o conjunto inicial de metas foi incompleto e uma nova meta precisa ser adicionada.

Depois das metas terem sido identificadas, o segundo passo é categorizá-las e estruturá-las em uma árvore de metas. O resultado é o diagrama de hierarquia de metas, cujos nós representam metas e os arcos definem o relacionamento entre metas e submetas. No caso onde há mais de uma meta principal, estas devem ser

sumarizadas como uma meta de alto nível, que é decomposta em uma série de submetas que são fáceis de gerenciar e entender. Para decompor uma meta em submetas, deve-se analisar o que deve ser feito para alcançar a meta mãe. A decomposição de metas deve parar quando qualquer decomposição futura resultar em funções e não em submetas. A decomposição de metas da metodologia MaSE é similar a da metodologia KAOS [25] exceto que as metas da MaSE não precisam ser decompostas estritamente em ligações AND ou OR. O diagrama de hierarquia de metas é acíclico. Entretanto, há algumas submetas que podem ter mais de uma meta mãe.

Há quatro tipos de metas no diagrama de hierarquia de metas: de resumo, particionadas, combinadas e não funcionais. Qualquer meta ou submeta pode assumir os atributos de qualquer outra meta ou destes tipos de metas.

Uma vez identificadas e estruturadas, as metas são traduzidas em casos de uso. Estes capturam o cenário definido no passo anterior promovendo uma descrição textual e uma lista de diagramas de seqüência, que são similares aos diagramas de seqüência UML [11], [26]. A principal diferença é que em MaSE eles são usados para representar seqüências de eventos entre papéis ao invés de objetos. Os eventos enviados entre papéis são usados nos próximos passos, para ajudar a definir as comunicações entre os agentes que interpretam esses papéis. Os casos de uso podem ser convertidos em diagramas de seqüência, quando os papéis identificados tornam-se a lista inicial de papéis.

Os requisitos críticos são mapeados em casos de uso positivos ou negativos. Casos de uso positivos definem o comportamento desejado do sistema, enquanto que casos de uso negativos descrevem um colapso ou um erro no sistema. Os dois são úteis para definir os papéis interpretados no sistema.

Com base no diagrama de hierarquia de metas e nos casos de uso é realizada a definição de papéis, associando-as com tarefas específicas. Os papéis são definidos de tal forma a assegurar que cada meta do sistema é contabilizada a fim de formar o alicerce para os agentes do sistema.

A metodologia MaSE assume a hipótese de que o sistema de metas será satisfeito se cada meta leva a um papel, e todo papel é desempenhado por no mínimo uma classe de agente. Em geral, o mapeamento (transição) das metas para papéis envolve uma correspondência de um para um. Contudo, é permitido que um papel seja responsável por várias metas, por motivos de conveniência ou eficiência. É possível combinar vários papéis, embora isto possa aumentar a complexidade individual dos mesmos, podendo simplificar significativamente o projeto.

A definição de metas é capturada em um modelo de papéis [23]. Enquanto os papéis são decompostos em um conjunto de tarefas, as tarefas individuais são desenvolvidas para atingirem as metas pelas quais o papel é responsável. Os papéis não devem dividir tarefas com outros papéis, pois indica uma decomposição imprópria. Se uma tarefa necessita ser compartilhada, então um papel separado deve ser criado para aquela tarefa. Isto permitirá que a tarefa seja incorporada em diferentes classes de agentes, sendo efetivamente dividida.

Depois que os papéis são definidos, cada tarefa deve ser analisada no modelo de papéis e definida no Diagrama de Tarefas Concorrentes, que é baseado no diagrama de transição de estado. Semanticamente, assume-se que cada tarefa possa ser executada concomitantemente e se comunicar com outras tarefas, tanto internas quanto externas. Em geral, o conjunto de tarefas para um específico papel deve definir o comportamento deste.

Uma tarefa concorrente consiste em um conjunto de estados e transições. Os estados nas tarefas concorrentes representam o funcionamento interno de um agente, enquanto as transições definem as comunicações entre as tarefas. Toda transição no modelo possui uma fonte de estado, destinação do estado, um "gatilho" (disparo), condição de guarda e transições. As transições usam a sintaxe: "trigger[guard]^transmission(s)".

