UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JOSÉ FERDINANDY SILVA CHAGAS

SISTEMA MULTIAGENTE DE APOIO À GESTÃO DE CONHECIMENTO EM PROJETOS DE SOFTWARE QUE IMPLEMENTAM CMMI-NÍVEL 2

MOSSORÓ – RN

2013

JOSÉ FERDINANDY SILVA CHAGAS

SISTEMA MULTIAGENTE DE APOIO À GESTÃO DE CONHECIMENTO EM PROJETOS DE SOFTWARE QUE IMPLEMENTAM CMMI-NÍVEL 2

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação – associação ampla entre a Universidade do Estado do Rio Grande do Norte e a Universidade Federal Rural do Semi-Árido, para a obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Francisco Milton Mendes Neto – UFERSA.

Coorientadora: Profa. Dra. Mariela Inés Cortés – UECE.

MOSSORÓ – RN

2013

Ficha catalográfica preparada pelo setor de classificação e catalogação da Biblioteca "Orlando Teixeira" da UFERSA

C426s Chagas, José Ferdinandy Silva

Sistema multiagente de apoio à gestão de conhecimento em projetos de software que implementam cmmi-nivel 2. / José Ferdinandy Silva Chagas -- Mossoró-RN: 2013.

85f.: il.

Dissertação (Pós-graduação em Ciências da Computação) – Universidade Federal Rural do Semi-Árido. Pró-Reitoria de Pesquisa e Pós-Graduação.

Orientador: Profº. Dr. Francisco Milton Mendes Neto
Coorientador: Profª. Dra. Mariela Inés Costés

1.Sistemas multiagente. 2.Ontologias. 3.Gestão de conhecimento. 4.Gestão de projetos. I.Título.

CDD:004

Bibliotecária: Marilene Santos de Araújo
CRB-5/1033

JOSÉ FERDINANDY SILVA CHAGAS

SISTEMA MULTIAGENTE DE APOIO À GESTÃO DE CONHECIMENTO EM PROJETOS DE SOFTWARE QUE IMPLEMENTAM CMMI-NÍVEL 2.

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de Mestre em Ciência da Computação.

APROVADA EM: ___ / ___ / _____.

BANCA EXAMINADORA

_____

Prof. Dr. Francisco Milton Mendes Neto – UFERSA
Presidente

_____

Prof. Dra. Mariela Inés Cortés – UECE
Primeiro Membro

_____

Prof. Dr. Pedro Fernandes Ribeiro Neto – UERN
Segundo Membro

_____

Prof. Dr. Gabriel Antoine Louis Paillard – UFC
Terceiro Membro

DEDICATÓRIA

Dedico este trabalho a todas as pessoas que acreditaram, apoiaram e me orientaram nessa difícil caminhada.

In memoriam Francisco Xavier Chagas

AGRADECIMENTOS

Agradeço a Deus meu fiel escudeiro. Sem ele nada disso seria possível. Desde a minha entrada até a conclusão deste trabalho.

A minha família que me apoiou muito em todo o processo e que se esforçou para que eu seguisse, mesmo na distância e dificuldades, com a busca para a realização desse sonho.

Ao meu orientador Francisco Milton Mendes, pessoa de valor inestimável. O grande idealizador, incentivador e colaborador de todos os trabalhos que desenvolvi desde a graduação até o mestrado.

A minha coorientadora Mariela Inés Cortés, pelo apoio e grandes contribuições para realização deste trabalho.

Ao Programa de Pós-graduação em Ciência da Computação UERN/UFERSA, por me permitir desenvolver esse trabalho e atingir mais esse objetivo de minha vida.

A Jessica Reenara, minha amada, por todo o apoio e compreensão principalmente nos momentos difíceis. Uma grande fonte de motivação nos momentos em que pensei em desistir.

Aos membros da banca Dr. Pedro Fernandes Ribeiro Neto e Dr. Gabriel Antoine Louis Paillard por aceitar participar da banca de avaliação deste trabalho de conclusão.

EPÍGRAFE

Bora rapaz desenrola! (Francisco Xavier Chagas)

RESUMO

Um dos principais modelos usados para melhorar os processos de desenvolvimento de software é o CMMI (*Capability Maturity Model Integration*). Organizações de software enfrentam muitos desafios para adotar esse modelo. Esta adoção requer muitos recursos: materiais, financeiros e humanos. Estes desafios aumentam em pequenas e médias empresas. Entre os problemas na implantação do CMMI está o aumento da carga cognitiva para o gerente de projeto. O gerente deve gerenciar projetos em andamento, enquanto entende e atende as recomendações do modelo. Este trabalho apresenta uma abordagem baseada em um sistema *web* multiagente. O sistema, chamado VPM, auxilia a gestão do conhecimento e automação de processo de gestão. O VPM fornece tarefas necessárias para alcançar e manter os objetivos de modelo de melhoria de processo. O objetivo deste trabalho é reduzir a carga cognitiva de gerente de projeto para implantação e manutenção de CMMI Nível-2 em pequenas e médias empresas. O aplicativo monitora e envia notificações para o gerente de projeto dentro das recomendações do CMMI e oferece alguns recursos para atendê-las.

Palavras-Chave: sistemas multiagente, ontologias, gestão de conhecimento, gestão de projetos, CMMI.

ABSTRACT

One of the main models used to improve the processes of software development is the CMMI (Capability Maturity Model Integration). Software organizations faces many challenges to adopt this model. This adoption requires many resources: financial, material and human. These challenges increase to small and medium enterprises. Among the problems in deploying CMMI is the increased cognitive load for the project manager. The manager must manage ongoing projects while understands and meets the recommendations of the model. This work presents an approach based on a multiagent web system. The system, called VPM, assists knowledge management and automation of management process. The VPM provides tasks necessary to achieve and maintain the goals of process improvement model. The objective of this work is reduces the cognitive load of project manager to deploying and maintaining CMMI Level-2 in small and medium enterprises. The application monitors and sends notifications to the project manager within the recommendations of CMMI and provides some resources to answer them.

Keywords: multiagent system, knowledge management, project management, CMMI.

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE SIGLAS

ACL – *Agent Communication Language*

AI – Assistente de Interface

AID – *Agent Identifier*

AMC – Assistente de Monitoramento e Controle

AMS – *Agent Management System*

ANSI – *American National Standards Institute*

AP – Assistente de Plano

API – *Application Programming Interface*

AR – Assistente de Requisitos

CMMI – *Capability Maturity Model Integration*

CMMI-ACQ – *CMMI for Acquisition*

CMMI-DEV – *CMMI for Development*

CMMI-SVC – *CMMI for Services*

CMMs – *Capability Maturity Models*

DF – *Directory Facilitator*

EIA SECM – *Electronic Industries Alliances' System Engineer Capability Model*

ES – Engenharia de Software

FIPA – *Foundation for Intelligent Physical Agents*

FIPA-ACL – *Specifications of the Agent Communication Language*

FIPA-SL – *Content Language Specification*

IEEE – *Institute of Electrical and Electronics Engineers*

IPD-CMM – *Integrated Product Development CMM*

JADE – *Java Agent Development Framework*

MTS – *Message Transport System*

OA – Ontologia de Atividades

ODS – Organizações de Desenvolvimento de software

OR – Ontologia de Riscos

ORH – Ontologia de Recursos Humanos

OWL – *Web Ontology Language*

PMA – *Project Management Application*

PMBOK – *Project Management Body of Knowledge*

PMI – *Project Management Institute*

RDF – *Resource Description Framework*

RDFS – *Resource Description Framework Schema*

SEI – *Software Engineering Institute*

SMA – Sistema Multiagente

SPI – *Software Process Improvement*

SW-CMM – *Capability Maturity Model for Software*

UML – *Unified Modeling Language*

VPM – *Virtual Project Manager*

W3C – *World Wide Web Consortium*

SUMÁRIO

1 INTRODUÇÃO

O processo de software constitui um processo intensivo em conhecimento, envolvendo diversos perfis, que manipulam uma gama de informações. A adoção de uma solução que satisfaça todas as necessidades de conhecimento relacionadas aos processos da Engenharia de Software (ES) não é uma tarefa trivial. Isto se torna mais difícil quando a organização adota um modelo de qualidade de processos, como o CMMI (*Capability Maturity Model Integration*) (CHRISSIS, KONRAD e SHRUM, 2011). O modelo CMMI, no momento da escrita deste trabalho, está na versão 1.3 e é composto por 3 (três) modelos principais: o CMMI-ACQ (*CMMI for Acquisition*), CMMI-DEV (*CMMI for Development*), CMMI-SVC (*CMMI for Services*). O modelo foco do trabalho é o CMMI-DEV, que é formado por 22 (vinte e duas) áreas de processos, cada uma com suas metas genéricas e específicas.

Apesar dos investimentos por parte de instituições de pesquisa e das Organizações de Desenvolvimento de Software (ODS) em ambientes automatizados de suporte a processos de software, não se tem alcançado a qualidade desejada e a produtividade esperada da equipe. Na ES, a experiência de trabalho, que ajuda a evitar erros do passado e melhorar decisões futuras, ainda reside, principalmente, nos colaboradores da ODS.

A implantação de modelos de melhoria de processos cria desafios para as ODS, principalmente por inserir passos adicionais que sobrecarregam o gerente de projetos, além da existência de possíveis conflitos entre a cultura organizacional e as recomendações do modelo. O desafio torna-se maior quando a organização em questão é de pequeno ou médio porte, pois não possui os recursos financeiros e humanos para incentivar e executar a implantação do modelo de melhoria do processo de software de modo adequado (FURUCHO e AGUIAR, 2012).

1.1 PROBLEMÁTICA

O trabalho apresentado em (HUANG e ZHANG, 2010) aborda os problemas que surgem quando o CMMI é implantado em pequenas e médias empresas. Um dos problemas que o artigo aponta é que o processo de implantação do CMMI requer muitos recursos financeiros, humanos e tecnológicos. Além disso, causa um aumento na carga de trabalho do

gerente de projetos e isto leva a um relativo aumento na expropriação de recursos. Outro fator importante que afeta a melhoria dos processos é a falta de pessoal especializado dedicado à qualidade dos processos.

Segundo Huang e Zhang (2010), o CMMI apresenta uma média de 600 (seiscentas) práticas entre gerais e específicas. Todas essas práticas precisam ser planejadas, monitoradas, medidas, analisadas e validadas simultaneamente com as atividades de gerenciamento do projeto. Este é um processo muito complexo para pequenas e médias empresas. Devido aos vários fatores técnicos e humanos envolvidos na gestão de projetos, essa é uma atividade de grande complexidade. Em vista dessa complexidade, a função desempenhada por um gerente de projetos requer a capacidade de análise e processamento de grandes quantidades de informações. Essa complexidade pode gerar problemas na gestão, refletindo diretamente na qualidade, custo e tempo de entrega do projeto.

Assim, é muito importante utilizar ferramentas de suporte ao CMMI durante o processo de implementação do modelo. Tais ferramentas podem contribuir para minimizar a carga cognitiva do gerente de projeto que, além de precisar monitorar todos os aspectos relativos à execução do projeto em si, precisa também ter conhecimento das práticas relativas ao CMMI.

Este trabalho pretende contribuir com a gestão de projetos de software propondo uma solução automatizada de apoio ao gerente de projetos seguindo as especificações das áreas de processo do CMMI-Nível 2. Esse nível de maturidade é o primeiro grande desafio para as organizações em suas iniciativas para melhoria de processos. No CMMI-Nível 2 a organização precisa atender aos requisitos básicos para o correto gerenciamento de um projeto de software. Nesse nível a organização possui o mínimo controle sobre seus processos para que possa sistematizá-los e melhorá-los.

1.2 OBJETIVO

O objetivo deste trabalho é auxiliar na implantação e manutenção do Nível 2 do CMMI, reduzindo a carga cognitiva do gerente de projetos. A ferramenta proposta, denominada de VPM (*Virtual Project Manager*), consiste em um sistema web suportado por agentes de software que monitoram e recomendam tarefas seguindo as práticas do CMMI com base nos processos do PMBOK (*Project Management Body of Knowledge*) (ANSI e PMI,

2008), além de notificar o gerente de projetos quanto a outras metas que os agentes não podem controlar.

O trabalho realizado envolve a integração de agentes de suporte ao CMMI a uma ferramenta de gerenciamento de projetos. Essa integração visa auxiliar na Gestão de Conhecimento (DAVENPORT e PRUSAK, 1998) e realizar a automação de algumas tarefas recomendadas nas áreas de processo do CMMI para o Nível 2. Esse nível é considerado um grande desafio para implantação do CMMI visto que é o marco inicial. As tarefas apoiadas pelo VPM estão relacionadas com as seguintes **áreas de processo**: gerenciamento de requisitos; monitoramento e controle do projeto; e planejamento do projeto.

Para prover suporte automatizado à implantação do CMMI, é utilizada a tecnologia de agentes de software. Para representação do conhecimento que é utilizado pelos agentes do sistema são utilizadas ontologias. Esse modelo de conhecimento é utilizado pelos agentes que auxiliam o gerente de projeto na gestão do conhecimento relacionado ao projeto.

1.3 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação foi organizada em cinco capítulos. O Capítulo 2 contextualiza os fundamentos teóricos do CMMI, ontologias, sistemas multiagente, gestão de conhecimento, gestão de projetos e ferramentas de gerenciamento de projetos. O Capítulo 3 apresenta alguns trabalhos relacionados com a proposta apresentada. O Capítulo 4 descreve a abordagem proposta neste trabalho. O Capítulo 5 aborda as considerações finais sobre o trabalho e os objetivos futuros para a sua continuação.

2 REFERENCIAL TEÓRICO

A seguir é realizada uma breve contextualização sobre os principais conceitos envolvidos no desenvolvimento do trabalho.

2.1 MODELOS DE MELHORES PRÁTICAS

De acordo com o CMMI, que busca a melhoria dos processos de software, o primeiro ponto para melhoria dos processos de uma organização é um gerenciamento mínimo dos processos. Segundo o guia PMBOK (ANSI e PMI, 2008), a crescente aceitação do gerenciamento de projetos reflete o reconhecimento dos profissionais da área que a aplicação adequada de processos, técnicas e ferramentas pode influenciar de forma significativa no sucesso dos projetos. A seguir esses modelos são detalhados.

