Uma Abordagem Baseada em Agentes de Apoio ao Ensino a Distância Utilizando Técnicas de Engenharia de Software

Enyo J. T. Gonçalves[1], Marcos A. de Oliveira[1], João H. Freires Junior[1], Guilherme E. S. Feitosa[1], Douglas H. Mendes[1], Mariela I. Cortés[2], Robson G. F. Feitosa[3], Yrleyjander S. Lopes[2]

[1]Universidade Federal do Ceará – Campus Quixadá

[2]Universidade Estadual do Ceará

[3]Instituto Federal de Educação Ciência e Tecnologia do Ceará, Campus Crato

{enyo,marcos.oliveira}@ufc.br, junior0160@hotmail.com, guilhermeesteves@alu.ufc.br,douglasjva@hotmail.com, mariela@larces.uece.br, robsonfeitosa@ifce.edu.br, yrleyjander@gmail.com

Abstract. *Distance Education has been put in the spotlight in the last years supported by computational tools that help its growth. Virtual Learning Environments (VLE) are the main tools available, and the follow up of students through them can be automated in a way that the environments can be more responsive to the action of the parts involved in the learning process. This work presents the development of a Multiagent System that interacts with the MOODLE using agent based software engineering in development.*

Resumo. *A Educação à Distância vem crescendo nos últimos anos com o apoio de ferramentas computacionais. Os ambientes virtuais de aprendizagem (AVAs) merecem destaque neste cenário, por permitirem que os cursos nesta modalidade se desenvolvam de modo mais interativo e dinâmico. No entanto, o acompanhamento dos alunos e demais envolvidos através dos AVAs ainda é realizado por agentes humanos, tornando esta atividade propensa a falhas. Neste contexto, há a necessidade de um software autônomo para apoiar tal tarefa. Este artigo apresenta o desenvolvimento de um Sistema Multiagente para MOODLE utilizando técnicas de Engenharia de Software baseada em Agentes para guiar o desenvolvimento.*

1. Introdução

Sistemas Multiagentes (SMAs) são compostos por entidades de software autônomas (Agentes) [Russell e Norvig 2004] e conceitos relacionados como: organizações, papéis de agentes, ambiente, normas, compromissos, contratos, confiança e reputação. Pesquisas com SMAs vêm avançando no contexto das mais diversas necessidades de desenvolvimento e a engenharia de software orientada a agentes emerge de modo a preencher a lacuna por soluções que viabilizem o desenvolvimento destes sistemas da melhor forma possível.

Por outro lado, Ambientes Virtuais de Aprendizagem (AVAs) são ferramentas muito utilizadas na modalidade de Educação à Distância (EAD) [Mercado 2007]. Um AVA constitui-se de um ambiente complexo e que pode ser utilizado para disponibilizar material didático e também ferramentas de comunicação síncronas e assíncronas para a formação de alunos sob a supervisão de professores e tutores.

Por conta do ambiente complexo que se constituem os AVAs, vários problemas emergem deste cenário, alguns problemas são listados por [Mercado 2007], como dificuldades nas interações e trabalhos em grupo e orfandade online. O acompanhamento destes problemas é uma atividade propensa a falhas quando realizada por humanos, desta forma, o uso de ferramentas computacionais autônomas como SMAs podem colaborar significativamente neste cenário.

Este artigo apresenta o desenvolvimento de um SMA para acompanhamento do ensino através da ferramenta MOODLE (*Modular Object-Oriented Dynamic Learning Environment*) [Moodle 2013], utilizando técnicas de engenharia de software baseada em agentes. O trabalho encontra-se dividido assim: na seção 2 o referencial teórico é apresentado; na seção 3 é mostrado o processo de desenvolvimento de um SMA para EAD; na seção 4 é mostrado o relato experimental para a fase inicial do processo; os trabalhos relacionados são explorados na seção 5; e, finalmente, na seção 6 são apresentadas as conclusões e trabalhos futuros.

2. Referencial Teórico

Nesta seção inicialmente são apresentados conceitos de SMAs através de ferramentas relacionadas ao desenvolvimento do trabalho, e em seguida alguns conceitos de educação a distância.

2.1. Sistemas Multiagentes e Engenharia de Software Baseada em Agentes

Os sistemas centrados em agentes [Padilha e Jacome 2002] são amplamente explorados pela comunidade acadêmica como uma abordagem adequada para o desenvolvimento de sistemas computacionais complexos. Segundo Russell e Norvig (2004), um agente de software é uma entidade capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores.

Engenharia de Software orientada a agentes [Castro, Alencar e Silva 2006] representa uma nova maneira de analisar, projetar, construir, gerenciar e manter software complexo envolvendo o paradigma de desenvolvimento orientado a agentes.