Se forem necessárias múltiplas transmissões, elas podem ser concatenadas usando o ponto e vírgula como separador, contudo nenhuma ordenação é implícita. Em geral, eventos enviados como *triggers* ou transmissões são associados com eventos enviados para tarefas com a mesma instanciação de papéis, permitindo desta maneira a coordenação interna das tarefas. Para representar mensagens enviadas entre agentes, contudo, dois eventos especiais: *send*, denotado por "send (mensagem, agente)" e *receive* denotado por "receive (mensagem, agente)". É possível mandar uma mensagem para vários agentes ao mesmo tempo usando *multicasting*.

Estados de tarefas podem conter atividades que representam o raciocínio interno, através de sensores, ou de ações. Várias atividades podem ser incluídas em um único estado. A tarefa permanece naquele estado até que a seqüência de atividades seja completada. As variáveis usadas nas atividades e nas definições de eventos são visíveis dentro da tarefa, mas não fora da tarefa. Todas as mensagens enviadas são enfileiradas para garantir que todas as mensagens sejam recebidas mesmo se o agente ou tarefa não estiver no estado apropriado para lidar imediatamente com a mensagem ou evento.

Uma vez que uma tarefa é ativada, a mesma é executada instantaneamente. Se múltiplas transações são ativadas, então lidamos primeiro com os eventos internos, depois com mensagens externas (eventos de envio e recebimento) e por último, as transições com condições de guarda [23].

O Diagrama de Tarefas Concorrentes possibilita embutir um controlador de tempo**.** Um agente pode definir um controlador de tempo com a notação "t=setTimer(time)". O controlador de tempo da atividade emprega um tempo como dado de entrada e retorna um controlador de tempo que irá se esgotar exatamente no tempo especificado. O controlador de tempo pode então, ser testado através do tempo de término da atividade, "timeout(t)".

3.2. Fase de Projeto

Na fase de *design* o objetivo é converter papéis e tarefas em uma forma que permita a implementação do modelo. Consiste em modelar o diagrama de classes e o diálogo entre os agentes, para que seja possível construir e finalmente, implantar os agentes.

Para modelar as classes de agente individuais, deve ser mapeado cada papel definido na fase de análise para pelo menos uma classe de agente. Considerando as metas, deve ser realizada a validação de cada meta em no mínimo uma classe de agente para que o sistema ajude a garantir que as metas sejam de fato implementadas no sistema. Se mais de um papel for atribuído para a mesma classe de agente, deve ser considerada a execução tanto concomitantemente, quanto seqüencialmente. A atribuição dos papéis dos agentes permite que a organização do sistema multi-agente seja facilmente modificada, seguindo os vários princípios da engenharia de software, tais como coesão funcional ou temporal.

Uma vez que os agentes são criados através da identificação dos papéis que eles irão desempenhar, os diálogos entre os agentes são desenvolvidos de acordo com os mesmos critérios. O diagrama de classes de agentes, que resulta deste passo, é similar ao diagrama de classes orientado a objetos**.** No entanto, as classes de agentes são definidas pelos papéis que eles desempenham no lugar de seus atributos e métodos, sendo que as relações entre as classes de agentes são sempre diálogos.

4. Estudo de Caso Guardian Angel

A modelagem com a metodologia MaSE utilizou a ferramenta AgentTool e os cenários padronizados apresentado por Cysneiros e Yu [7] para o sistema Guardian Angel.

Como pacientes crônicos precisam de acompanhamento dinâmico, o sistema dá apoio a esse monitoramento, como por exemplo, permitir o monitoramento da glicose, com a ajuda de um medidor portátil, armazenando as diversas medições. Deve, também, dar orientações ao paciente, se baseando em perguntas e interpretando suas respostas, cruzando-as com o perfil do paciente e os parâmetros fornecidos pelo médico. Nos casos extremos, pode notificar de forma eletrônica os médicos responsáveis pelo tratamento, marcar uma consulta ou mudar o tratamento atual do paciente (medicamentos, alimentação, exercícios físicos, etc).

Neste cenário padronizado, existem pelo menos três instâncias do Guardian Angel: a implementação pessoal (GA_PDA) em um PDA para cada paciente, a implementação do software em um computador fixo do paciente (GA_Home) e o sistema utilizado pelo médico (GA_Hospital). Neste estudo de caso, a análise foi realizada assumindo que o módulo GA_Home e o módulo GA_PDA teriam a mesma funcionalidade, abstraindo o dispositivo a ser utilizado.

Na subseção a seguir são apresentados os diagramas e artefatos resultantes da fase de Análise. A modelagem da fase de projeto não foi totalmente modelada nesse experimento.