2.1.1 CMMI

O CMMI (*Capability Maturity Model Integration*) (CHRISSIS, KONRAD e SHRUM, 2011) é um modelo evolutivo criado pelo Instituto de Engenharia de Software (*SEI - Software Engineering Institute*) para melhoria de processos organizacionais com base em alguns CMMs (*Capability Maturity Models*) que se originaram devido às diferentes necessidades das organizações. Inicialmente, o modelo SW-CMM (*Capability Maturity Model for Software*) era a principal referência em qualidade para o processo de engenharia de software. Porém, sua estrutura não atendia às organizações que trabalhavam em áreas multidisciplinares. Assim, surgiram outros CMMs como o EIA SECM (*Electronic Industries Alliances' Systems Engineer Capability Model*) e o IPD-CMM (*Integrated Product Development CMM*). A heterogeneidade de CMMs motivou a criação do CMMI, um modelo integrado, flexível e modular. A Figura 1 apresenta de forma resumida a evolução dos modelos de melhoria que culminou na criação do CMMI.

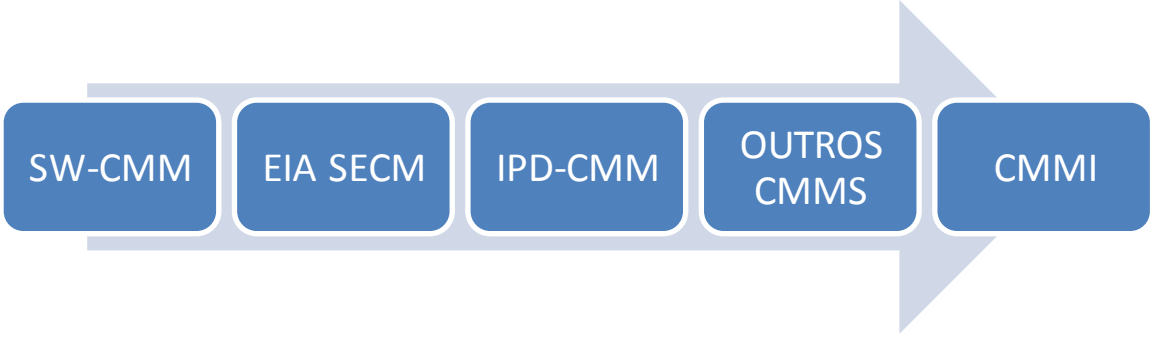


Figura 1. Evolução dos CMMs (CHAGAS, 2013)

O CMMI apresenta dois tipos de abordagens para melhoria dos processos, chamadas representações. Uma é a representação por estágios que define níveis de maturidade, enquanto a outra é a representação contínua que define níveis de capacidade. As duas representações apresentam o mesmo conteúdo e os mesmos componentes. Ambas as representações se direcionam à melhoria dos processos da organização, porém por caminhos diferentes.

Em sua representação por estágios, o CMMI-DEV é dividido em 5 (cinco) níveis de maturidade. Esses níveis representam a evolução da maturidade dos processos utilizados na ODS e podem ser descritos como a seguir (CHRISSIS, KONRAD e SHRUM, 2011):

- Nível 1: É o nível inicial, quando a ODS não possui um ambiente controlado para os seus processos. Nesse nível, geralmente, o sucesso vai depender da capacidade dos profissionais da equipe. Geralmente, os projetos, apesar de serem entregues, excedem o tempo e os custos previstos;
- Nível 2: A ODS já possui certo controle. Nesse nível, o ambiente dos processos é gerenciado, o que permite que a ODS replique os processos, adaptando-os às necessidades dos projetos. As principais áreas de processos são: gestão de configuração, gestão de qualidade, gestão de aquisição, análise, monitoramento, planejamento e gestão de requisitos;
- Nível 3: A ODS já alcançou as metas genéricas das áreas de processos dos níveis 2. Nesse nível, os processos são completamente compreendidos e descritos em padrões, procedimentos, ferramentas e métodos. As principais áreas de processos envolvidas são: revisão, coordenação da equipe, gestão de integração, programa de treinamento e definição do processo organizacional;
- Nível 4: A ODS possui informação, em nível quantitativo, para controle dos processos. Para isso, utilizam métodos estatísticos e outros métodos quantitativos. As principais áreas de processos são: gestão de qualidade e gestão quantitativa dos processos;
- Nível 5: A ODS possui controle total dos processos e pode realizar um melhoramento contínuo a partir do seu entendimento, e das análises qualitativas e quantitativas. As principais áreas de processos são: gerenciamento de mudanças e prevenção de falhas.

A **Erro! Fonte de referência não encontrada.** ilustra os níveis de maturidade do MMI seguindo a representação por estágios. O Nível 1 é desconsiderado, pois não há controle sobre os processos. Para os níveis seguintes é necessário contemplar todas as recomendações

do nível anterior. Por exemplo, para atingir o Nível 4, é necessário ter alcançado o Nível 3, e para este consequentemente o Nível 2.

Considerando a complexidade da abordagem por estágios, que envolve todos os processos de uma organização, o CMMI também desenvolveu uma abordagem contínua onde são avaliados os níveis de capacidade em departamentos específicos da organização. Na representação contínua, são definidos 4 (quatro) níveis de capacidade para processos de uma organização. São eles:

- Nível 0 (Incompleto): Os processos não são realizados ou são parcialmente realizados;
- Nível 1 (Realizado): Os processos necessários para produção dos devidos produtos são realizados, mas ainda não são institucionalizados, o que pode ocasionar a perda da melhoria com o tempo;
- Nível 2 (Gerenciado): Os processos são gerenciados, isto é, são planejados e executados adequadamente, gerando as saídas correspondentes;
- Nível 3 (Definido): Os processos são gerenciados da forma definida pela organização. Esses processos são documentados e padronizados para todos os projetos da organização.

As organizações que cumprem com as exigências do modelo recebem uma certificação CMMI-DEV com base no seu nível de maturidade ou capacidade. O foco deste trabalho é o nível 2 de maturidade do CMMI-DEV. Nesse nível, a organização se dedica a melhoria dos processos básicos que se não forem realizados resultarão em problemas para o projeto e/ou para a organização. Segundo o SEI (SEI, 2012), entre os anos de 2009 a 2011, aproximadamente 636 (seiscentas e trinta e seis) empresas alcançaram o Nível 2 do CMMI.

O modelo CMMI-DEV constitui uma referência para melhoria de processos, porém não enfatiza detalhes sobre como as organizações devem proceder para atingir tais requisitos. Isto ocorre pela variação das disciplinas envolvidas nos processos de cada ODS. Contudo, segundo Ehsan *et al.* (2010), o PMBOK pode ser utilizado como referência para preencher essa lacuna e detalhar técnicas e procedimentos úteis para cumprir com o modelo CMMI. Assim, este trabalho utiliza o PMBOK como guia para detalhar as atividades necessárias para melhoria dos processos das organizações.

2.1.2 PMBOK

O PMBOK consiste em um guia para a gestão de projetos, desenvolvido pelo Instituto de Gestão de Projetos (*PMI - Project Management Institute*) com base nas boas práticas amplamente reconhecidas pelos profissionais afiliados ao instituto. O guia não é uma metodologia para gestão de projetos, mas ele apresenta uma visão geral de boas práticas que podem aumentar a possibilidade de sucesso nos mais variados projetos.

O modelo é estruturado em 9 (nove) áreas de conhecimento que abrangem 42 (quarenta e dois) processos e são classificados em 5 (cinco) grupos de processos (ANSI e PMI, 2008). Os grupos de processos são:

Processos de Iniciação – agrupa os processos realizados durante a definição de um novo projeto ou nova fase de um projeto em andamento;

Processos de Planejamento – processos relacionados com a identificação do escopo geral das atividades e com a especificação dos requisitos e objetivos do projeto;

Processos de Execução – processos que definem as tarefas que serão executadas de acordo com o plano do projeto;

Processos de Monitoramento e Controle – processos de análise e mudança de planos para que sejam alcançados os objetivos do projeto;

Processos de Encerramento – processos para finalizar todas as atividades do projeto encerrando formalmente o projeto ou a fase.

Esses grupos de processos apresentam a estrutura básica dos processos envolvidos em cada fase de um projeto ou de um projeto completo que possui apenas uma fase (ANSI e PMI, 2008). As áreas de conhecimento especificadas pelo PMBOK detalham todos os processos da estrutura de grupos. As áreas são (ANSI e PMI, 2008):

Gerenciamento de Integração – identificar, definir, combinar, unificar e coordenar os vários processos e atividades de gerenciamento;

Gerenciamento de Escopo – processos necessários para assegurar que o projeto inclui todo o trabalho necessário, e apenas o necessário, para terminar o projeto com sucesso;

Gerenciamento de Tempo – processos para gerenciar o término pontual do projeto;

Gerenciamento de Custo – processos envolvidos na estimativa, orçamentos e controle de custos, para que seja concluído com o orçamento aprovado;

Gerenciamento de Qualidade – processos e atividades que determinam a qualidade para que o projeto satisfaça tais necessidades;

Gerenciamento de Recursos humanos – processos que organizam e gerenciam a equipe do projeto;

Gerenciamento de Comunicações – processos necessários para que as informações sejam geradas, coletadas, distribuídas, armazenadas, recuperadas e organizadas de modo apropriado;

Gerenciamento de Riscos – processos de planejamento, identificação, análise, planejamento de respostas, monitoramento e controle de riscos;

Gerenciamento de Aquisições – processos necessários para comprar ou adquirir produtos, serviços ou resultados externos à equipe do projeto.

Este guia não é restrito ao gerenciamento de projetos de software. Ele é direcionado para qualquer tipo de projeto. O PMBOK enfatiza que suas recomendações são boas práticas, que não necessariamente precisam ser seguidas exatamente como descritas pelo guia. É responsabilidade do gerente de projetos escolher quais processos e atividades serão executadas durante um projeto.

2.2 ONTOLOGIAS

Historicamente, o termo Ontologia tem origem do grego “ontos”, que significa “ser” e “logos”, que significa “palavra”. É um termo novo na história da filosofia, introduzido originalmente com o objetivo de distinguir o estudo do ser como tal (ou seja, do ser humano em sua essência) a partir do estudo dos vários tipos de outros seres das ciências naturais. O termo original é a palavra “categoria”, que pode ser usada para classificar e categorizar alguma coisa (ALMEIDA e BAX, 2003).

No campo da inteligência artificial, os primeiros conceitos que surgiram definiram ontologias como “termos e relações que compreendem o vocabulário de uma área, como também as regras para combinar estes termos e relações para definir extensões deste vocabulário” (NOVELLO, 2002).

A literatura sobre ontologias apresenta uma série de definições distintas. Por sua vez, essas diferentes definições apresentam pontos de vista distintos e até complementares para uma mesma realidade, uma vez que seu significado tende a variar conforme o objetivo do uso da ontologia (ZAMITH, 1998). As ontologias são estruturas de organizações que possibilitam uma compreensão comum e compartilhada de um domínio do conhecimento e, assim, possuem papel importante no intercâmbio de informações. Elas podem ser consideradas extensões das taxonomias, contendo, vocabulários e seus significados, com semântica expressiva e bem definida. Sendo assim, essas estruturas podem ser utilizadas, por exemplo, para expressar semântica legível por máquinas de forma eficiente no contexto da *Web* Semântica (BERNERS-LEE, HENDLER e LASSILA, 2001). Possibilitam também a comunicação entre os agentes envolvidos no processo de recuperação da informação ao reduzir as diferenças conceituais. Portanto, uma ontologia corresponde a uma representação de um domínio a partir de seus conceitos abstratos e a forma como esses conceitos se relacionam entre si (ALMEIDA e BAX, 2003).

Gruber (1993) define ontologia como uma especificação explícita de uma conceitualização. No caso de sistemas baseados em conhecimento, podemos descrever a ontologia como um conjunto de termos que representam um domínio. O conjunto de objetos que são representados por esses termos formam o universo do discurso. Os objetos representados e os relacionamentos existentes entre eles são formalmente descritos através de um vocabulário representativo. Assim, através deste vocabulário, um programa de computador pode representar o conhecimento.

2.2.1 Conceitos básicos

As ontologias não apresentam sempre a mesma estrutura, mas existem características e componentes básicos comuns presentes em grande parte delas. Mesmo apresentando propriedades distintas, é possível identificar tipos bem definidos. Os componentes básicos de uma ontologia são: conceitos, relacionamentos, funções, axiomas e indivíduos.

Conceitos – Uma ontologia abrange um conjunto de conceitos e uma hierarquia entre esses conceitos, ou seja, uma taxonomia. Esses conceitos podem ser abstratos (ex.: força) ou concretos (ex.: carro), elementares (ex.: elétron) ou compostos (ex.: átomo),

reais ou fictícios. Podemos citar como exemplo de taxonomia: “o conceito homem ser um subconceito do conceito pessoa” (ZAMITH, 1998).

Relacionamentos – Uma ontologia apresenta um conjunto de relacionamentos entre conceitos. Podemos citar como exemplo de relacionamento entre os conceitos de pessoa e carro, o relacionamento ser-dono (ZAMITH, 1998).

Funções – Uma função é um caso especial de relacionamento em que um conjunto de elementos tem uma relação única com outro elemento. Um exemplo é “ser pais biológicos”, onde um conceito homem e um conceito mulher estão relacionados a um conceito filho (ZAMITH, 1998).

Axiomas – Axiomas modelam sentenças que são sempre verdadeiras. Estas são classificadas em estruturais e não estruturais (NOVELLO, 2002). Um exemplo de axioma é afirmar que toda pessoa tem mãe (ZAMITH, 1998).

Indivíduos – Indivíduos são objetos do mundo; que pertencem a classes e são relacionados a outros indivíduos (e classes) através de propriedades (NOVELLO, 2002).

2.2.2 Classificação de ontologias

As ontologias são classificadas por (GUARINO, 1997) sob duas dimensões: Nível de Detalhamento e Nível de Dependência (NOVELLO, 2002).

NÍVEL DE DETALHAMENTO - essa dimensão trata de ontologias extremamente ricas e detalhadas, que estão mais próximas da semântica esperada em um vocabulário, em oposição a outras mais simples, que tem uma dependência maior com a compreensão do usuário sobre o contexto ao qual se refere a ontologia (NOVELLO, 2002).