O processo de software é extremamente importante ao desenvolvimento de sistemas, uma vez que ordenam as atividades e artefatos relacionados e geralmente um processo toma como base uma metodologia de desenvolvimento. Dentre as metodologias de desenvolvimento destacamos MAS-*School* [Sardinha 2005].

O MAS-*School* utiliza um procedimento de avaliação das técnicas que consiste no desenvolvimento incremental do sistema. Primeiro, é desenvolvido um SMA onde o objetivo dessa primeira versão é simplesmente "testar a comunicação entre os agentes e a interação com o mundo externo" [Sardinha 2005]. Após a primeira versão, há um processo de inserção incremental de aprendizagem de máquina nos agentes do sistema. Cada nova versão do sistema pode ser resultado da inserção de um algoritmo ou alguma técnica de aprendizagem de máquina. Após inserir uma nova funcionalidade ou uma nova técnica de aprendizagem de máquina, o sistema é testado. Se for detectado que a técnica não melhorou o desempenho global do sistema, é necessária uma revisão da codificação. Após a revisão, o sistema é testado novamente.

A literatura da área dispõe de *frameworks* adaptados ao desenvolvimento de SMA, dentre elas destaca-se a plataforma JADE (*Java Agent Development Enterprise*)

[Bellifemine et al. 2007]. JADE é um *framework* implementado na linguagem Java e também um ambiente de execução de agentes, simplificando o desenvolvimento de SMAs através de uma arquitetura que está de acordo com as especificações FIPA (*Foundations of Intelligent Physical Agents*) [FIPA 2013], e com um conjunto de ferramentas gráficas que apoiam o desenvolvimento.

Uma extensão do JADE foi proposta por Lopes et al. (2011), na qual as entidades de MAS-ML 2.0 (*Multi Agent System Modeling Language*) [Gonçalves 2009] foram representadas de forma aderente ao JADE. A extensão chamada de JAMDER (*JADE to MAS-ML 2.0 Development Resource*) contempla as seguintes entidades de MAS-ML 2.0: ambientes, papéis de agente, organização e as arquiteturas de agentes.

2.2. Educação a Distância

A Educação a Distância (EAD) é uma modalidade do processo ensino-aprendizagem, na qual estudantes de diferentes idades e experiências anteriores estudam em grupos ou individualmente, nos locais que lhes for mais oportuno, fazendo uso de materiais autodidáticos, produzidos especificamente para esta finalidade, distribuídos através de diversos meios de comunicação [Bordenave 1988]. Esta não é uma abordagem recente e vem evoluindo de acordo com recursos disponíveis, caracterizando-se inicialmente, pelo uso de correspondência, depois, rádio, televisão e teleconferências e, finalmente, chegando aos recursos computacionais [Novello e Laurino, 2012].

No cenário atual a EAD é inerentemente associada à Tecnologia de Informação e Comunicação, sendo que os Ambientes Virtuais de Aprendizagem (AVA) são utilizados como ferramenta que viabiliza a realização de cursos nesta modalidade. De modo geral, um AVA refere-se ao uso de recursos digitais de comunicação, principalmente, através de softwares educacionais via web que reúnem diversas ferramentas de interação [Valentini e Soares 2005].

Vários AVA vêm sendo disponibilizados para uso, dentre eles podemos destacar o MOODLE (Modular Object-Oriented Dynamic Learning Environment) por ser amplamente utilizado em cursos de EAD nacional e internacionalmente. O MOODLE trata-se de um software livre disponível para download em https://www.moodle.org, este pode ser utilizado sem custos e vem viabilizando o desenvolvimento de cursos na modalidade EAD ao redor do mundo. Este software é disponibilizado em uma versão padrão (genérica) que pode ser customizada para o uso em determinado curso/instituição de ensino.

3. Processo de Desenvolvimento

Um processo de software é importante para guiar o desenvolvimento em qualquer domínio. Nesta seção é apresentado o processo de desenvolvimento de um SMA.

O processo baseia-se em alguns princípios da metodologia MAS-*School*, mais especificamente em relação ao desenvolvimento incremental do sistema, onde: a primeira versão do SMA visa a definição dos agentes e verficar a comunicação entre os agentes e a interação com o mundo externo [Sardinha 2005] e posteriormente deve haver a inserção incremental de aprendizagem de máquina nos agentes. Após inserir uma nova funcionalidade ou uma nova técnica de aprendizagem de máquina, o sistema é testado.

A modelagem de todo processo foi realizada na ferramenta EPF *Composer* [EPF 2014] conforme o Fluxo da Figura 1, que apresenta as fases do processo.

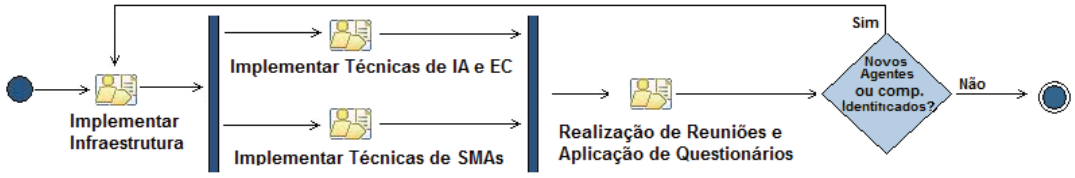

Figura 1 - Processo de Desenvolvimento do SMA para EAD.