4.1 A fase de Análise

A partir do levantamento de requisitos funcionais e não funcionais baseados nos cenários descritos em Cysneiros e Yu [8] para o sistema Guardian Angel foi definido o diagrama de metas apresentado na figura 2. A meta principal “Promover o bem estar do paciente” que tem como propósito disponibilizar diferentes visões. do sistema foi dividida nas submetas “Gerenciamento Clínico”, “Gerenciamento Administrativo” e “Prover Interfaces”. É importante mencionar que o diagrama de

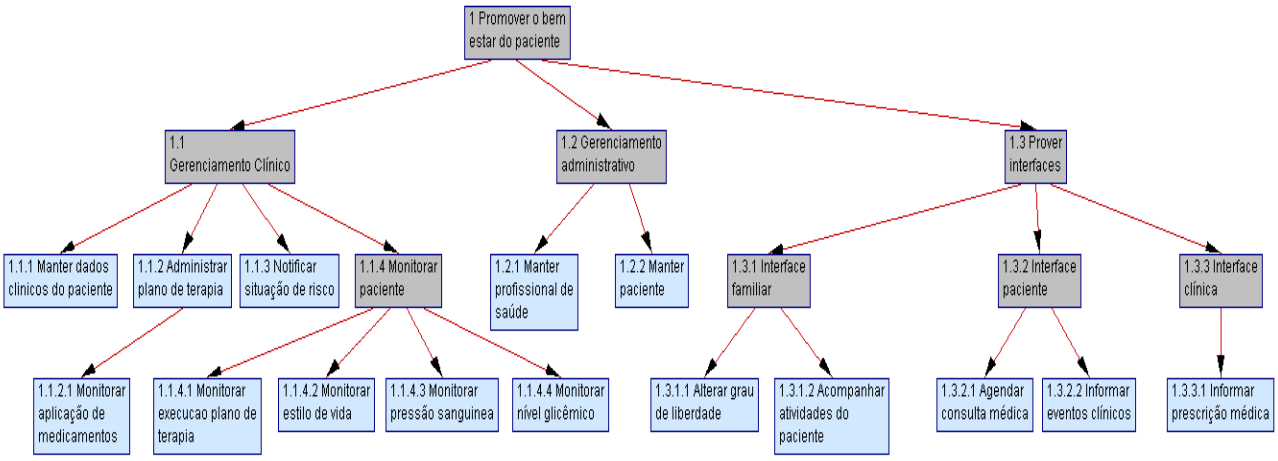