NÍVEL DE DEPENDÊNCIA - essa dimensão trata da dependência de uma tarefa em particular ou a um determinado ponto de vista. Ela distingue ontologias de alto nível, de domínio, de tarefa e de aplicação (NOVELLO, 2002).

Ontologias de Alto Nível (de topo) - descrevem conceitos gerais, como espaço, tempo, matéria, objeto, evento, ação, etc., que são independentes de um problema particular ou domínio (NOVELLO, 2002).

Ontologias de Domínio - descrevem um vocabulário relacionado a um domínio genérico, como medicina, direito, cadastro técnico, etc. (NOVELLO, 2002).

Ontologias de Tarefa - descrevem uma tarefa ou uma atividade genérica, como diagnóstico, vendas, etc., pela especialização de termos introduzidos na ontologia de alto nível (NOVELLO, 2002).

Ontologias de Aplicação - Descrevem conceitos que dependem tanto de um domínio específico como de uma tarefa específica, e que geralmente são uma especialização de ambos (NOVELLO, 2002).

2.2.3 Vantagens da utilização de ontologias

As ontologias tornam possível definir uma infraestrutura para integrar sistemas inteligentes ao nível conceitual do conhecimento (NOVELLO, 2002), nível este que é independente do nível de implementação. Sua utilização traz as seguintes vantagens (NOVELLO, 2002):

Colaboração: possibilita o compartilhamento do conhecimento entre os membros interdisciplinares de uma equipe;

Interoperação: facilita a integração da informação, especialmente em aplicações distribuídas;

Informação: pode ser usada como fonte de consulta e de referência do domínio;

Modelagem: pode ser representada por blocos estruturados que podem ser reusáveis na modelagem de sistemas no nível de conhecimento;

Reuso: permite que domínios de conhecimento sejam reutilizados.

2.3 GESTÃO DE CONHECIMENTO

A maneira como as organizações administram o conhecimento para alcançar melhores resultados é denominada de Gestão de Conhecimento. Segundo Davenport e Prusak (DAVENPORT e PRUSAK, 1998), seria inadequado referir-se à gestão de conhecimento como sendo única e exclusivamente compartilhamento de informações. O processo de gestão de conhecimento envolve a geração, a codificação e o compartilhamento do conhecimento na organização. A geração do conhecimento refere-se a todas as formas de criação do conhecimento, a partir da interação com o ambiente externo ou, até mesmo, por meio da interação entre os indivíduos da organização (DAVENPORT e PRUSAK, 1998).

Segundo Takeuchi e Nonaka (2008), uma organização cria e utiliza conhecimento convertendo o conhecimento tácito em conhecimento explícito, e vice-versa. O conhecimento explícito é o conhecimento transmitido através da linguagem formal e de forma sistemática. Pode ser armazenado e compartilhado através de manuais e livros. O conhecimento tácito é o conhecimento próprio de cada indivíduo, incluindo suas habilidades. Takeuchi e Nonaka identificaram quatro modos de conversão de conhecimento: (1) socialização: de tácito para tácito; (2) externalização: de tácito para explícito; (3) combinação: de explícito para explícito; e (4) internalização: de explícito para tácito. Este ciclo se tornou conhecido na literatura como modelo SECI e está no núcleo do processo de criação do conhecimento. Este modelo descreve como o conhecimento tácito e explícito são amplificados em termos de qualidade e quantidade, do indivíduo para o grupo e, então, para a organização.

O conhecimento é amplificado passando pelos quatro modos de conversão que envolve uma combinação diferente das entidades de criação do conhecimento, esse processo é listado a seguir (TAKEUCHI e NONAKA, 2008):

1. Socialização (indivíduo para indivíduo): Compartilhamento e criação do conhecimento tácito através da experiência direta.

2. Externalização (indivíduo para grupo): Articular conhecimento tácito através do diálogo e da reflexão;

3. Combinação (grupo para organização): Sistematizar e aplicar o conhecimento explícito e a informação;

4. Internalização (organização para indivíduo): Aprender e adquirir novo conhecimento tácito na prática.

O capital intelectual é um ativo muito importante de uma organização, devido, principalmente, à grande dificuldade de ser adquirido, criado, mantido, transmitido e aperfeiçoado. Sendo assim, em face à competitividade e à busca de melhores resultados, gerenciar o capital intelectual das organizações é um grande desafio, mas que pode propiciar um ótimo retorno.

2.4 SISTEMA MULTIAGENTE

Segundo Russel e Norvig (2010), um agente é uma entidade que pode perceber o seu ambiente por meio de sensores e age sobre ele por meio de atuadores. Agentes de software são programas automatizados que executam ações de forma autônoma e, se necessário, se comunicam com outros agentes a fim de atingir os seus objetivos.

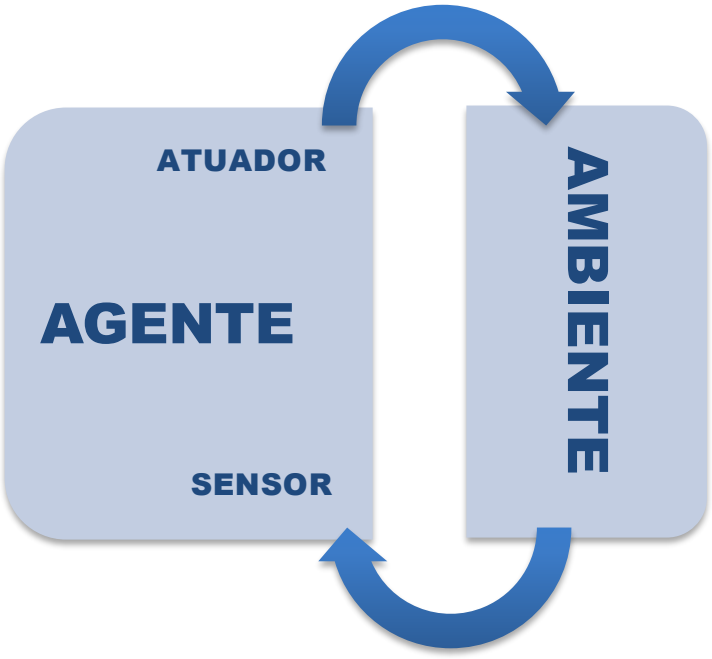


Figura 2. Modelo de Agente (CHAGAS, 2013)

Os sistemas computacionais podem utilizar um ou mais agentes de software para melhor atender às suas necessidades. Quando o sistema trabalha com um conjunto de agentes que interagem entre si, ele é denominado de Sistema Multiagente (SMA). Um SMA é formado por um conjunto de agentes que podem trabalhar de forma cooperativa ou competitiva. As principais características que diferenciam um agente de software de um

programa comum são sua autonomia e a capacidade de socialização para alcançar objetivos (RUSSELL e NORVIG, 2010).

Os problemas para os quais o agente é a solução são chamados de ambiente de tarefas. Para projetar um agente, o primeiro passo que devemos realizar é definir o ambiente de tarefas da melhor forma possível.

2.4.1 Tipos de Agentes

Os principais tipos de agentes citados na literatura são (RUSSELL e NORVIG, 2010):

a) Agentes reativos simples

É o tipo mais simples de agente. Esses agentes selecionam ações com base na percepção atual do ambiente, sem levar em conta percepções anteriores. Agentes reativos simples se caracterizam pela inteligência muito limitada. Geralmente eles só funcionam quando o ambiente é completamente observável.

b) Agentes reativos baseados em modelos

O modo mais efetivo de lidar com a possibilidade de observação parcial do ambiente é o agente controlar a parte do mundo que ele não pode ver agora. Para isso, o agente deve manter algum tipo de estado interno que dependa do histórico de percepções e assim reflita pelo menos alguns dos aspectos não observados do estado atual. Mas, para manter e atualizar essas informações internas, o agente precisa saber como o mundo evolui, como ele funciona, ou seja, precisa de um modelo. Por isso que esse tipo de agente é dito baseado em modelo.

c) Agentes baseados em objetivos

Da mesma forma que o agente precisa de uma descrição do estado atual, ele também precisa de alguma espécie de informação sobre objetivos que descrevam situações desejáveis. O programa de agente pode combinar isso com informações sobre os resultados de ações possíveis, a fim de escolher ações que alcancem o(s) objetivo(s). Às vezes, a seleção da ação

baseada em objetivos é direta, quando a satisfação do objetivo resulta de imediato de uma única ação. Outras vezes ela será mais complicada, quando o agente tiver de considerar longas sequências de ações até encontrar um meio de atingir o objetivo. Apesar de parecer menos eficiente, esse tipo de agente é mais flexível, porque o conhecimento que apoia suas decisões é representado de maneira explícita e pode ser modificado.

d) Agentes baseados na utilidade

Os objetivos não são suficientes para gerar um comportamento de alta qualidade na maioria dos ambientes. Os objetivos simplesmente permitem uma distinção binária entre "útil" e "inútil", enquanto uma medida de desempenho mais geral deve permitir uma comparação entre diferentes estados do mundo de acordo com o grau de utilidade. Uma especificação completa da função de utilidade permite decisões racionais nos casos de objetivos contraditórios (por exemplo, velocidade e segurança) e quando há vários objetivos incertos onde o agente pondera a probabilidade de sucesso em relação à importância dos objetivos.

2.4.2 Plataformas para desenvolvimento de agentes

LALO: *Langage d'Agents Logiciel Objet* - é um sistema de desenvolvimento de sistemas multiagente. Este sistema tem sido desenvolvido e mantido desde 1993 pelo CRIM (*Centre de Recherche Informatique de Montréal*). Os usuários do LALO estão principalmente em universidades e centros de pesquisa. O sistema LALO consiste de 83 (oitenta e três) classes que foram desenvolvidas manualmente, e 7 (sete) classes adicionais, que foram geradas automaticamente por um gerador de códigos. Estas 90 (noventa) classes em C++ têm um total de 40K de linhas de código-fonte. Em adição as 90 (noventa) classes específicas da aplicação, um número de classes de bibliotecas padrões de E/S (Entrada e Saída), *socket communication*, dentre outras, são usadas no sistema LALO. LALO foi desenvolvido em boa parte sobre o Windows NT usando o Visual C++, sendo então portado para o Sun OS e o Solaris (BRIAND, WÜST e LOUNIS, 2001).

PHANTOM: É uma linguagem interpretada projetada para aplicações distribuídas, interativas e em larga escala, tais como sistemas de conferência distribuída, jogos em rede e ferramentas

de trabalho colaborativo (BRAGA e PEREIRA, 2001). A Phantom é uma linguagem que suporta um número de considerado de características imperativas de programação, incluindo: interfaces, objetos, *threads*, coletor de lixo e exceções. Todas essas características são da linguagem de programação Modula-3 (NELSON, 1991). Onde possível, Phantom reutilizou sintaxe e semântica da Modula-3. Phantom estende o modelo de objeto da Modula-3, e omite muitas outras características complexas da Módula-3, que complicaria a semântica partilhada do Phantom. A linguagem também inclui suporte por declarações implícitas, listas definidas dinamicamente, e funções de propósito geral. Estas características são modeladas em suas partes correspondentes em outras linguagens interpretadas (COURTNEY, 1995).

JADE: *Java Agent Development Framework* - Framework de Desenvolvimento de Agentes em JAVA - é um software estruturado completamente implementado na linguagem JAVA. Ele simplifica a implementação de sistemas de multiagente através de um middleware que busca cumprir com as especificações da FIPA (*Foundation for Intelligent Physical Agents*) e através de uma série de ferramentas que suporta depuração. A plataforma de agente pode ser distribuída através de diferentes máquinas (que não precisam ter o mesmo sistema operacional) e a configuração pode ser controlada via uma API. A configuração pode ser modificada em tempo de execução movendo agentes de uma máquina para outra, como e quando for necessário. O único requisito de sistema é o Java Run Time version 1.4 ou mais recente. A arquitetura de comunicação oferece comunicação flexível e eficiente, onde JADE cria e gerencia a fila de espera das mensagens ACL, privadas para cada agente. Agentes podem acessar sua interface via variados modos de combinações: bloco, votação, *timeout*, entre outros (BELLIFEMINE, CAIRE e GREENWOOD, 2007).

2.4.3 FIPA - *The Foundation for Intelligent Physical Agents*

A FIPA é uma organização da *IEEE Computer Society* que promove a tecnologia baseada em agentes e a interoperabilidade de seus padrões com outras tecnologias. Foi criada em 1996 como uma associação internacional para desenvolver uma coleção de padrões relacionados com a tecnologia de agentes de software (FIPA, 2004).

As especificações chaves do padrão FIPA são:

- Arquitetura abstrata FIPA;
- Estrutura de mensagem FIPA-ACL;

- Atos de comunicação FIPA-ACL;
- Linguagem de conteúdo FIPA-SL;
- Protocolo de Interação por Requisição;
- Protocolo de Interação por Rede de Contrato.

A arquitetura abstrata especificada pela FIPA fornece um ponto de referência comum para implementações FIPA. O modelo de referência para gerenciamento de agentes definido pelo padrão FIPA pode ser observado na Figura 3.