Cada uma das fases é descrita a seguir:

Implementar Infraestrutura - Esta fase se dedica à implementação do SMA através de atividades de requisitos, projeto, implementação e testes. O fluxo inicia com a documentação dos agentes e os comportamentos que resolvem os problemas identificados. A segunda atividade envolve a modelagem do SMA e a documentação da arquitetura do sistema. A terceira atividade consiste na implementação dos agentes e entidades relacionadas, em conformidade com os problemas levantados.

Implementar Técnicas de IA (Inteligência Artificial) e EC (Engenharia do Conhecimento) - Esta fase objetiva a implementação de técnicas de IA nos agentes. Inicialmente ocorre a identificação das técnicas de IA e EC através de levantamento bibliográfico e identificação das técnicas mais adequadas aos problemas identificados na fase anterior. Em seguida as técnicas são implementadas nos agentes.

Implementar Técnicas de SMAs - Esta atividade ocorre em paralelo à fase anterior. Ela objetiva a implementação de técnicas de SMAs nos agentes. Inicialmente ocorre a identificação das técnicas de SMAs (tais como normas, reputação e coordenação) através de levantamento bibliográfico e identificação das técnicas mais adequadas aos problemas identificados na fase anterior. Em seguida as técnicas são implementadas nos agentes.

Realização de Reuniões e Aplicação de Questionários - Nesta fase, são realizadas reuniões com grupos de pesquisa em EAD com o intuito de identificar problemas e ações que podem ser adotadas no SMA de forma a resolver os problemas específicos detectados. Paralelamente questionários devem ser aplicados a outros grupos de pesquisa aos quais não for possível a realização de visita in loco.

Ao final das fases 1, 2 e 3, testes dos agentes são realizados através de simulação. E o fluxo pode ser percorrido várias vezes.

4. Implementar Infraestrutura – Relato experimental

As decisões tecnológicas tomadas nesta fase envolvem o uso de documentação, a modelagem do SMA através de MAS-ML 2.0, e do *framework* de desenvolvimento JADE juntamente com o JAMDER. Adicionalmente, protocolos de comunicação FIPA são utilizados para a interação entre agentes, e *framework* Hibernate [Hibernate 2014] recuperar os dados do repositório do ambiente virtual de aprendizagem. O AVA escolhido foi o MOODLE por ser um software livre utilizado em larga escala em cursos a distância, além de documentação disponível e comunidade de usuários.

As seções a seguir descrevem o relato experimental em relação às atividades de requisitos, projeto, implementação e teste da fase Implementar Infraestrutura.

4.1. Requisitos

Foi realizado um estudo bibliográfico abrangendo os anais das últimas edições do Simpósio Brasileiro de Sistemas de Informação (SBSI), Simpósio Brasileiro de Informática na Educação (SBIE), bem como artigos publicados da Revista Brasileira de Informática na Educação (RBIE), nos últimos cinco anos. Este estudo resultou na definição dos agentes e suas respectivas competências, resultando na documentação de cinco agentes, a saber:

Agente Companheiro Aprendizagem: Agente responsável por acompanhar o aluno durante o curso oferecendo auxílio no processo de aprendizagem como, por exemplo, exibindo mensagens de apoio ou reforço dependendo do desempenho do aluno, e sugerindo atividades complementares, como participação em fóruns e *chats*.

Agente Pedagógico: Agente responsável por acompanhar o usuário nos diferentes cursos, disciplinas, projetos em que este participar, e dar sugestões relacionadas às disciplinas como pré-requisitos e atividades a serem realizadas;

Agente Acompanhante de Tutores: Agente que monitora a participação dos tutores nos fóruns e na postagem de materiais, além de dar sugestões relacionadas ao melhor desempenho do tutor na disciplina como leitura do conteúdo antes da disciplina começar;

Agente Fornecedor de Materiais: Agente que envia conteúdo didático digital complementar referente à disciplina em questão. Este agente realiza seu trabalho em conjunto com o companheiro de aprendizagem que realiza um acompanhamento do desempenho do aluno. À medida que o aluno tem rendimento abaixo do esperado em tópicos este agente interfere e oferece material complementar por meio de postagem de arquivo ou link no MOODLE ou por meio do envio de *e-mail*, *twitter* ou SMS (*Short Message Service*). O cadastramento de materiais complementares é feito por meio de uma ferramenta de customização.