Figura 2. Diagrama de Metas do Sistema Guardian Angel

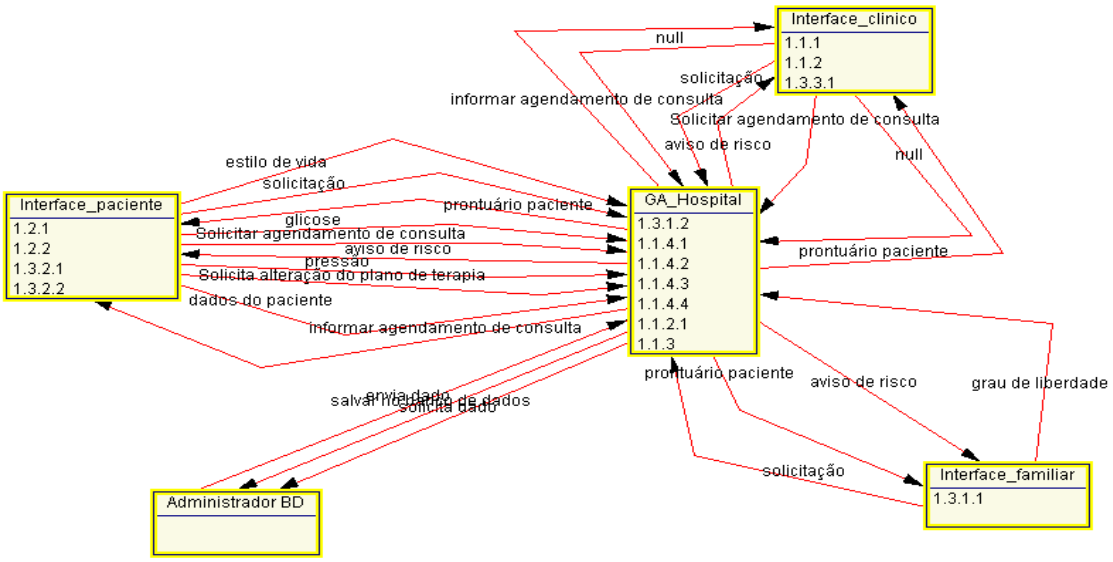

Figura 3. Diagrama de Papéis do Sistema Guardian Angel

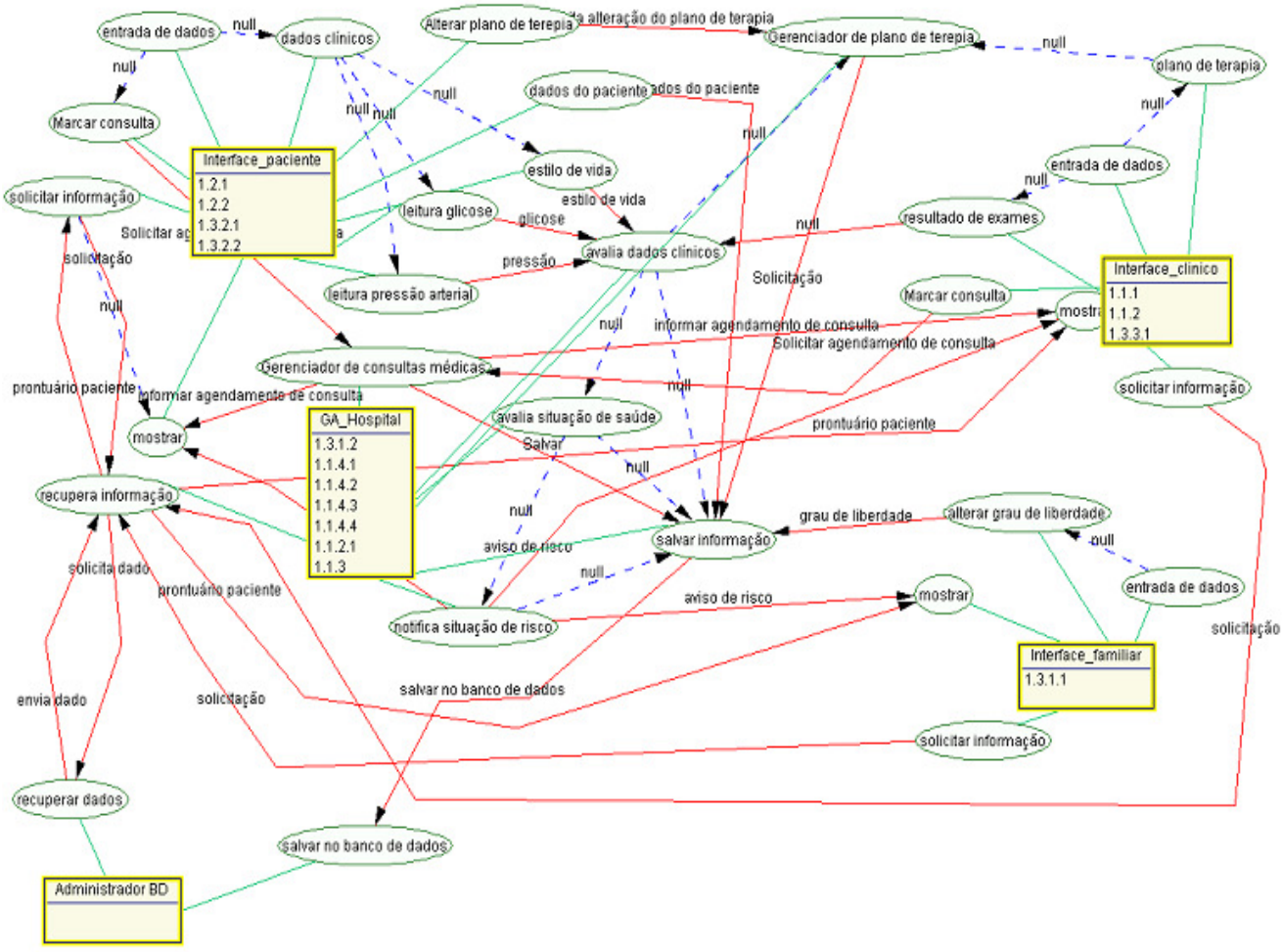

Figura 4. Diagrama de Papéis Completo do Sistema Guardian Angel

metas evoluiu em paralelo ao levantamento de requisitos

Após a finalização do levantamento de requisitos, foi realizada a descrição dos casos de uso. Estes foram descritos de forma textual sendo padronizados nas seguintes seções: Atores, Agentes, Pré-condições e Fluxo de eventos.