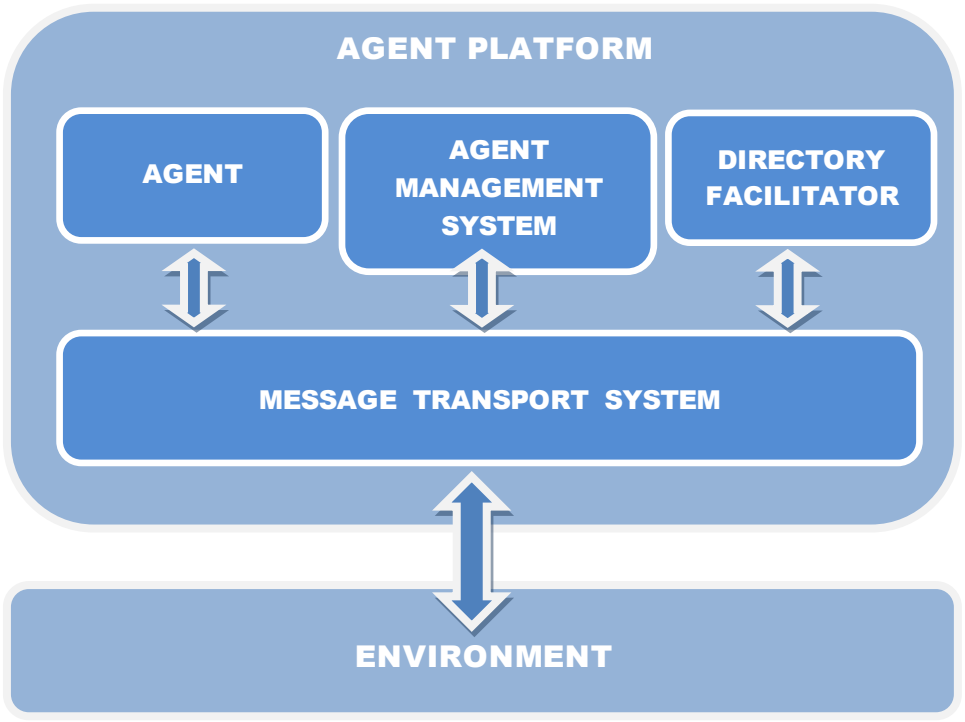


Figura 3. Modelo de Referência para Gerenciamento de Agentes (FIPA, 2004)

De acordo com a Figura 3, o modelo de referência FIPA é composto por (FIPA, 2004):

***Agent Platform* (AP):** Fornece a infraestrutura física na qual o agente é instalado. O AP consiste de máquina, sistema operacional, componentes de gerenciamento de agentes da FIPA, os próprios agentes e qualquer software de suporte adicional. O projeto interno específico de um AP é deixado para os desenvolvedores de um sistema de agente e não é assunto da padronização da FIPA. Em um único AP pode haver múltiplos computadores e os agentes não precisam estar no mesmo *host*.

***Agent*:** Um agente é um processo computacional que habita em uma AP e normalmente oferece um ou mais serviços computacionais que podem ser publicados como uma descrição de serviço. O projeto destes serviços não é foco da FIPA, que apenas indica a estrutura e codificação das mensagens usadas para troca de informações entre agentes. Um agente deve ter pelo menos um proprietário e deve suportar pelo menos uma noção de identidade que pode ser descrita usando o FIPA *Agent Identifier* (AID), que rotula um agente para distingui-lo sem ambiguidade. Um agente pode ser registrado em um número de endereço de transporte no qual pode ser contatado.

***Directory Facilitator* (DF):** O DF é um componente opcional de um AP fornecendo serviços de páginas amarelas para outros agentes. Ele mantém uma precisa, completa e atualizada lista de agentes e deve fornecer a informação mais atual sobre agentes em seu diretório, em uma base não discriminatória para todos os agentes autorizados.

***Agent Management System* (AMS):** AMS é um componente obrigatório de um AP e é responsável pelo gerenciamento do AP, tais como criação e remoção de agentes, além de monitoramento de migração de agentes de um AP. Cada agente deve se registrar com um AMS no intuito de obter um AID, que é então retido pelo AMS como um diretório de todos os agentes presentes dentro do AP e seu estado atual (ativo, suspenso, esperando).

***Message Transport System* (MTS):** MTS é um serviço fornecido por um AP para transportar mensagens FIPA-ACL entre agentes em qualquer AP dado e entre agentes de diferentes APs. Mensagens são fornecidas em um envelope de transporte que possui um conjunto de parâmetros de detalhamento, como, por exemplo, para quem a mensagem está sendo enviada.

2.5 GESTÃO DE PROJETOS

Gerência de projetos é a aplicação de conhecimentos, habilidades, ferramentas e técnicas às atividades do projeto a fim de alcançar os seus objetivos (ANSI e PMI, 2008).

Um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo (ANSI e PMI, 2008). Nesse sentido, entende-se que deve haver um início e um fim bem definidos, apesar de poderem ser alterados se necessário, e seu término indica que seus objetivos foram alcançados, ou chegou-se à conclusão que não seria possível ou viável alcançá-los. Sua duração depende da complexidade do projeto.

O termo “exclusivo” indica que cada projeto cria um produto, serviço ou resultado único (ANSI e PMI, 2008). Embora as partes de um projeto possam ser semelhantes, ou partes do produto, serviço ou resultado possam ter características iguais, isso não altera a sua singularidade. Por exemplo, um projeto de um software pode ter a mesma equipe, utilizar os mesmos recursos e, entretanto, produzir um sistema novo, completamente distinto.

O ciclo de vida do projeto consiste nas suas fases, que geralmente são sequenciais, mas que muitas vezes podem sobrepor-se (ANSI e PMI, 2008). Cada fase consiste em uma subdivisão do projeto, onde é necessário um controle adicional para gerenciar de forma efetiva o término de uma entrega importante.

A estrutura das fases permite que o projeto seja segmentado em subconjuntos para facilitar o gerenciamento, o planejamento e o controle. O número e a necessidade de fases, e o grau de controle aplicado, depende do tamanho, da complexidade e do impacto potencial do projeto.

As fases podem ainda relacionar-se de três formas diferentes (ANSI e PMI, 2008):

- Sequencial, onde uma fase só inicia após o término da anterior;
- Sobreposta, em que uma fase pode ter início antes do término da anterior.

A gestão de projetos consiste em um conjunto de ações aplicadas às atividades do projeto de modo a atender os seus requisitos, adaptando-se às diferentes necessidades, preocupações e expectativas das partes interessadas durante o andamento do projeto. Além disso, é necessário um balanceamento das restrições conflitantes do projeto, que envolvem: escopo, qualidade, cronograma, orçamento, recursos e riscos (ANSI e PMI, 2008).

O responsável pela gestão é o gerente de projetos. Para realizar seu trabalho, o gerente utiliza ferramentas e técnicas específicas. Mesmo possuindo habilidades e competências de gestão, é necessário que o gerente possua algumas outras competências básicas. As competências essenciais para o profissional gerente de projetos são: conhecimento sobre o domínio; bom desempenho quanto à aplicação do seu conhecimento; capacidade de orientar a equipe do projeto enquanto atinge o objetivo; e saber equilibrar as restrições do projeto (ANSI e PMI, 2008).

2.6 SISTEMAS DE GERENCIAMENTO DE PROJETOS

Sistemas de gerenciamento de projetos ou PMA (*Project Management Application*) (JORDAN, 2008) são aplicativos utilizados para facilitar o trabalho de gerenciamento de projetos através do suporte por computadores. Todos os aplicativos de gerenciamento de projetos devem conter determinados recursos básicos para que sejam úteis aos usuários. Um PMA normalmente tem estas características:

- Os projetos são criados em uma localização central, utilizando um processo padronizado. Todos os usuários envolvidos na criação e planejamento do projeto criam e armazenam informações sobre o projeto no mesmo lugar e sempre da mesma maneira;
- Informações sobre o progresso podem ser monitoradas utilizando gráficos e sistemas de alerta criados automaticamente. Notificações via e-mail e alertas de data/hora codificados pela cor mantêm todos os envolvidos em projeto facilmente informados do status das atribuições;
- Os projetos podem ser categorizados e as tarefas definidas de forma detalhada ou superficial. Um usuário pode criar um projeto sobre uma pequena tarefa que tem um curto período de execução, como uma pequena atualização de *site*, ou um grande projeto anual que exigirá recursos de vários departamentos, muitos arquivos e uma lista de atividades cuidadosamente planejada;
- Ferramentas de medição, como cronograma, alocação de recursos e calculadores de data/hora, ajudam os usuários a gerenciar os projetos. Essas ferramentas devem ser fáceis de utilizar e localizadas perto de onde são necessárias. Uma ferramenta de alocação de recursos deve estar acessível como uma aba ou janela quando uma nova tarefa é criada.

Há muitos benefícios em usar um aplicativo de gerenciamento de projeto. Partes interessadas (*stakeholders*) e usuários têm um local centralizado para informações de projetos. Sistemas bem projetados incluem algum tipo de sistema de troca de informações entre os membros da equipe de projeto, um local para armazenar os arquivos de projeto, controle de versão, status do projeto na forma de um gráfico ou diagrama e um local para discutir tópicos sobre o projeto. Motivar os membros da equipe de projetos para que eles adicionem novos

projetos com informações atualizadas é fundamental para um sistema ser útil (JORDAN, 2008).

A comunicação entre membros do projeto é parte crucial de qualquer sistema de gerenciamento de projetos. Se um sistema de gerenciamento de projeto é utilizado, os membros ainda podem trocar e-mails entre si, mas todas as informações fundamentais estão armazenadas em uma localização central. Outras formas de comunicação estão disponíveis a qualquer momento: fóruns, repositórios de arquivos ou registro em *log* de tarefas. Qualquer PMA realmente útil deve fornecer ferramentas de comunicação como estas (JORDAN, 2008).

3 TRABALHOS RELACIONADOS

A seguir são apresentados trabalhos que abordam o desafio da implantação de modelos de melhoria de processos discutido neste trabalho e como a solução proposta se diferencia das demais.

Em (ZHANG e SHAO, 2011), os autores propõem um *framework* para melhoria de processos de pequenas e médias organizações de software. O objetivo é auxiliar essas organizações nos estágios iniciais do seu desenvolvimento, focando na qualidade dos seus produtos. O *framework* é um guia que estrutura todo o processo de desenvolvimento do projeto de software, seguindo as recomendações das áreas de processos do CMMI para os níveis 2 e 3. O trabalho apresenta a importância e um referencial básico sobre o modelo CMMI, e propõe um *framework* como guia prático.

Em (GARCIA, PACHECO e CALVO-MANZANO, 2010), os autores propõem uma ferramenta web para auxiliar na melhoria de processos nas organizações de software implantando o CMMI. No trabalho foi realizada uma pesquisa sobre as características mais comuns em ferramentas de apoio à melhoria de processos. Os autores enfatizam que existem poucos trabalhos sobre desenvolvimento de ferramentas SPI (*Software Process Improvement*). A maioria das ferramentas explora o processo de avaliação baseando-se em padrões. A ferramenta apresentada propõe um guia para melhoria dos processos das organizações seguindo modelos como o CMMI. O sistema disponibiliza recursos para acompanhamento e monitoramento do processo de melhoria facilitando a compreensão e a integração com os processos da organização.

Kovacheva e Todorov (2011) propõem o uso da ferramenta baseada na web TRAC (TEAM, 2012) para integrar o modelo CMMI e as práticas ágeis. Segundo os autores, os fatores mais importantes para o sucesso da integração são: automação de processos; cumprimento da gestão; motivação da equipe; e documentação do processo. A ferramenta TRAC suporta todos esses critérios, aumentando a motivação da equipe, fornecendo estimativas mais precisas, documentação detalhada o suficiente, e baixos custos de mudanças.

Em (ALI e IBRAHIM, 2011), é apresentada uma ferramenta *desktop* para auto-avaliação de processos. O objetivo é auxiliar as pequenas e médias empresas a avaliarem seus processos com base nos modelos de melhoria de processos, como: CMMI; ISO/IEC 15504; ISO/IEC 90003; e ISO/IEC 12207. A ferramenta permite a avaliação de processos em nível de projetos, processos e programas. Além disso, a ferramenta disponibiliza recursos para geração de relatórios de avaliação e geração do plano de melhoria.

Em Homchuenchom *et aL*. (2011) , os autores apresentam a ferramenta SPIALS para auto-avaliação de processos em organizações de software. A aplicação implementa as recomendações do CMMI mapeando as recomendações para práticas do SCRUM (SCHWABER e BEEDLE, 2002). O SCRUM é uma metodologia ágil utilizada para desenvolvimento ágil de software. A ferramenta proposta realiza uma auto-avaliação dos processos de uma organização utilizando o método SCAMPI (TEAM, 2011). Este é o método padrão utilizado pelo SEI para avaliação de maturidade e capacidade de acordo com o CMMI.

Os trabalhos destacados discutem os problemas que ocorrem na implantação de modelos de melhoria de processos de software e como solução são propostos sistemas de apoio. Ferramentas que apoiem a melhoria dos processos podem contribuir de forma significativa para melhoria na qualidade dos produtos, principalmente, de pequenas e médias empresas de software. Essas ferramentas podem reduzir custos ao passo que promovem a qualidade dos produtos.

A ferramenta VPM, proposta pelo presente trabalho, se diferencia dos trabalhos apresentados por guiar o desenvolvimento de projetos seguindo as recomendações de melhoria de processos de forma proativa. Nesse sistema o gerente de projetos é orientado durante o desenvolvimento do projeto sobre quando, e quais atividades devem ser realizadas em cada etapa do projeto. A orientação segue as melhores práticas listadas no PMBOK como roteiro para atender os requisitos do CMMI no Nível 2 de maturidade. As ferramentas propostas nos trabalhos relacionados exploram principalmente a auto-avaliação de processos das organizações. O processo de auto-avaliação consiste na comparação entre os processos organizacionais efetivamente utilizados e as recomendações do CMMI. Assim, elas

funcionam como guias no caminho da implantação do modelo. Porém, soluções como estas criam passos adicionais para o gerente de projeto sem auxiliar diretamente nas tarefas de gerenciamento. A VPM disponibiliza recursos automatizados para gerenciar projetos de software, de forma proativa em tempo de execução, contemplando as práticas do CMMI.

4 VPM – *VIRTUAL PROJECT MANAGER*

Para apoiar a implantação do Nível 2 de maturidade do CMMI, foram identificadas as áreas de processos vinculadas a esse nível de maturidade. Na abordagem por estágios, para ser atingido um determinado nível, é necessário cumprir todas as metas genéricas de todas as áreas de processos do CMMI para aquele nível. A seguir são listadas as áreas de processos para o Nível 2 do CMMI:

- Gerenciamento de Configuração;
- Medição e Análise;
- Monitoramento e Controle do Projeto;
- Planejamento do Projeto;
- Garantia de Qualidade de Processos e Produtos;
- Gerenciamento de Requisitos;
- Gerenciamento de Acordo com Fornecedores.