Agente Formador de Grupos: Agente capaz de sugerir autonomamente a formação de grupos de trabalho por afinidade de temas ou de perfis e aprendizagem. Este agente também trabalha em conjunto com o companheiro de aprendizagem;

Agente Auxiliar de Usabilidade: Responsável por dar sugestões sobre como fazer melhor uso do ambiente virtual de aprendizagem;

O conhecimento empírico dos autores em EAD, bem como o resultado do estudo bibliográfico foram utilizados para identificar problemas de EAD que poderiam ser mitigados ou solucionados pelos agentes. Assim sendo, cada um dos agentes foi documentado de forma mais específica. A documentação dos requisitos dos agentes é bastante simples, conta com uma descrição, as tarefas que o agente deve realizar e a estimativa de arquitetura que o mesmo deve seguir. O Quadro 1 exemplifica a documentação do Agente Companheiro de Aprendizagem.

Quadro 1 – Documento de requisito do Agente Companheiro de Aprendizagem

Agente Companheiro de Aprendizagem		
Descrição: Este agente é responsável por exibir mensagens de apoio ou reforço dependendo do desempenho do aluno, e sugerir atividades (como participação em fóruns e *chats*).		
Comportamentos		
B1) O agente deve notificar os	B2) O agente deve notificar os	B3) O agente deve notificar os

alunos quando há atividades avaliativas com datas próximas de se encerrar	alunos com notas baixas que eles precisam estudar mais e buscar auxílio do tutor	alunos quando a disciplina é criada para que eles leiam o conteúdo da disciplina
B4) O agente deve agendar encontro via *chat* entre os alunos e o tutor quando uma prova ou atividade avaliativa está próximo da data final. Os alunos e o tutor devem ser notificados	B5) O agente deve notificar os alunos quando uma atividade avaliativa ou prova tiver a data alterada	B6) O agente deve enviar *e-mail* para o aluno quando ele está há muito tempo sem entrar no ambiente
Comunicação		
C1) O agente deve comunicar-se com o agente acompanhante de tutores quando a situação B4 ocorrer	C2) O agente deve comunicar-se com o agente formador de grupos, informando os alunos com notas baixas, para que estes alunos sejam notificados a entrar em contato com líder de seu grupo no MOODLE. Esta comunicação ocorre quando o comportamento B2 ocorrer	

Uma visão geral dos Agentes e seus comportamentos documentados nos requisitos do SMA foram relacionados de forma resumida no mapa mental da Figura 2.

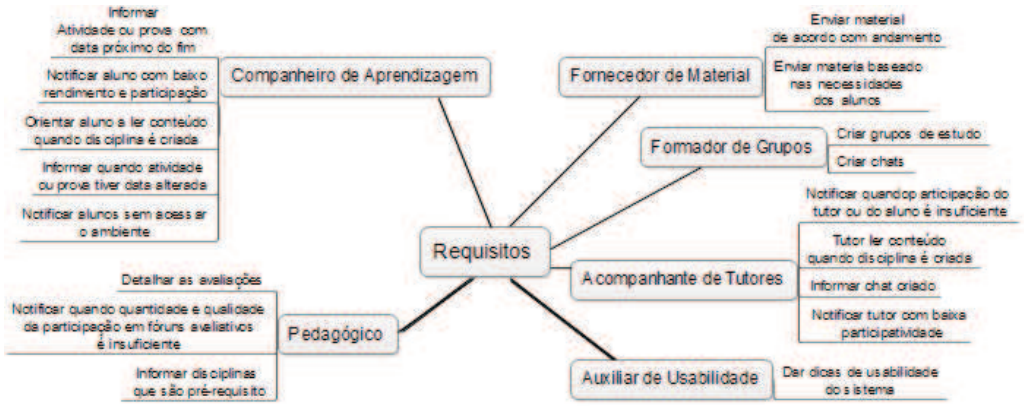

Figura 2 - Mapa Mental dos Agentes e Comportamentos

4.2. Modelagem

A arquitetura da aplicação é composta pelo MOODLE e pelo SMA. A Figura 3 ilustra a arquitetura do sistema. Onde podemos analisar os perfis do MOODLE, como aluno, professor e tutor, a forma de comunicação e a interação entre o SMA, o MOODLE e outras formas de comunicação.

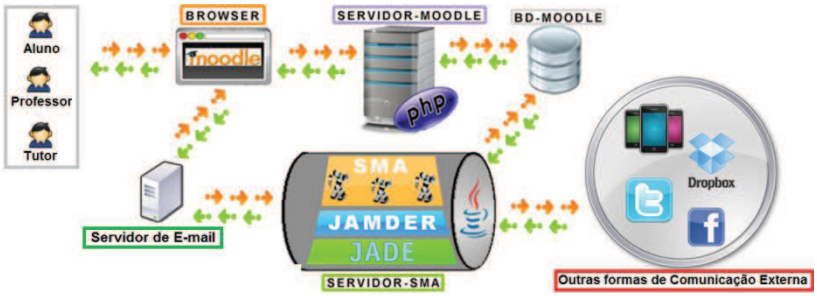

Figura 3 - Arquitetura da Aplicação

O SMA comunica-se com os dados do MOODLE através do acesso ao banco de dados deste AVA. O acesso ao banco de dados do MOODLE dá-se basicamente com duas finalidades: o ambiente capta os dados e atualiza as informações acessíveis aos agentes e, a postagem das mensagens dos agentes, além de utilizar *chats*, postagens em

fóruns e criação de links ou arquivos no ambiente (Recursos nativos do MOODLE).