Eventualmente, para alguns casos de uso foram descritos cenários alternativos que levaram a fluxos de eventos alternativos.

A partir dos casos de uso, foram construídos os diagramas de sequência a fim de melhor mapear os papéis existentes no sistema. Definiram-se então cinco papéis: as interfaces (Interface_Paciente, Interface_Profissional e Interface_Familiar), GA_Hospital e Administrador_BD conforme pode ser visto na figura 3. Pode ser observado nessa figura que cada uma das metas, do diagrama hierárquico foi mapeada em, pelo menos, um papel. Os números das metas aparecem abaixo do nome dos papéis.A figura 4 apresenta o diagrama de papéis completo onde os papéis são representados por retângulos, enquanto as tarefas de um papel são representadas por elipses anexadas. As flechas entre as tarefas designam protocolos de comunicação, apontando do iniciador do protocolo para o respondedor.

Linhas cheias representam comunicações externas (papel para papel), enquanto linhas tracejadas indicam comunicações internas entre tarefas pertencentes ao mesmo papel.

Cada tarefa associada aos papéis foi detalhada através de um diagrama de tarefas concorrentes. A tarefa "Notifica situação de risco" foi detalhada conforme mostra a figura 5.

5. Avaliação da metodologia MaSE

Nesta fase utilizamos o *framework* de avaliação proposto em Yu e Cysneiros [18] que prevê um conjunto de questões de avaliação (primeira coluna da Tabela 1) para apoiar a avaliação de como a metodologia suporta a modelagem desses cenários.

As questões metodológicas complementam esses cenários, identificando situações em que a metodologia deve orientar na elicitação e análise dos requisitos e também, os casos específicos de modelagem que a metodologia deve atender. Trabalhos anteriores de avaliação das metodologias Tropos, Gaia, MESSAGE, Mas-CommonKADS, Adelfe e UML/RUP [13], [14], [15], [16], [17] mostraram o uso eficiente desse *framework* de avaliação, ressaltando os pontos fortes, as deficiências e as potencialidades de cada metodologia.

A Tabela 1 sintetiza a avaliação geral da metodologia MaSE, com os seus principais aspectos identificados, onde foram atribuídos (S) para os pontos fortes, (W) para as deficiências e (N) para os pontos neutros.

A metodologia MaSE foi considerada forte nas questões A1 a A3 relacionadas a pro-atividade, autonomia e raciocínio de autonomia pois a pro-atividade é possível de ser modelada através do Diagrama de Seqüência, Diagrama de Papéis e Diagramas de Estado e da existência do conceito de BDI (Belief-Desire-Intention), o que permite identificar estruturas que aprendem e tomem ação mediante certas situações. A Autonomia é representada na fase de análise, pelos Casos de uso e seus respectivos Diagramas de Seqüência, pelo Diagrama de Papéis, Diagrama de Tarefas Concorrentes suportadas por cada papel e, pela troca de mensagens entre as tarefas, definidas na fase de Projeto. Desta forma, é possível retratar as condições relatadas e o comportamento das tarefas em variadas situações. E durante a execução, através da funcionalidade de BDI dos agentes, é possível estabelecer a autonomia humana e do software. No aspecto de diferentes níveis de abstração (A4) também foi considerado forte o relacionamento entre os agentes, que é representado e descrito através do Caso de Uso e dos Diagramas de Seqüência. A fase de descrição e análise termina no momento em que temos o diagrama com os papéis, tarefas e Diagrama de Tarefas Concorrentes concluídos.

Os participantes no domínio são modelados nos Casos de uso, Diagramas de Seqüência e Diagrama de Papéis (A5).

Os diversos níveis de abstração são obtidos através das diversas fases da metodologia, partindo do entendimento das metas do sistema até a implementação do sistema. Ao todo, são nove etapas e se pode avançar ou retroceder ao longo das etapas, conforme a necessidade.