A partir de uma análise mais detalhada de cada área de processo, foram identificadas quais áreas seriam apoiadas através da ferramenta. Com base nesse estudo, as áreas de processo selecionadas como foco principal do presente projeto foram: gerenciamento de requisitos; planejamento do projeto; e monitoramento e controle do projeto. As outras áreas de processos pertencentes ao Nível 2 e suas respectivas práticas envolvem procedimentos interpessoais, ou são áreas com difícil abstração para uma solução genérica. Procedimentos interpessoais são os que envolvem a interação entre os membros da equipe e as partes interessadas (*stakeholders*). As áreas de difícil abstração são áreas muito específicas, cujas soluções estão diretamente ligadas ao perfil do projeto. Essas características dificultariam ou limitariam a implementação de uma solução genérica automatizada, uma vez que a tentativa de automatizar tais processos poderia criar passos adicionais desnecessários. Por exemplo, a área de processo de Gerenciamento de Acordo com Fornecedores envolve contato com

fornecedores, definição de acordos, revisão de documentos e outras tarefas manuais. Uma solução para apoiar esta área de processos poderia ser implementada, mas estaria sujeita a limitações de áreas de aplicação.

Através da análise dos trabalhos relacionados, foi identificado que a maioria das ferramentas trabalha na auto-avaliação dos processos organizacionais de forma a obter um melhor alinhamento dos processos às recomendações. O processo de auto-avaliação envolve a revisão dos processos organizacionais. Durante esta revisão são identificadas quais alterações são necessárias para que os processos atendam as recomendações. Esse processo e a implantação do CMMI de forma geral implicam na realização de passos adicionais que aumentam a carga de trabalho do gerente de projetos. A ferramenta apresentada neste trabalho propõe um guia e a automação de algumas tarefas de gerenciamento seguindo as recomendações do CMMI. A ideia é reduzir a carga de trabalho do gerente de projetos e da equipe durante e após a implantação do CMMI.

Para a automação das tarefas são utilizadas as tecnologias de agentes de software e ontologias. Os agentes auxiliam de forma proativa no monitoramento das informações envolvidas nos processos realizados durante o desenvolvimento do projeto. Enquanto as ontologias são utilizadas como bases de conhecimento flexíveis e como entrada para mecanismos de inferência. Para cada área de processo selecionada foram atribuídos agentes, assim, os principais agentes do sistema são: Assistente de Requisitos, Assistente de Plano, e Assistente de Monitoramento e Controle. Além desses, temos um Assistente de Interface, responsável pela comunicação com o gerente de projetos e com a equipe. A partir das áreas de processo selecionadas, foram identificadas metas, práticas e subpráticas de cada área de processo, e cada subprática foi associada a um objetivo de um agente.

Na Tabela 1 são apresentadas as práticas e subpráticas contempladas por cada agente da ferramenta.

Tabela 1. Práticas e Subpráticas apoiadas pela VPM

Agente	Prática	Subprática
Assistente de Requisitos	Gerenciar Requisitos	Entender requisitos; Gerenciar mudanças de requisitos; Manter a rastreabilidade bidirecional dos requisitos; Garantir o alinhamento entre artefatos[1] e requisitos.
Assistente de Plano	Estabelecer Estimativas.	Estimar esforço e custo.

[1] Artefato no contexto deste trabalho é o produto de uma ou mais atividades desenvolvidas durante a execução de um projeto de software.

	Desenvolver o Plano do Projeto.	Identificar riscos do projeto; Planejar recursos do projeto; Planejar as necessidades de conhecimento e habilidades.
Assistente de Monitoramento e Controle	Monitorar o Projeto em Relação ao Plano	Monitorar riscos do projeto.
Assistente de Interface	Monitorar o Projeto em Relação ao Plano	Conduzir revisões de progresso.

4.1 ARQUITETURA DA APLICAÇÃO

Para evitar o retrabalho de implementar funções já existentes de PMA's *(Project Management Application*), foi selecionada uma ferramenta de PMA, *open-source* e modular, chamada dotProject (JORDAN, 2008). Assim, foi criado um módulo para o dotProject que possui acesso a todas as informações dos projetos cadastrados e dispõe das funcionalidades do dotProject no que diz respeito às funções gerais de gestão. Enquanto as informações são gerenciadas através da ferramenta PMA, o SMA monitora o banco de dados do dotProject e as ações do gerente.

A arquitetura geral da aplicação é apresentada na Figura 4. O gerente de projeto interage com a ferramenta através da interface da aplicação web de gestão de projetos, que, por sua vez, utiliza um banco de dados. Os agentes interagem com o dotProject e monitoram o banco utilizado pela ferramenta, são eles: Assistente de Plano (AP), Assistente de Monitoramento e Controle (AMC), Assistente de Requisitos (AR) e Assistente de Interface (AI), que interage diretamente com o gerente de projetos e a equipe através da interface da ferramenta ou por e-mail. Os agentes AP e AR possuem bases de conhecimento representadas em ontologias, onde cada uma delas está diretamente relacionada com os objetivos do agente. O ambiente possui três ontologias: ontologia de riscos (OR), ontologia de atividades (OA) e ontologia de recursos humanos (ORH). Além dos agentes já citados, a arquitetura também apresenta o agente DF (*Directory Facilitator*), que é o agente responsável por disponibilizar publicamente os serviços dos demais agentes. Esse mecanismo permite que os agentes se comuniquem. Esse tipo de abordagem segue as especificações do padrão FIPA (*Foundation for Intelligent Physical Agents*) (FIPA, 2004).

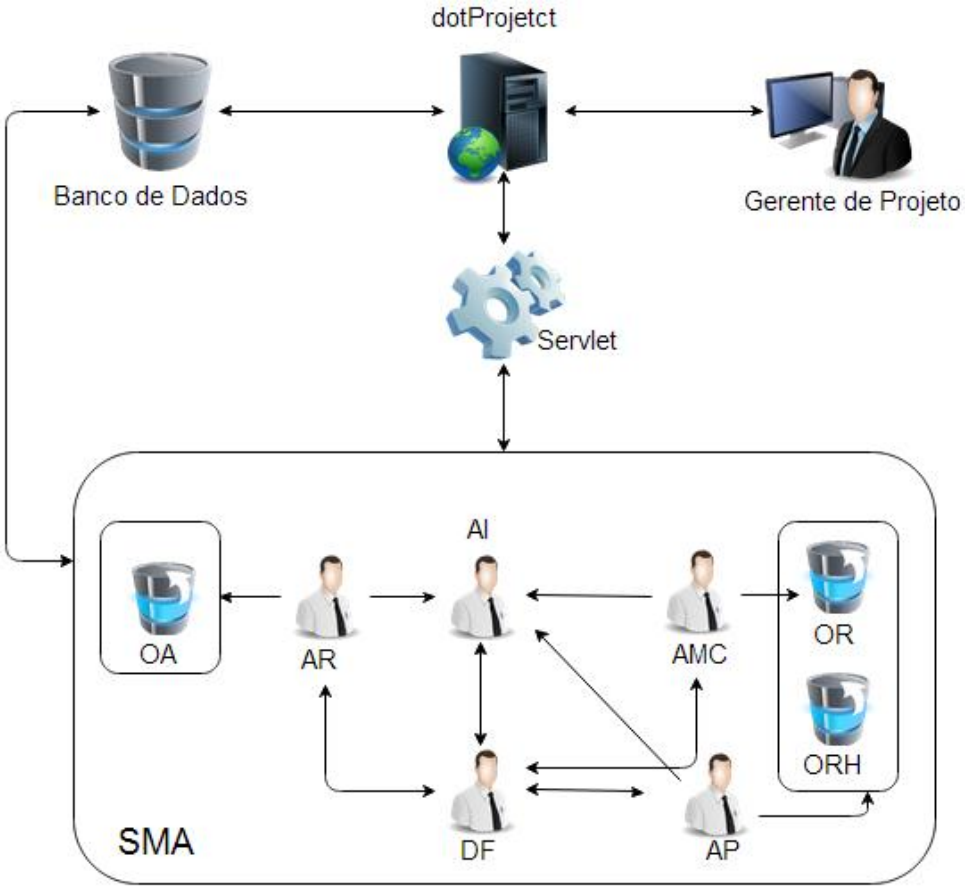


Figura 4. Arquitetura do Sistema VPM (CHAGAS, 2013)

4.2 MODELAGEM DOS AGENTES

Seguindo a arquitetura apresentada na seção anterior, foram modelados os agentes utilizando a metodologia MAS-CommonKADS+ (OLIVEIRA, 2010), selecionada, principalmente, por: (i) permitir a modelagem detalhada do comportamento dos agentes; (ii) ser baseada em UML (RUMBAUGH, JACOBSON e BOOCH, 2004), o que facilita a compreensão do modelo por todos que conhecem esta linguagem; e (iii) apresentar ferramenta de suporte à modelagem.

4.2.1 Modelo de Tarefas

Inicialmente foi realizado o levantamento dos requisitos. Após a fase de requisitos, o passo seguinte foi a definição do modelo de tarefas, através de diagramas referentes às tarefas do sistema. Nele são descritas as funcionalidades que o sistema deve contemplar. O modelo de tarefas do sistema pode ser observado na Figura 5. Todas as atividades do sistema são apresentadas no diagrama composto de suas respectivas subtarefas, que deverão ser realizadas para completar as funcionalidades.

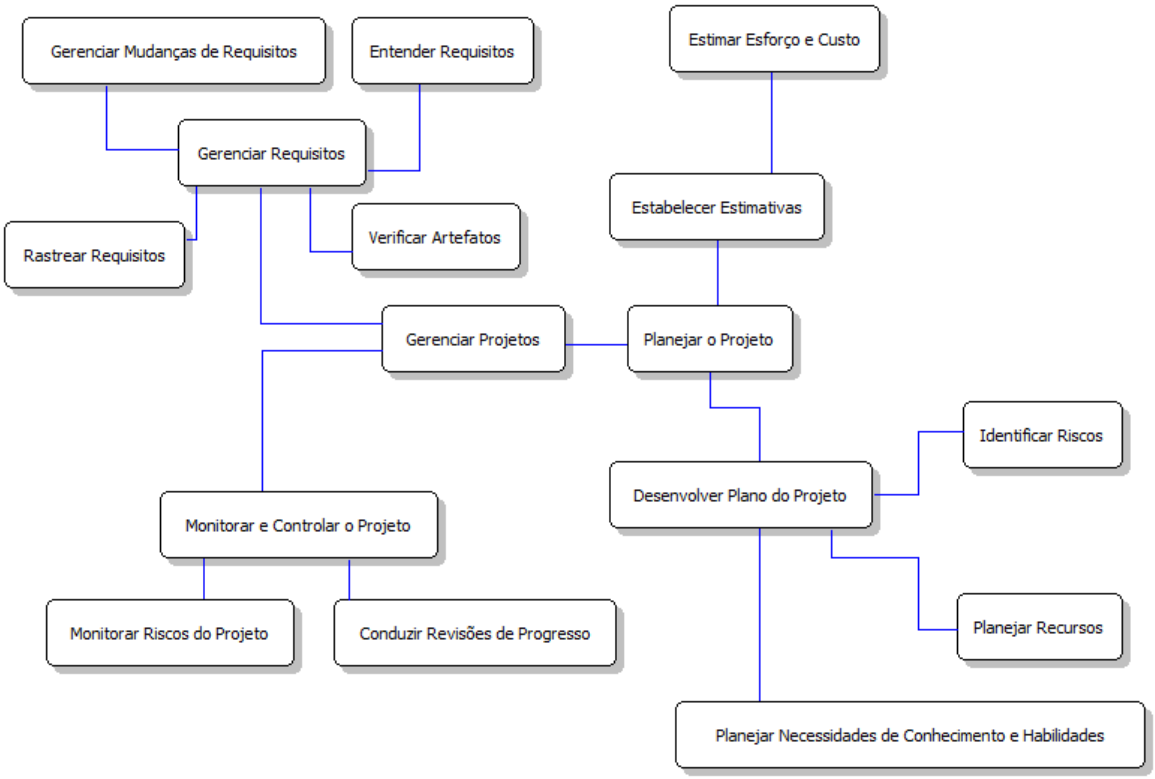


Figura 5. Modelo de Tarefas (CHAGAS, 2013)

A principal função do sistema está relacionada com a tarefa de "Gerenciar Projetos" situada no centro do modelo (Figura 5). O VPM contempla recursos para realizar as tarefas de "Planejar o Projeto", "Gerenciar Requisitos" e "Monitorar e Controlar o Projeto" que são subtarefas da tarefa principal "Gerenciar Projetos". De acordo com a Figura 5, cada subtarefa possui suas respectivas subtarefas, que estão diretamente ligadas aos objetivos de cada agente. A tarefa "Planejar o Projeto" possui as subtarefas "Estabelecer Estimativas" e "Desenvolver

Plano do Projeto". A tarefa "Desenvolver Plano do Projeto" possui as subtarefas "Identificar Riscos", "Planejar Recursos" e "Planejar Necessidades de Conhecimento e Habilidades". A tarefa "Estabelecer Estimativas" possui a subtarefa "Estimar Esforço e Custo". Enquanto a tarefa "Monitorar e Controlar o Projeto", subtarefa de "Gerenciar Projetos", está relacionada com as tarefas de "Monitorar Riscos do Projeto" e "Conduzir Revisões de Progresso". As tarefas relacionadas aos requisitos dos projetos são contempladas na subtarefa "Gerenciar Requisitos", subtarefa de "Gerenciar Projetos". Esta tarefa possui as subtarefas "Entender Requisitos", "Verificar Artefatos", "Rastrear Requisitos" e "Gerenciar Mudanças de Requisitos".

4.2.2 Modelo de Recursos e Objetos

O próximo passo da modelagem é a definição dos objetos do sistema identificados a partir dos requisitos do sistema e do modelo de tarefas. Para este sistema foram incluídas as tarefas que necessitam de autonomia e os recursos, como bancos de dados e ontologias. O Modelo de Recursos e Objetos é apresentado na Figura 6. Este modelo foi desenvolvido com o objetivo de fornecer uma melhor definição do sistema, sem focar apenas nos agentes.