Vale destacar as inúmeras formas de comunicação existentes na web nos dias de hoje e recursos da computação ubíqua e em nuvem. A integração de AVAs com estes recursos é importante uma vez que o ensino deve envolver cada vez mais os recursos que o aluno já utiliza no dia-a-dia, para que não seja algo distante da realidade do aluno, conforme mencionado por Brasileiro (1996). No entanto, muitas vezes os AVAs não estão integrados a estes recursos, sendo o SMA utilizado para realizar a integração. Desta forma, o desenvolvimento do SMA conta com uma classe utilitária que dá apoio ao envio de mensagens, arquivos e Links aos alunos através do próprio recurso de mensagens, arquivos e links existentes no MOODLE, do envio de *e-mails* através de recursos existentes na própria API Java, mensagens de texto através do *Twitter* ou envio de mensagens SMS através do Luso SMS (http://www.lusosms.com/).

Foi criada uma ferramenta de customização do SMA que funciona acoplada ao MOODLE, com a funcionalidade de customizar o texto das mensagens que os agentes enviam, a forma de envio (Mensagens do MOODLE, SMS, *e-mail* ou *twitter*), cadastrar material complementar seja em arquivos ou através de links para ser disponibilizado por uma das formas de envio já descritas. Também podem ser selecionados os agentes que irão executar e quais os cursos/as disciplinas onde eles irão executar. Estas funcionalidades estão disponíveis para o perfil de administrador do sistema, já o perfil de professor é capaz de postar arquivos referentes aos assuntos de suas disciplinas.

De forma complementar um painel foi criado para que as mensagens enviadas pelos agentes sejam visualizadas de forma mais fácil pelos alunos. Este painel fica visível na parte superior do ambiente desde o momento que o aluno realiza seu *login*.

O funcionamento do SMA envolve a plataforma de execução de agentes JADE em conjunto com o *framework* de desenvolvimento JAMDER utilizado para implementação das entidades (ambiente, agentes, papéis e organização) do SMA.

Figura 4 – Diagrama de Organização de MAS-ML simplificado dos agentes

Além destes agentes, o SMA conta com o ambiente que disponibilizará as informações necessárias, e papéis de agente definidos para cada um dos agentes implementados. A modelagem do SMA foi realizada através de MAS-ML 2.0 na ferramenta MAS-ML *tool* [Gonçalves 2009] e a Figura 4 apresenta o diagrama de Organização da linguagem de modelagem MAS-ML [Silva, 2004] para SMA do MOODLE.

4.3. Implementação

A implementação do SMA foi feita de acordo com a arquitetura da aplicação mostrado na subseção anterior, sendo que a implementação inicial das entidades do SMA em JAMDER foram derivados dos diagramas de MAS-ML *tool* através da abordagem de geração de código proposta por Lopes (2011).

Para exemplificar a implementação de um dos comportamentos, a ação "*InformarAndamento*" do Agente Companheiro de Aprendizagem, é responsável por informar ao aluno como está seu andamento no curso através de mensagens contendo: uma lista de atividades que o aluno realizou com suas respectivas notas, uma lista de atividades em aberto e uma frase de estímulo caso o aluno esteja com sua nota final abaixo da média ou um elogio, caso contrário. A implementação do método *execute()* dessa ação é mostrado na Figura 5. Esse método é sobrescrito por qualquer classe que representa uma ação em JAMDER assim como o *setup()* em todas que representam agentes.

Na linha 50 é instanciado um objeto da classe *GerenciaCurso* que contém todos os métodos para "alimentar" as *beans* e possui um método chamado *getCursos()* que retorna uma lista com todos os cursos no ambiente, o comportamento continua com o agente percorrendo essa lista com um laço iniciado na linha 53. O agente enviará uma mensagem para todos os alunos do curso corrente a partir de outro laço na linha 54, nas linhas 57 e 58 uma *String* chamada de *smallmessage* inicia o conteúdo da mensagem.

O próximo laço (linhas 59 a 68) percorre todas as atividades de cunho inteiramente avaliativas do curso, onde um teste é realizado para saber se aluno já cumpriu com as atividades concatenando o nome da atividade e sua respectiva nota à *smallmessage*, se o teste falhar as atividades serão adicionadas em uma lista alocada na linha 52 que armazena as atividades que o aluno não realizou.