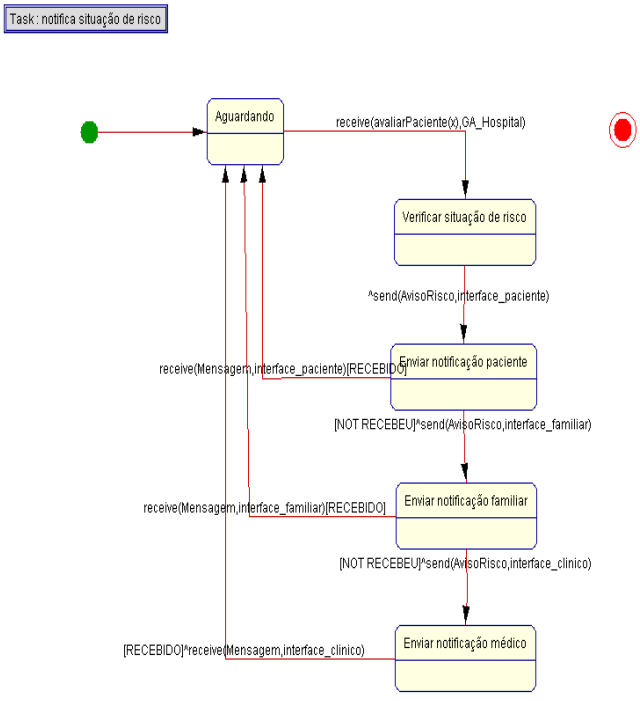

Figura 5: Diagrama de Tarefas Concorrentes

Tabela 1. Questões Metodológicas e Avaliação *MaSE*

QA1 -Pro-atividade	S	**QA18 -Projeto e raciocínio da arquitetura**	W
QA2 -Autonomia Humana X autonomia de software	S	**QA19 -Elicitando e Raciocinando requisitos não-funcionais**	N
QA3 -Raciocínio da autonomia	S	**QA20 -Projeto e raciocínio da arquitetura e evolução**	W
QA4 -Diferentes níveis de Abstração	S	**QA21 -Projeto e raciocínio da arquitetura**	S
QA5 -Identificando participantes do domínio	S	**QA22 -Validando especificação ao longo do ciclo de vida**	S
QA6 -Capturando,entendendo e registrando terminologia	W	**QA23 -Rastreando mudanças dos requisitos ao projeto**	S
QA7 -Análise de domínio	S	**QA24 -Rastreando mudanças do projeto ao código**	S
QA8 -Levantamento de requisitos	S	**QA25 -Concorrência**	S
QA9 -Cooperação homem-máquina	S	**QA26 -Rastreando de volta aos requisitos**	S
QA10 -Projeto de banco de dados	N	**QA27 -Modularidade de software**	S
QA11 -Evolução de banco de dados	W	**QA28 -Verificação e Validação Formal**	W
QA12 -Projeto de banco de dados legados	S	**QA29 -Gerenciamento de Projeto**	S
QA13 -Raciocinando diferentes requisitos não-funcionais	N	**QA30 -Trabalhando equipes distribuídas**	W
QA14 -Mobilidade	W	**QA31 -Suporte de ferramentas**	S
QA15 -Projeto de interface com usuário	S	**QA32 -Curva de aprendizagem**	N
QA16 -Gerando casos de teste	S	**QA33 -Integração com outras metodologias**	W
QA17 -Projeto de interface com usuário	S		

A Captura, entendimento e registro da terminologia (A6) é uma opção a ser suportada futuramente pela metodologia com uso de ontologia por isso esse critério foi avaliado como deficiente. Entretanto, a Análise de domínio, Levantamento dos Requisitos e Cooperação máquina-homem são aspectos fortes em MaSE que se baseiam na definição dos Diagrama de Casos de Uso, Diagrama de Papéis e Diagramas de Tarefas Concorrentes, além dos Diagramas de Sequência que apresentam a modelagem da Cooperação máquina-homem.

Em relação aos Requisitos Não-Funcionais (A13, A19), MaSE em algumas situações suporta uma representação para qualquer requisito não funcional. Por exemplo, no Diagrama de Papéis pode ser modelado o sincronismo de dados e nos Diagramas de Tarefas Concorrentes, o tratamento da falha de comunicações.

A Concorrência (A25) pode ser modelada no Diagrama de Papéis, nos Diagramas de Tarefas Concorrentes e na fase de projeto nas Conversações.

A validação da especificação (A22) é facilitada pelo uso da ferramenta agentTool que também permite o rastreamento das alterações ao longo do processo de desenvolvimento (A23 e A24). Ao se identificar um novo requisito este é representado na fase de análise e todas as demais etapas são refeitas, para que a alteração se reflita no modelo.