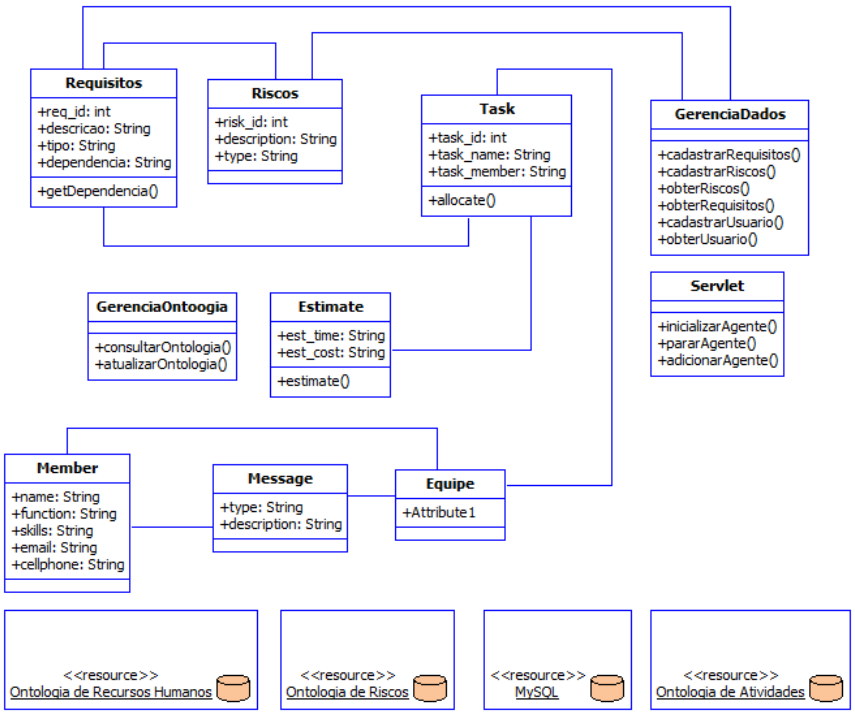


Figura 6. Modelo de Recursos e Objetos (CHAGAS, 2013)

No modelo são ilustradas as classes de controle: GerenciaDados, Estimar, GerenciaOntologia e Servlet. Além das classes de controle são representadas as classes de entidade: Member, Message, Equipe, Riscos e Requisitos. Outro elemento importante representado no modelo são os recursos utilizados pelo sistema: as bases de conhecimento e o banco de dados. As bases de conhecimento são: Ontologia de Recursos Humanos, Ontologia de Atividades e Ontologia de Riscos. O banco de dados utilizado é o MySQL.

4.2.3 Modelo de Papéis

Após a definição do Modelo de Recursos e Objetos, foi definido o Modelo de Papéis. Este modelo identifica quais são os papéis que cada agente pode desempenhar no sistema. Esses papéis são responsáveis por cumprir as tarefas existentes no modelo de tarefas. A Figura 7 apresenta o modelo de papéis definido para o sistema e as tarefas que eles realizam.

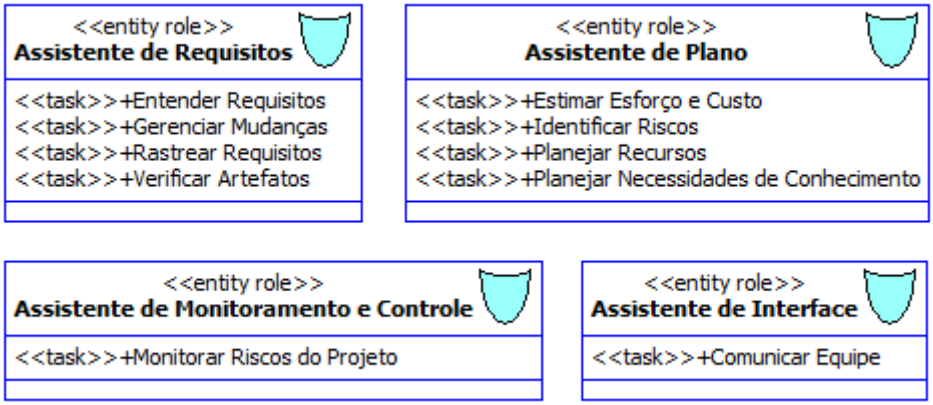


Figura 7. Modelo de Papéis (CHAGAS, 2013)

Seguindo o modelo o agente AR possui o papel responsável por realizar as tarefas "Entender Requisitos", "Gerenciar Mudanças", "Rastrear Requisitos" e "Verificar Artefatos". O papel do agente AP é responsável por realizar as tarefas "Estimar Esforço e Custo", "Identificar Riscos", "Planejar Recursos", "Planejar Necessidades de Conhecimento". O agente AMC é responsável apenas por "Monitorar Riscos do Projeto". O Assistente de Interface é responsável por "Comunicar Equipe".

4.2.4 Modelo de Organização

Seguindo os passos da metodologia, após a modelagem dos papéis do sistema, foi desenvolvido um diagrama que representa a organização do SMA. Neste diagrama estão contidas as hierarquias entre os papéis e os grupos de papéis. O modelo é dividido entre organização interna e externa. Na organização interna são apresentados os relacionamentos entre os papéis. Na externa, pode ser demonstrado o relacionamento entre grupos de agentes. O Modelo de Organização dos agentes pode ser observado na

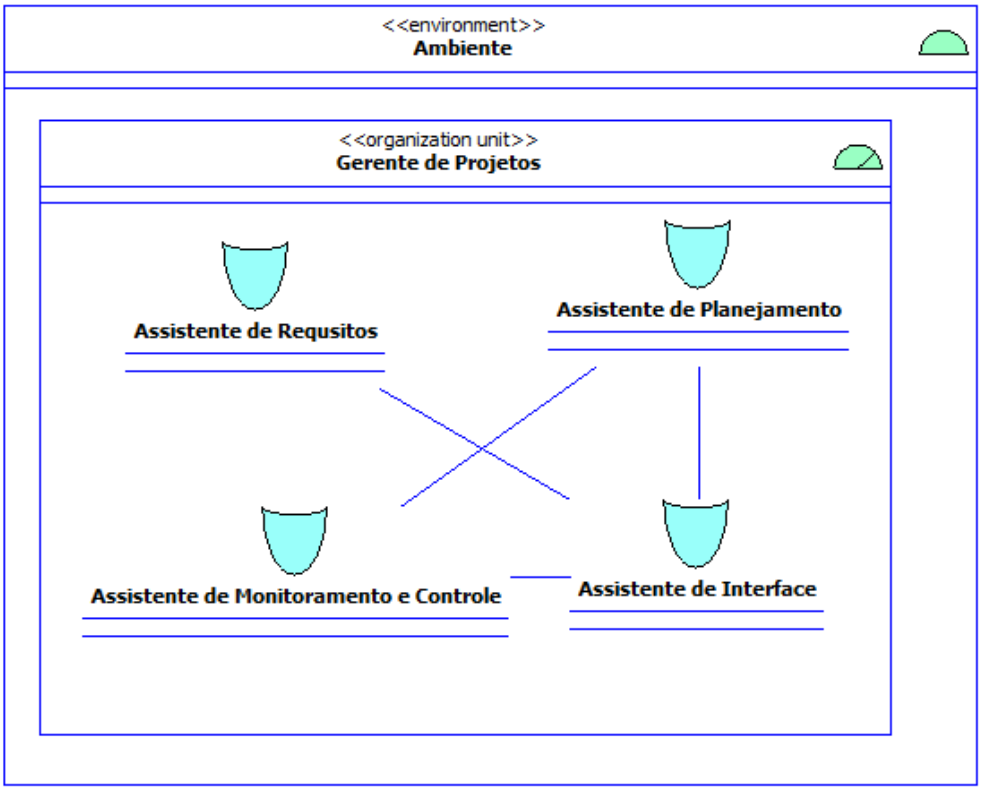


Figura 88. Este modelo representa o relacionamento entre os agentes e seu ambiente. O SMA é uma unidade organizacional dentro do ambiente de suporte ao gerenciamento do projeto de software.

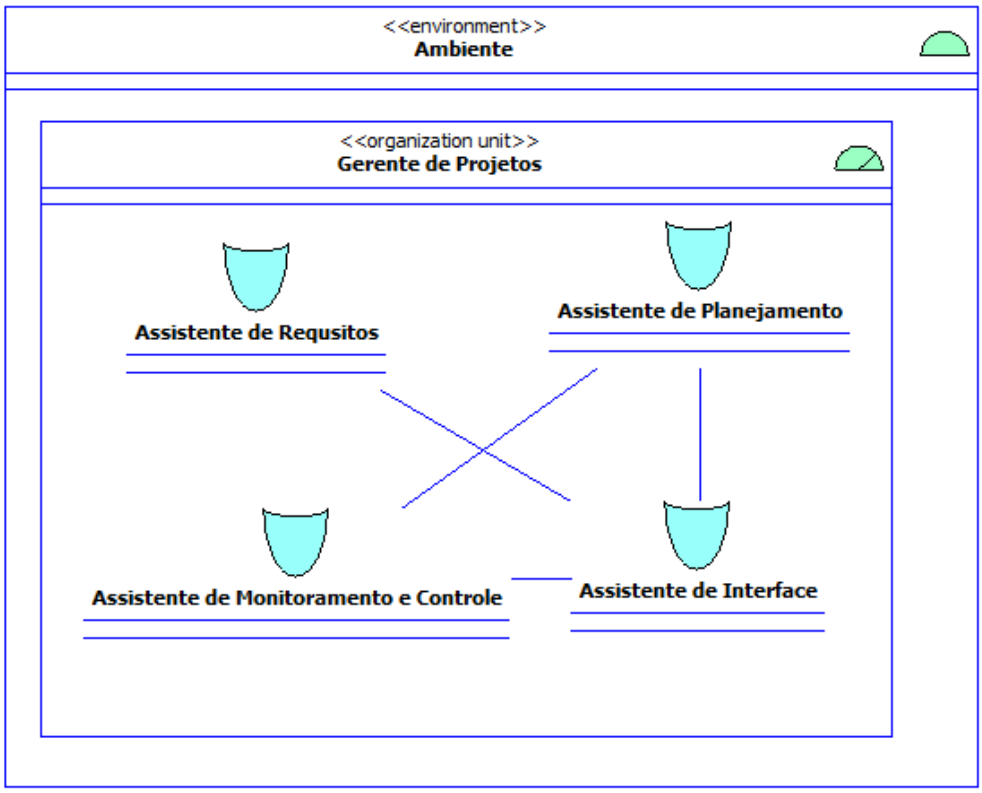
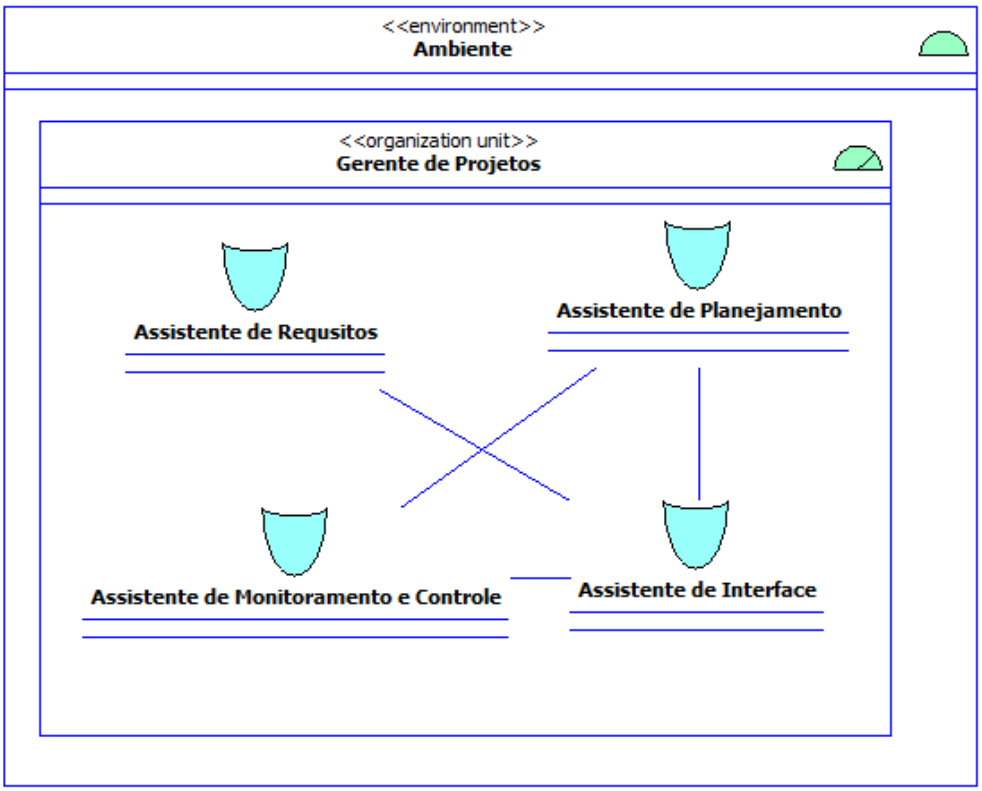


Figura 8. Modelo de Organização (CHAGAS, 2013)



O modelo na

Figura 88 apresenta o SMA como uma unidade da organização dentro do ambiente de gerenciamento de projetos. Nesta unidade estão representados os agentes por seus respectivos papéis e seus relacionamentos. O papel "Assistente de Monitoramento e Controle" se

relaciona com o "Assistente de Planejamento". Os papéis "Assistente de Requisitos", "Assistente de Monitoramento e Controle" e "Assistente de Planejamento" se relacionam com o "Assistente de Interface".

4.2.5 Modelo de Agentes

A partir dos modelos anteriores e dos requisitos levantados para o sistema, foram desenvolvidos os modelos dos agentes do SMA. Este modelo é responsável por definir: (i) quais agentes serão responsáveis por cada papel; (ii) a arquitetura dos agentes; e (iii) seus objetivos e características, tais como: entrada de dados, condições de ativação do agente e tipos de informação disponível. Antes de apresentar os modelos de agente, serão listados a seguir os *templates* textuais com objetivos e características de cada agente.

AGENTE ASSISTENTE DE REQUISITOS - AR

Descrição: Auxilia o gerente de projetos no gerenciamento de requisitos.

Objetivo 1 – Alertar sobre verificação de requisitos

Parâmetros de entrada: Critérios para avaliação de requisitos.
Condição de ativação: Inclusão ou alteração de requisitos do projeto.
Condição de finalização: Finalização do projeto.
Descrição: O agente AR monitora alterações nos requisitos do projeto e notifica o gerente de projetos informando que deve ser realizada uma verificação seguindo os critérios de avaliação e aceitação de requisitos. A notificação é realizada e exibida na tela com a lista de critérios.
Informação associada: Os critérios para avalição de requisitos fazem parte do processo de gerenciamento de escopo definido no PMBOK. A gestão de escopo está relacionada à área de processos de gestão de requisitos do CMMI. Esta área de processos faz parte dos requisitos para o Nível 2 de maturidade.