O próximo passo do agente é inserir a nota geral do aluno no curso à mensagem, se a nota estiver abaixo da média (nesse caso 70), frases de estímulos serão concatenadas à mensagem. Posteriormente, se a lista *atividadesAlunoSemNota* (linha 52) não estiver vazia, terá seu conteúdo inserido em *smallmessage* com o laço entre as linhas 92 e 97. Com o conteúdo da mensagem completo, um objeto do tipo *Mensagem* será instanciado, contendo todos os atributos necessários para que a mensagem chegue ao aluno.

Os principais atributos requeridos são: o *id* de quem está enviando a mensagem (Linhas 51 e 106); o *id* do aluno destinatário (linhas 56 e 107); o assunto que diz respeito à mensagem (linha 105); o conteúdo da mensagem (linha 108) e a data (em milissegundos) que a mensagem será enviada (linhas 103 e 110). E finalmente na linha 111 a mensagem é enviada ao ambiente e após isso o mesmo *Thread* que atualiza as *beans* também é responsável por levar essa mensagem para a tabela *mdl*_message na base de dados do MOODLE.

```
@Override
public void execute(Environment env, Object[] params) {
    MoodleEnv envir = (MoodleEnv) env;
    mantemAtivo = envir.getMantemAgentesAtivos();
    if (!mantemAtivo)
        return;
    GerenciaCurso manager = envir.getGerenciaCurso();
    BigInteger useridfrom = new BigInteger("2");
    List<Atividade> atividadesAlunoSemNota = new ArrayList<Atividade>();
    for (Curso curso : manager.getCursos()) {
        for (Aluno aluno : curso.getAlunos()) {
            atividadesAlunoSemNota.clear();
            BigInteger useridto = aluno.getId();
            String smallmessage = "Prezado " + aluno.getCompleteName()+ ", \n";
            smallmessage += "Na disciplina " + curso.getFullName()+ " , seu atual andamento é: \n\n";
            for (AtividadeNota at : curso.getAtividadesNota()) {
                if (at.getAlunosComNotas().containsKey(aluno)) {
                    smallmessage += "Atividade " + at.getName() + ": \n";
                    smallmessage += "Sua nota: "+ at.getAlunosComNotas().get(aluno)+
                    " / Nota máxima: " + at.getNotaMaxima()+ "\n\n";
                } else {
                    Date dataAtual = new Date();
                    if (at.getDataFinal().before(dataAtual))
                        atividadesAlunoSemNota.add(at);
                }
            }

            for (AtividadeParticipacao at : curso.getAtividadesParticipacao()) {
                if (at.isAvaliativo()) {
                    if (at.getAlunosComNotas().containsKey(aluno)) {
                        smallmessage += "Atividade " + at.getName()+ ": \n";
                        smallmessage += "Sua nota: "+ at.getAlunosComNotas().get(aluno)+
                        " / Nota mÃ¡xima: " + at.getNotaMaxima()+ "\n\n";
                    } else {
                        Date dataAtual = new Date();
                        if (at.getDataFinal().before(dataAtual))
                            atividadesAlunoSemNota.add(at);
                    }
                }
            }
            BigDecimal nota = curso.getNotaGeralAlunos().get(aluno);
            if (nota != null) {
                smallmessage += "\n Sua nota final atual no curso é: "+ nota.toString();
                if (nota.intValue() < 70) {
                    smallmessage += "\n Prezado aluno, sua nota final está abaixo da média necessaria."+
                    " Procure estudar mais, peça ajuda aos colegas e busque se envolver mais nas atividades";
                } else {
                    smallmessage += "\n Parabéns! Continue estudando e buscando evoluir em sua nota.";
                }
            }
            if (!atividadesAlunoSemNota.isEmpty()) {
                smallmessage += "\n\nFaltam notas suas nas seguintes atividades:\n";
                for (Atividade at : atividadesAlunoSemNota) {
                    smallmessage += " - " + at.getName() + "\n";
                }
            }
            if (nota == null && atividadesAlunoSemNota.isEmpty())
                smallmessage += "\n\n Sem atividades no curso até o momento.";

            smallmessage += "\n";
            String fullmessage = smallmessage;
            Long time = System.currentTimeMillis();
            Mensagem msg = new Mensagem();
            msg.setSubject("Nova mensagem do Administrador");
            msg.setUseridfrom(useridfrom);
            msg.setUseridto(useridto);
            msg.setSmallmessage(smallmessage);
            msg.setFullmessage(fullmessage);
            msg.setTimecreated(time);
            ((MoodleEnv)env).addMensagem(msg);
        }
        done = true;
    }
}
```

Figura 5 - Implementação do Comportamento *InformarAndamento* do Agente Companheiro de Aprendizagem

4.4. Teste

Os testes foram realizados para analisar o comportamento do sistema multiagente em um ambiente controlado, onde outros agentes foram criados para simular comportamento de alunos, tutores e professores no MOODLE. Adicionalmente uma disciplina foi criada com um fórum avaliativo, duas atividades e uma prova, além disto, os agentes foram cadastrados como alunos do curso com acesso às atividades avaliativas deste.