Neste caso, julgamos que todas as questões metodológicas apresentadas em Yu e Cysneiros (2002) foram respondidas, entretanto, algumas foram baseadas nas experiências anteriores dos participantes com esta metodologia já que a modelagem do projeto não foi realizada.

6. Conclusões

A metodologia MaSE proporciona uma abordagem detalhada na análise e projeto de sistemas multi-agentes, através da combinação de um conjunto de modelos. Segundo nossa experiência e estudo anteriores, também relatados em [27], a vantagem de sua utilização é a orientação fornecida durante todo o processo de desenvolvimento do software, desde os requisitos até a implementação e a independência de qualquer arquitetura ou ambiente.

A seqüência de modelos inter-relacionados permite que o analista acompanhe o mapeamento de entidades de um modelo para outro. Por exemplo, os protocolos de comunicação definidos no Diagrama de Tarefas Concorrentes devem ser detalhados nos Diagramas de Conversação.

MaSE é uma metodologia baseada numa estrutura de desenvolvimento *top-down* e tem no Modelo de Papéis o diagrama central para definição dos agentes, seus desempenhos e comunicações. Isso vem facilitando o desenvolvimento de diversos projetos de graduação, assim como muitos projetos de pesquisa, que têm explorado essa metodologia com êxito. Wood e DeLoach [20] citam como exemplo dois projetos: (i) *Multi-Agent Distributed Goal Satisfaction* que projeta um framework de agentes colaborativos na integração de satisfação de diferentes restrições e (ii) *Agend-Based Mixed-Initiative Collaboration que usa multi-agentes* para distribuição humana e planejamento de equipamentos, projetos de sistemas de banco de dados heterogêneos baseados em agentes, bem com como, uma abordagem computacional destinada aos sistemas biológicos para defesa de vírus. Mais recentemente, foi aplicada a um grupo de robôs autônomos, a busca heterogênea e recuperação [27].

Sistemas robóticos cooperativos baseados em uma abordagem teórica organizacional e a especialização do uso em multi-agentes adaptativos serão os focos no futuro. Um modelo organizacional que proporciona o conhecimento necessário para um grupo de agentes de software ou hardware se adaptarem às mudanças no ambiente e organizar ou reorganizar o comportamento

do grupo de metas está sendo desenvolvido [27]. Grande parte da informação necessária neste modelo organizacional pode ser capturada nesta metodologia, no entanto, a etapa de análise deverá ser estendida a fim de obter maiores detalhes dos papéis, incluindo as aptidões requeridas para executar os mesmos.

No futuro pretendemos organizar esse conhecimento dessas experiências em modelagem de metodologias orientadas a agentes do Guardian Angel, através da construção de uma ontologia para melhorias ou otimização do uso desses métodos.

7. Referências

[1] Wooldridge, M., Jennings, N.R. and Kinny, D., The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems, 3(3), 2000, pp 285-312.

[2] Zambonelli, F., Jennings, N. R. and Wooldridge, M., Developing Multiagent Systems: The Gaia Methodology, In ACM Transaction on Software Engineering and Methodology, Vol. 12, No. 3, July 2003, pp 317-370.

[3] Caire, G., Coulier, W., Garijo, F., Gómez-Sanz, J., Pavón, J., Kearney, P.and Massonet. P., Message: A Methodology for Development of Agent-Based Applications, In: Methodologies And Software Engineering For Agent Systems, edited by Federico Bergenti, Marie-Pierre Gleizes and Franco Zambonelli, to be published by Kluwer Academic Publishing (New York), 2004.

[4] Message, http://www.eurescom.de/~public-webspace/P900-series/P907/index.htm, May 23, 2000.

[5] Iglesias, C.A. e Garijo, M. (2005) "The Agent Oriented Methodology MAS-CommonKADS; IN Agented-Oriented Methodologies", Brian Henderson-Sellers e Paolo Giorgini (ed.), IDEA Group Publishing, p. 46-78.

[6] ADELFE (Atelier de Développement de Logiciels à Fonctionnalité Emergente) at http://www.irit.fr/ADELFE.

[7] Tropos at BRESCIANI, Paolo et al. Tropos: An Agent-Oriented Software Development Methodology, Kluwer Academic Publishers, 2004.