Objetivo 2 – Gerenciar mudanças de requisitos

Parâmetros de entrada: Requisitos do projeto.

Condição de ativação: Inclusão ou alteração de requisitos do projeto.

Condição de finalização: Finalização do projeto.

Descrição: O agente AR monitora as atividades relacionadas aos requisitos (inclusão, edição e exclusão) e faz o registro em um arquivo de *log* de todas as atividades. Cada atividade realizada é informada a todos os membros da equipe através do agente Assistente de Interface.

Informação associada: A gestão de modificações está relacionada à área de processos de gestão de requisitos do CMMI. Esta área de processos faz parte dos requisitos para o Nível 2 de maturidade.

Objetivo 3 – Manter a rastreabilidade de requisitos

Parâmetros de entrada: Requisitos do projeto.

Condição de ativação: Inclusão ou alteração de requisitos do projeto.

Condição de finalização: Finalização do projeto.

Descrição: O agente AR monitora a inclusão ou alteração de requisitos. Caso seja identificado um novo requisito ou um requisito modificado é gerada a matriz de rastreabilidade de acordo com suas dependências. A matriz fica disponível ao gerente de projetos e a qualquer membro da equipe. Se durante o projeto algum requisito for excluído, o agente AR informa se esse requisito é dependente ou não, através do agente Assistente de Interface.

Informação associada: A rastreabilidade de requisitos especifica as dependências entre requisitos, o que define a ordem na qual eles serão contemplados. Além disso, a rastreabilidade faz parte do processo de gerenciamento de escopo definido no PMBOK. A gestão de escopo está relacionada à gestão de requisitos. Esta área de processos faz parte dos requisitos para o Nível 2 de maturidade do CMMI.

Objetivo 4 – Garantir alinhamento entre artefatos e requisitos

Parâmetros de entrada: Novo artefato.

Condição de ativação: Acompanhamento de um novo projeto.

Condição de finalização: Finalização do projeto.

Descrição: Quando um novo projeto passa a ser acompanhado pelo SMA, o agente AR atribui atividades ao projeto de acordo com o PMBOK. Cada atividade possui artefatos de entrada e artefatos de saída. O agente AR monitora os artefatos associados ao projeto. Ao

passo que os artefatos de entrada para uma determinada atividade vão sendo atendidos, esta atividade é liberada. Quando for liberada uma nova atividade, o agente AR envia uma mensagem para o agente Assistente de Interface, que comunica o gerente de projetos e os membros da equipe.

Informação associada: Este objetivo está relacionado com a área de gerenciamento da integração do projeto definida pelo PMBOK. Esta atividade auxilia nas práticas do CMMI na área de processo de monitoramento e controle.

A Figura 9 ilustra o modelo do agente AR. Neste modelo estão representados os seus comportamentos e seus mecanismos internos. O quadro **Sensor** indica os mecanismos que são utilizados para o monitoramento do ambiente, são eles: *focoSecaoAdRequisitos(), alteracaoRequisitos(), novoArtefato()*. O quadro **Atuador** indica os mecanismos que são acionados quando o agente interage com o ambiente, são eles: *notificarGerente, exibirCriterios(), atualizarAtividades()*. Para a habilidade **Analisar Requisitos** o agente utiliza os mecanismos *registrarAlteracao(), atualizarMatriz()*. A habilidade **Analisar Artefatos** utiliza o mecanismo *analisarArtefatos()*. Esta habilidade envolve o uso da base de conhecimento sobre as atividades do projeto que está representada na Ontologia de Atividades.

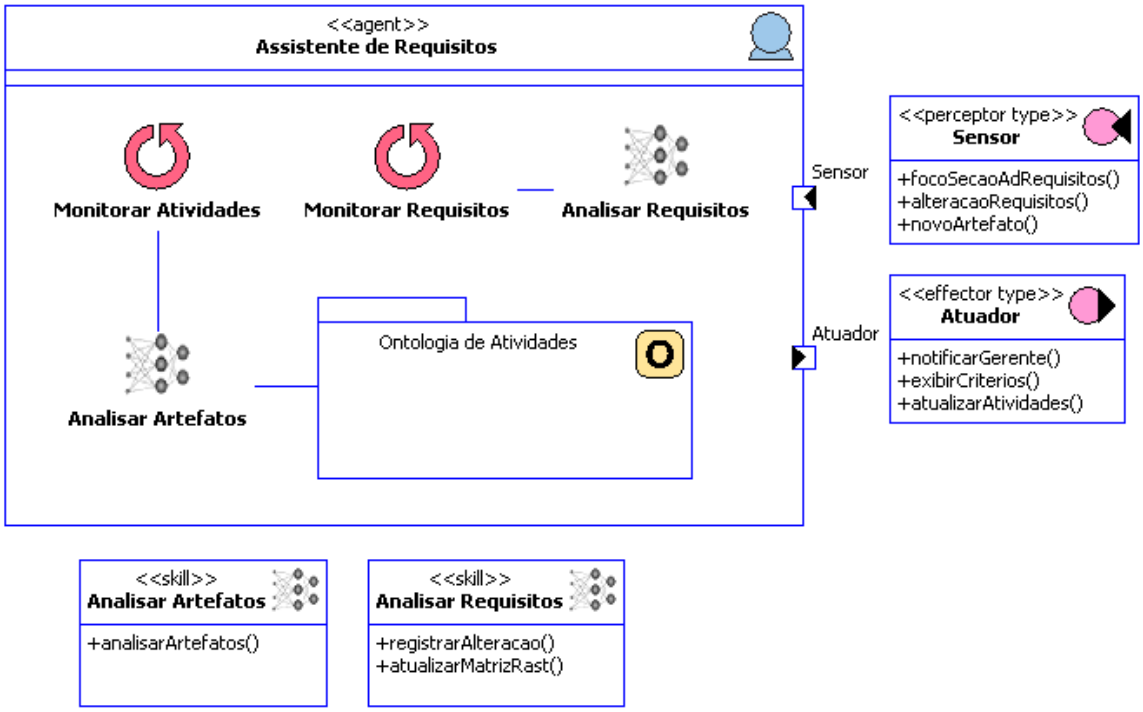


Figura 9. Modelo do Agente Assistente de Requisitos (CHAGAS, 2013)

O agente AR possui dois comportamentos: Monitorar Requisitos e Monitorar Atividades. O comportamento de Monitorar Requisitos tem o objetivo de: (i) auxiliar no entendimento dos requisitos; (ii) registrar alterações dos requisitos; e (iii) manter a rastreabilidade dos requisitos. O comportamento de Monitorar Atividades tem o objetivo de fazer o correto sequenciamento de atividades com base nos artefatos necessários para iniciá-las. Assim, só serão executadas atividades cujas necessidades de artefatos sejam completadas. Essas necessidades e a dependência das atividades estão especificadas na Ontologia de Atividades utilizada pelo agente.

AGENTE ASSISTENTE DE PLANO - AP

Descrição: Auxilia o gerente de projeto no plano do projeto.

Objetivo 1 – Estabelecer estimativas

Parâmetros de entrada: Especificações métricas do projeto.
Condição de ativação: O início de um novo projeto ou marco alcançado no projeto.
Condição de finalização: Encerramento do projeto.
Descrição: No momento em que um novo projeto é iniciado, o agente AP requisita os dados para estimativa inicial do projeto. Durante o desenvolvimento do projeto, o agente AP requisita periodicamente dados atualizados para uma nova estimativa.
Informação associada: As estimativas do projeto estão relacionadas com a área de gerenciamento de tempo e custo do projeto. Estas áreas de conhecimento estão relacionadas com a área de processo de planejamento do projeto especificada no CMMI.

Objetivo 2 - Analisar riscos do projeto

Parâmetro de entrada: Dados iniciais do projeto fornecidos pelo gerente de projeto.
Condição de ativação: Início de um novo projeto.
Condição de finalização: Após ser gerada a tabela dos possíveis riscos.
Descrição: Ao ser iniciado um novo projeto, o agente AP apresenta um questionário que deve ser preenchido pelo gerente de projetos. De acordo com as respostas, o agente AP consulta a Ontologia de Riscos – (OR) e identifica os possíveis riscos para o projeto. O resultado é armazenado em uma tabela que fica disponível aos membros da equipe.

Informação associada: A análise de riscos está vinculada à área de gerenciamento de riscos especificada no PMBOK. Esta atividade auxilia nas práticas especificadas no modelo CMMI para área de processos de monitoramento e controle do projeto.

Objetivo 3 - Alocar recursos humanos

Parâmetros de entrada: Atividades.

Condição de ativação: Tentativa de alocação.

Condição de finalização: Finalização do projeto.

Descrição: O agente AP sugere, em ordem maior compatibilidade e disponibilidade, dos membros da equipe e as necessidades das atividades. Para isso, o agente AP consulta a Ontologia de Recursos Humanos (ORH), onde está armazenada a árvore de conhecimento da organização. Ele faz um mapeamento entre o conhecimento necessário para a atividade e o conhecimento disponível entre os profissionais.

Informação associada: A alocação de recursos humanos está relacionada com a área de gerenciamento de recursos humanos definida no PMBOK. No CMMI, a alocação de recursos humanos está associada com o planejamento do projeto.

A

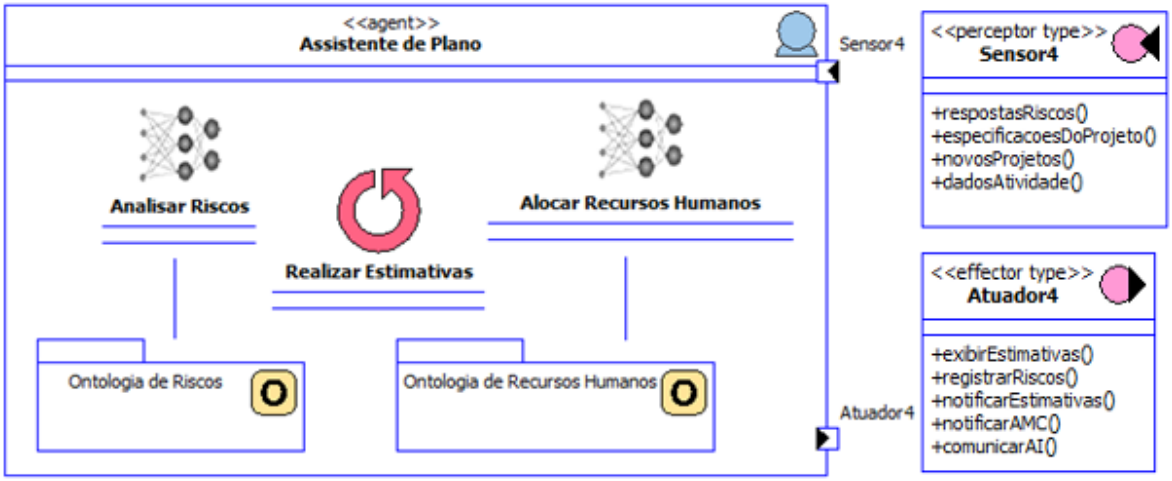


Figura 10 apresenta o modelo do agente AP. Seu comportamento **Realizar Estimativas** e seus mecanismos. O quadro **Sensor4** representa o conjunto de mecanismos que o agente utiliza para perceber as mudanças no ambiente, são eles: *respostasRiscos(), especificacoesDoProjeto(), novosProjetos(), dadosAtividade()*. O quadro **Atuador4** apresenta o conjunto de mecanismos relacionados com as ações do agente, são eles: *exibirEstimativas(), registrarRiscos(), notificarEstimativas(), notificarAMC(), comunicarAI()*. As habilidades do

agente são **Analisar Riscos** que envolve o conhecimento representado na Ontologia de Riscos, e **Alocar Recursos Humanos** que está associada à Ontologia de Recursos Humanos.

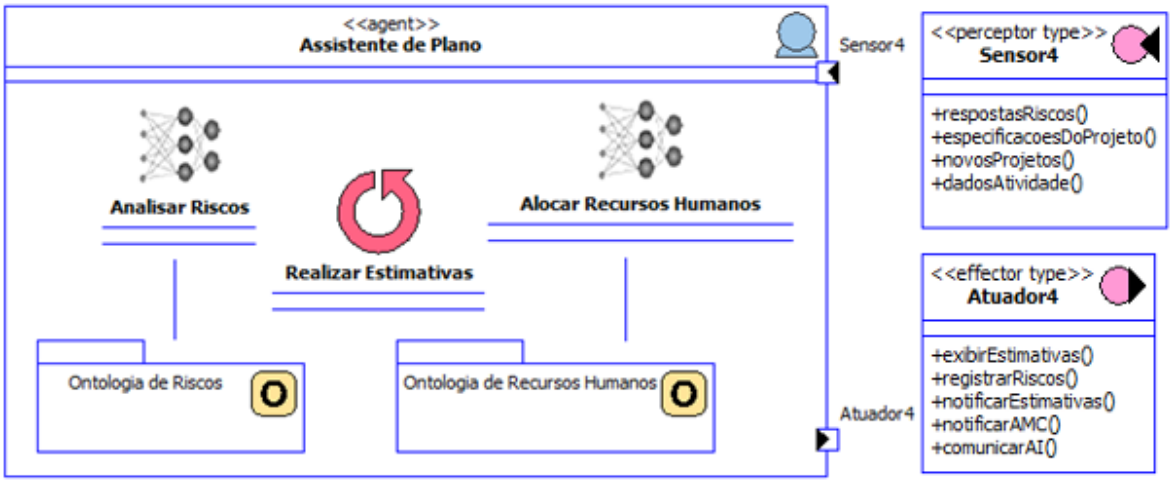


Figura 10. Modelo do Agente Assistente de Plano (CHAGAS, 2013)

O agente Assistente de Plano (AP) possui três comportamentos: Analisar Riscos, Alocar Recursos Humanos e Realizar Estimativas. O comportamento Analisar Riscos aciona um questionário aplicado ao gerente de projetos quando é iniciado um novo projeto. Após identificar os potenciais riscos comuns a projetos de software, o agente AP notifica os possíveis riscos ao agente Assistente de Monitoramento e Controle (AMC) que passa a monitorar esses riscos. Para atingir o objetivo de estabelecer estimativas do projeto, o agente AP inicia uma análise de pontos por função em cada marco do projeto. A Alocação de Recursos Humanos é um comportamento que busca aperfeiçoar a alocação de recursos humanos cruzando as habilidades pessoais dos membros da equipe e as competências necessárias apara realização das tarefas. Este agente possui acesso a duas bases de conhecimento: a Ontologia de Riscos (OR) e a Ontologia de Recursos Humanos (ORH).