Desta forma, os agentes de teste utilizaram uma classificação em seis níveis de desempenho, de acordo com a escala proposta por Sales (2010): Muito Bom, Bom, Regular, Fraco, Não Satisfatório e Neutro. Os comportamentos dos agentes da simulação foram implementados de acordo com esta classificação, sendo que o agente *AlunoMuitoBom* foi codificado com bom desempenho nas atividades avaliativas de modo a ficar com uma média de 9 a 10, já o agente *AlunoBom* foi programado para realizar as atividades avaliativas e ficar com média 7 a 8,99 e assim sucessivamente. Os agentes criados com o perfil de desempenho fraco foram programados para postar mensagens no último dia do fórum avaliativo e o agente com desempenho não satisfatório foi desenvolvido para não postar no fórum avaliativo.

Também foram criados agentes para tutores participativos e não participativos e professores atuantes e não atuantes e estes foram utilizados em diferentes simulações. Deste modo os agentes do SMA e da simulação foram utilizados juntamente com combinações dos agentes tutores e professores, sendo realizadas quatro simulações.

Assim sendo, o comportamento dos agentes do SMA puderam ser analisados de modo a aferir se realmente estavam acontecendo conforme o esperado. Para tanto mensagens, arquivos e links que foram enviadas pelos agentes do SMA aos agentes da simulação foram analisadas, sendo feitas buscas por situações que o agente deveria ter enviado mensagens e não o fez. Ao final as correções foram realizadas e novos testes foram realizados, no qual não foi possível mais encontrar falhas nos agentes do SMA e sendo possível partir para a implantação do SMA em uma disciplina real.

4.5. Considerações Finais

O processo, juntamente com os artefatos gerados para requisitos, modelagem e o código do SMA podem ser utilizados por outros pesquisadores ou instituições e estão disponíveis em http://code.google.com/p/moodle2gesma/source/browse/.

Atualmente o SMA foi implantado no dia 15/01/2014, na disciplina de Linguagem de Programação I do curso de Licenciatura em Informática oferecido pela Universidade Aberta do Brasil (UAB) em parceria com a Universidade Estadual do Ceará (UECE). Assim sendo o comportamento destes está sendo analisado em um cenário real e em breve será utilizado de forma mais abrangente em todas as disciplinas.

5. Trabalhos Relacionados

Em Campana et al. [2008] o MOODLE não é o AVA utilizado e sim o AVAUFES (Ambiente Virtual de Aprendizagem da UFES) [AVAUFES 2014] desenvolvido na Universidade Federal do Espírito Santo. Eles desenvolveram uma camada de agentes, implementados em JADE, a ser utilizada em AVA de uma maneira geral, com aplicação inicial no AVAUFES. Alguns perfis de agentes são definidos e identificados todos

como sendo reativos a eventos pontuais que ocorrem na comunicação entre as partes que utilizam o AVA. Regras simples são utilizadas e executadas sempre que certos eventos ocorrem no ambiente. No entanto, técnicas de Engenharia de Software não foram utilizados a contento, uma vez que um processo não foi criado, a documentação e modelagem do SMA é inexistente e a arquitetura da aplicação não foi apresentada.

Em Bremgartner e Netto (2011), alunos são categorizados em relação a suas habilidades e competências de maneira parte deles seja indicado como ajudante de alunos que tem dificuldades. Uma ontologia de modelo de aluno é definida baseado em dados estáticos, como por exemplo, dados de cadastro, e dados dinâmicos, como por exemplo, dados obtidos a partir da interação do aluno com o AVA. O desenvolvimento do SMA do presente trabalho segue uma abordagem baseada em MAS-*School*, por outro lado, o trabalho de Bremgartner e Netto (2011) não apresenta uma contribuição relacionada a engenharia de software orientada a agentes.

6. Conclusões e Trabalhos Futuros

O ensino na modalidade EAD vem crescendo devido ao apoio de ferramentas computacionais, como os AVAs. No entanto o acompanhamento dos alunos é uma atividade propensa a falhas que pode ser beneficiada com o uso de ferramentas computacionais autônomas como Sistemas Multiagentes (SMAs). Os SMAs podem colaborar significativamente neste cenário através da identificação de situações de problemas e utilizar a comunicação com alunos e demais envolvidos, visando a correção dos problemas identificados.