[8] MaSE, at http://macr.cis.ksu.edu/projects/mase.htm

[9] Prometheus at http://www.cs.rmit.edu.au/agents/ SAC2/methodology.html

[10] Schreiber, G et al, "Knowledge Engineering and Management: The CommonKADS Methodology", Cambridge, MA,1999.

[11] Rumbaugh, J., Jacobson, I. E Booch,G., The Unified Modeling Language Reference Manual, Second edition, Addison,-Wesley, 2004.

[12] AUML at http://www.auml.org/auml/

[13] Cysneiros, L. M., Werneck, V. M. B.; Yu, Eric, "Evaluating Methodologies: A Requirements Engineering Approach Through the Use of an Exemplar". Journal of Computer Science & Technology, USA, v. 5, n. 2, 2005, pp. 71-79.

[14] Cysneiros, L. M., Werneck, V. M. B., Amaral, J. and Yu, E., "Agent/Goal Orientation versus Object Orientation for Requirements Engineering: A Practical Evaluation Using an Exemplar", In: Proc. of VIII Workshop in Requirements Engineering, Porto, 2005, pp.123-134, ISBN 972-752-079-0.

[15] Coppieters, A. M., Marzulo, L.A.J., Kinder, E. e Werneck, V.M., "Modelagem Orientada a Agentes utilizando MESSAGE", Cadernos do IME-Série Informática, Rio de Janeiro, v.18, 2005, pp. 38-46.

[16] Werneck, V. M.; Pereira, L. F.; Silva, T. S.; Almentero, E. K.; Cysneiros, L. M.; . "Uma Avaliação da Metodologia MAS-CommonKADS", In: Proceedings of the Second Workshop on Software Engineering for Agent-oriented Systems, (SEAS´06), Florianópolis, 2006, pp. 13-24.

[17] Werneck, Vera M. B. ; Cysneiros, Luiz Marcio ; KANO, A. Y.. Evaluating ADELFE Methodology in the Requirements Identification. In: Proceedings Of 10th Workshop On Requirements Engineering. Toronto: York University Printing Services, 2007. V. 1. P. 13-24.

[18] Yu, E., Cysneiros. L.M., "Agent-Oriented Methodologies Towards a Challenge Exemplar" in Proc of the 4th Intl. Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS 2002) Toronto May 2002.

[19] DeLoach, S. A. (2000). Specifying Agent Behavior as Cocurrent Tasks: Defining the Behavior of Social Agents. Technical Report, U. S. Air Force Institute of Technology, AFIT/EN-TR-00-03.

[20] Wood, M. F e DeLoach, S. A, (2001). An Overview of the Multiagent Systems Engineering Methodology. In Lecture Notes in Computer Science. Vol. 1957, Limerick, Ireland. P. Ciancarini, M. Wooldridge, (Eds.), Springer Verlag, Berlin, January 2001. 207-221

[22] DeLoach, S. A., Matson, E. T. & Li, Y. (2003). Exploiting Agent Oriented Software Engineering in Cooperative Robotics Search and Rescue. The International Journal of Pattern Recognition and Artificial Intelligence, 17(5), 817-835.

[23] DeLoach, S. A and Kumar, M., "Multi-agent Systems Engïneering: aN Overview and Case Study, IN: Agent-Oriented Methodologies, Brian Henderson-Sellers e Paolo Giorgini (ed.), IDEA Group Publishing, 2005, pp 317-340

[24] Szolovits, P., Doyle, J., Long, W.J., Kohane, I. e Pauker, S. G. (1994) "Guardian Angel: Patient-Centered Health Information Systems", Technical Report MIT/LCS/TR-604. Disponível em http://groups.csail.mit.edu/medg/projects/ga/manifesto/GAtr.html

[25] Van Lamsweerde, A. & Letier, E. (2000). Handling Obstacles in Goal-oriented Requirements Engeneering. IEEE Transactions on Software Engeneering, 26(10), 978-1005.

[26] Bernhard Bauer - UML Class Diagrams Revisited in the Context of Agent-Based Systems AFIT/GCS/ENG/00M-22. School of Engeneering, Air Force Institute of Technology, Wright-Patterson AFB.

[27] DiLeo, J., Jacobs, T. e Deloach, S. A. (2002). Integrating Ontologies into Multiagent Systems Engeneering. In Proc of the 4th Intl. Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS 2002) at AAMAS'02, Bologna, Italy, July 16. Avaiable from http://CEUR-WR.org/Vol-59.