Neste artigo foi apresentado o desenvolvimento de um SMA para o AVA MOODLE utilizando técnicas de engenharia de software orientada a agentes, a saber: um processo criado especificamente para o desenvolvimento deste projeto, uma forma de documentar requisitos simplificada, diagramas de MAS-ML e implementação utilizando JADE + JAMDER, de modo a solucionar problemas recorrentes da área de Educação a Distância através de software autônomo. Os resultados do desenvolvimento tomando este processo como base foram apresentados neste artigo, a saber: requisitos, a modelage do sistema multiagente e sua arquitetura, a implementação dos comportamentos dos agentes para resolução dos problemas de EAD identificados e uma simulação realizada para testar o comportamento do SMA desenvolvido utilizando as atividades da fase Implementar Infraestrutura.

Como trabalhos futuros podemos citar uma análise qualitativa a qual será realizada na disciplina de Linguagem de Programação I da UAB-UECE para analisar o desempenho dos agentes e a percepção dos alunos acerca dos mesmos. Além disto, o desenvolvimento terá continuidade com as fases restantes do processo, a saber: Fase de Implementar Técnicas de IA e EC, Fase de Implementar Técnicas de SMAs e Fase de Realização de Reuniões e Aplicação de Questionários.

Referências

AVAUFES. (2014) “Ambiente Virtual de Aprendizagem da UFES”, http://ava.ufes.br/, Janeiro.

Bellifemine, Fabio L., Giovanni Caire, and Dominic Greenwood. (2007) "*Developing multi-agent systems with jade (wiley series in agent technology*).", New Jersey.

Bordenave, Juan Días. (1988) "Pode a Educação à Distância ajudar a resolver os problemas educacionais do Brasil?" In: *Tecnologia Educacional*, Rio de Janeiro.

Brasileiro, S. A. (1996) "O computador como mediador dos processos pedagógicos. Um estudo exploratório em escolas de Belo Horizonte". UFMG.

Bremgartner, V., & Netto, J. F. M. (2011). "Auxílio Personalizado a Estudantes em Ambientes Virtuais de Aprendizagem Utilizando Agentes e Competências". In *Anais do Simpósio Brasileiro de Informática na Educação* (Vol. 1, No. 1).

Campana, V. F., Sanches, D. R., Tavares, O. D. L., & Souza, S. F. D. (2008). Agentes para Apoiar o Acompanhamento das Atividades em Ambientes Virtuais de Aprendizagem. In *Anais do Simpósio Brasileiro de Informática na Educação*.

Castro, J.; Alencar, F. ; Silva, C. (2006). Engenharia de Software Orientada a Agentes. Atualizações em Informática. Rio de Janeiro: PUC-Rio.

EPF (2014) "*Eclipse Process Framework Project*", https://www.eclipse.org/epf, Janeiro.

FIPA (2013) "*Foundations of Intelligent Physical Agents*", http://jade.tilab.com, Março.

Gonçalves, E. J. T. (2009) "Modelagem de arquiteturas internas de agentes de software utilizando a linguagem MAS-ML 2.0". Dissertação de Mestrado. Universidade Estadual do Ceará. Centro de Ciência e Tecnologia. Fortaleza.

Hibernate. (2014) "*Hibernate*", http://hibernate.org/, Janeiro.

Lopes, Y. S.; Gonçalves, E. J. T.; Cortés, M. I. e Freire, E. S. S. (2011) "*Extending JADE Framework to Support Different Internal Architectures of Agents*" In: *9th European Workshop on Multi-agent Systems (EUMAS 2011)*, Maastricht, Holanda.

Mercado, L. P. M. (2007) "Dificuldades na Educação a Distância Online". Em: *13º Congresso Internacional de Educação a Distância*,Curitiba-PR.

Moodle. (2013) "MOODLE". http://moodle.org, Novembro.

Novello, T.P., Laurino, D.P., Educação a distância: seus cenários e autores. Revista Iberoamericana de Educación. Número 58/4, 2012.

Padilha, T. P.,Jácome,T.(2002) "O Uso de Técnicas de Modelagem de Agentes em Ambientes Educacionais".Em:*VI Congresso Iberoamericano de Informática Educativa*.

Russell, S. Norvig, P. (2004) "Inteligência Artificial: uma abordagem moderna". 2 Ed. São Paulo: Prentice-Hall.

Silva, V. T. da (2004). Uma linguagem de modelagem para sistemas multi-agentes baseada em um framework conceitual para agentes e objetos. Tese de doutorado. Rio de Janeiro: PUC, Departamento de Informática.

Sales, G. L. (2010) LEARNING VECTORS (LV): Um Modelo de Avaliação da Aprendizagem em EaD online Aplicando Métricas Não-Lineares. Tese (Doutorado em Engenharia de Teleinformática) — Universidade Federal do Ceará. Fortaleza-CE.

Sardinha, J. A. R. P. (2005) MAS-School e ASYNC: Um Método e um Framework para Construção de Agentes Inteligentes, Tese de doutorado em Informática. PUC-RJ.

Valentini, C. B., Soares, E. M. S. (orgs.). (2005) Aprendizagem em Ambientes Virtuais: compartilhando ideias e construindo cenários. Caxias do Sul: EDUCS.