AGENTEASSISTENTE DE MONITORAMENTO E CONTROLE - AMC

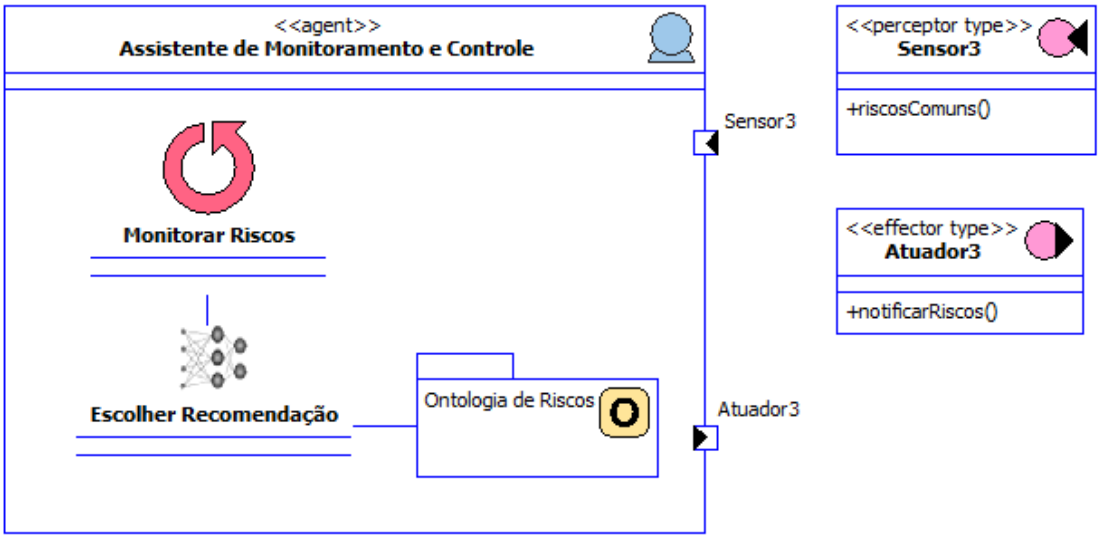
Descrição: Auxilia no monitoramento dos riscos do projeto, comunicando os possíveis riscos ao gerente de projetos durante sua execução.

Objetivo 1 - Monitorar Riscos do Projeto

Parâmetro de entrada: Dados iniciais sobre perfil do projeto.
Condição de ativação: Ações relacionadas aos riscos mais frequentes em projetos.
Condição de finalização: Finalização do projeto.

Descrição: Durante o desenvolvimento do projeto, o agente AMC alerta o gerente de projetos sobre os possíveis riscos identificados pelo agente Assistente de Plano (AP). As notificações são exibidas na interface durante uma ação associada a um possível risco. Ex.: Alocação de recursos humanos.

Informação associada: O monitoramento de riscos está associado a área de gerenciamento de riscos especificada no PMBOK. Esta atividade auxilia nas práticas especificadas no modelo CMMI para área de processos de monitoramento e controle do projeto.



A

Figura 11 apresenta o modelo de agente para o AMC. Ele apresenta o comportamento **Monitorar Riscos** e seus mecanismos internos. O quadro **Sensor3** possui o mecanismo relacionados à forma como o agente percebe o ambiente: *riscosComuns()*. O **Atuador3** representa o mecanismo utilizado pelo agente para interagir com o ambiente, *notificarRiscos()*. O comportamento **Monitorar Riscos** está associado à habilidade **Escolher Recomendação** que faz uso da base de conhecimento representada na *Ontologia de Riscos*.

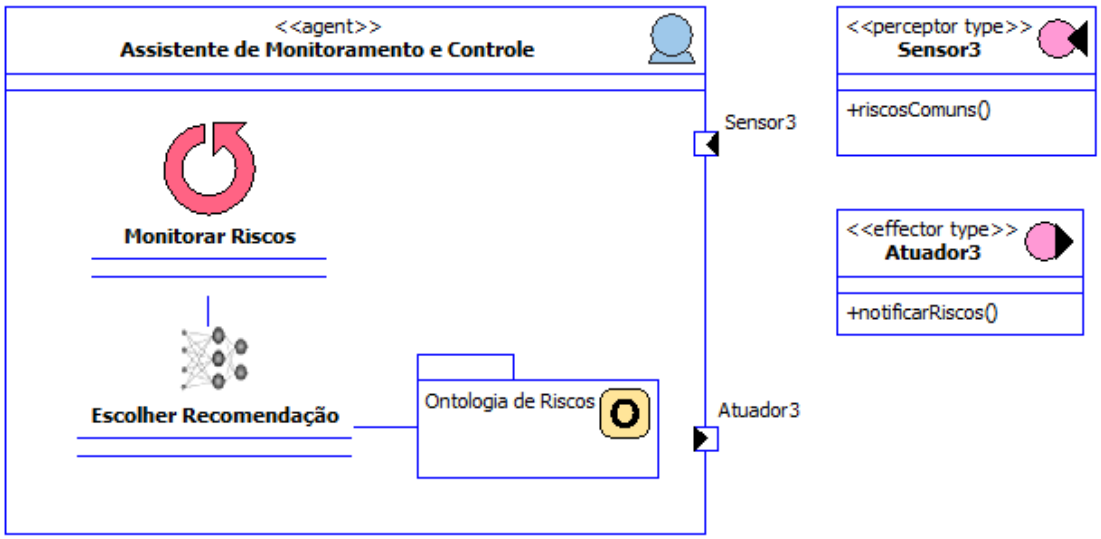


Figura 11. Modelo do Agente Assistente de Monitoramento e Controle (CHAGAS, 2013)

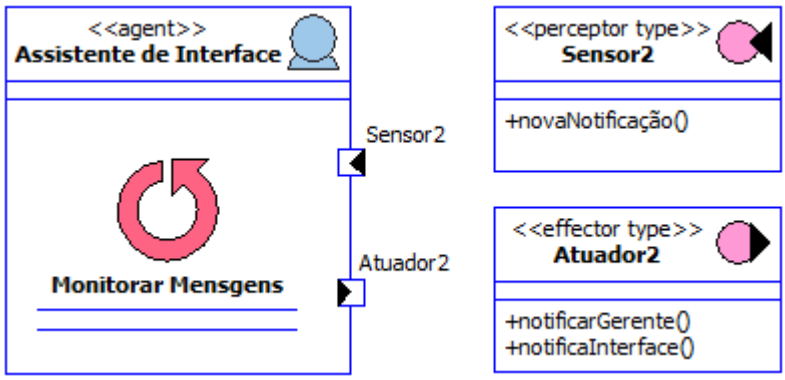
O agente Assistente de Monitoramento e Controle (AMC) tem um único comportamento com o objetivo de Monitorar os Riscos do projeto. Esse comportamento monitora as informações relacionadas aos possíveis riscos identificados pelo agente Assistente de Plano (AP) no início do projeto. Esse monitoramento foca em problemas comuns nos projetos de software, como: atrasos de atividades, saída de membros da equipe, segurança das informações, etc. Ao ser identificada alguma atividade relacionada aos riscos em potencial, o gerente de projeto é notificado. Este agente possui acesso à base de conhecimento onde são registrados os possíveis riscos, a Ontologia de Riscos (OR).

AGENTE ASSISTENTE DE INTERFACE - AI

Descrição: Notifica o gerente de projetos e os membros da equipe sobre qualquer informação enviada pelos outros agentes.

Objetivo: Notificar o gerente de projetos e os membros da equipe

Parâmetros de entrada: Nova notificação vinda de outros agentes.
Condição de ativação: Acompanhamento de um novo projeto.
Condição de finalização: Encerramento do projeto.
Descrição: O agente AI recebe mensagens que devem ser processadas e repassadas como notificações para o gerente de projeto ou para os outros membros da equipe.
Informação associada: A atividade de compartilhamento de informações está relacionada à área de gerenciamento de comunicação especificada no PMBOK.



A

Figura 122 ilustra o modelo de agente para o Assistente de Interface. O modelo apresenta o comportamento **Monitorar Mensagens** e os mecanismos internos utilizados durante a execução do agente. O quadro **Sensor2** representa o mecanismo do agente utilizado para perceber o ambiente, neste caso *novaNotificação()*. O quadro **Atuador2** representa os

mecanismos de atuação do agente, a forma como ele interage com o ambiente. Os mecanismos são: *notificarGerente(), notificaInterface().*

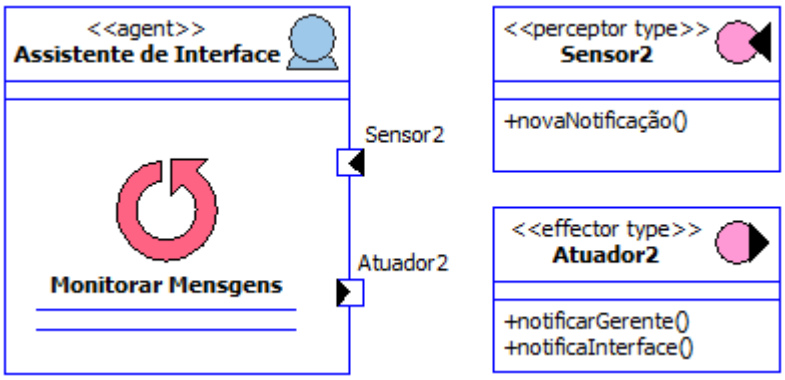


Figura 12. Modelo do Agente Assistente de Interface (CHAGAS, 2013)

O agente Assistente de Interface (AI) possui dois comportamentos: Notificar Gerente e Atualizar Interface. Esses dois comportamentos estão diretamente ligados ao gerenciamento de comunicação. Os outros agentes acionam o agente AI sempre que necessitam notificar o gerente de projetos ou a equipe de projeto através de e-mail ou interface do sistema.

4.2.6 Modelo de Atividade

Seguindo o processo de modelagem, após ser definido o modelo de agentes, pode ser desenvolvido o modelo de atividade dos agentes. Este modelo representa o fluxo principal de execução que será realizado pelo agente na tentativa de alcançar seus objetivos.

Agente Assistente de Requisitos - AR

a) Alertar sobre verificação de requisitos

A Figura 13 apresenta o modelo de atividade para o objetivo de alertar sobre a verificação de requisitos. Este objetivo está relacionado com os critérios adequados para a aceitação dos requisitos do projeto.

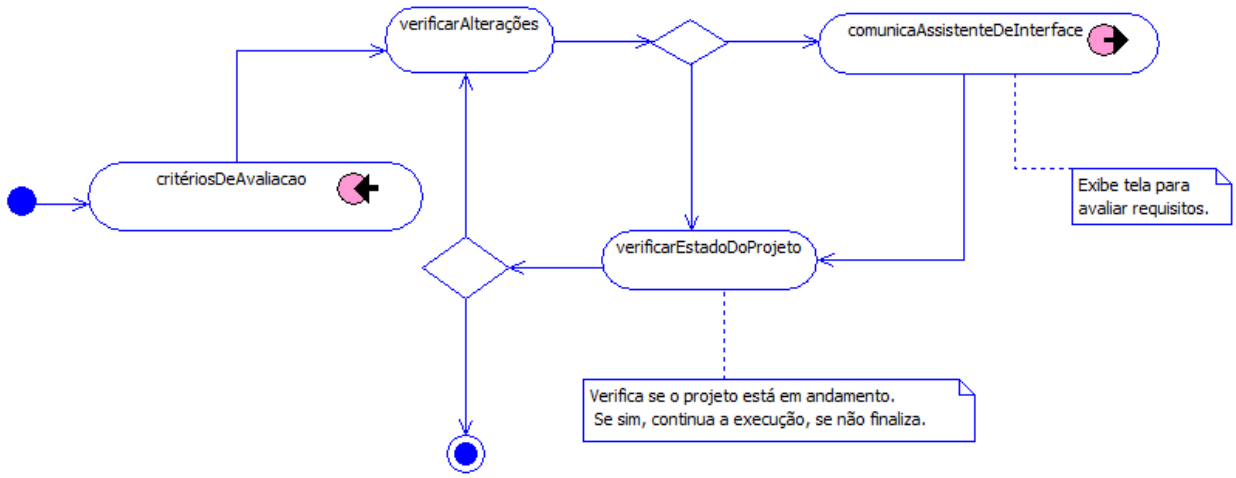


Figura 13. Modelo de Atividade - Verificar Requisitos (CHAGAS, 2013)

Como pode ser visto na Figura 13, ao ser iniciado um novo projeto, o agente AR recebe uma lista com os critérios para avaliação dos requisitos do projeto. Esta lista é definida pelo gerente de projetos após ser solicitado pelo sistema VPM no início do projeto. Então, o agente AR monitora os requisitos verificando alterações realizadas pelo gerente de projetos, sejam elas: adicionar, remover e/ou editar requisitos. Se o agente AR identifica uma tentativa de alteração, ele exibe uma tela com os critérios para avaliação de requisitos, para que o gerente de projetos lembre-se desses critérios e seja cauteloso no momento de realizar alterações. Caso não identifique alterações, o agente AR verifica se o projeto ainda está em andamento. Se sim, ele continua a execução, se não, ele finaliza sua atuação.

b) Gerenciar mudanças de requisitos

O modelo de atividade da Figura 14 representa o objetivo de gerenciar mudanças de requisitos. Com esse objetivo, o agente AR registra todas as informações de alterações realizadas sobre os requisitos durante o desenvolvimento do projeto.

Como apresentado na Figura 14, o agente AR monitora alterações nos requisitos. Ao identificar novas alterações, ele registra as alterações em um arquivo de *log* e comunica as alterações ao agente Assistente de Interface (AI). O agente AI, por sua vez, comunica o gerente de projetos sobre as alterações. O arquivo de *log*, com o histórico de alterações, fica disponível ao gerente de projetos. Essas informações podem ser utilizadas como dados históricos para futuros projetos.

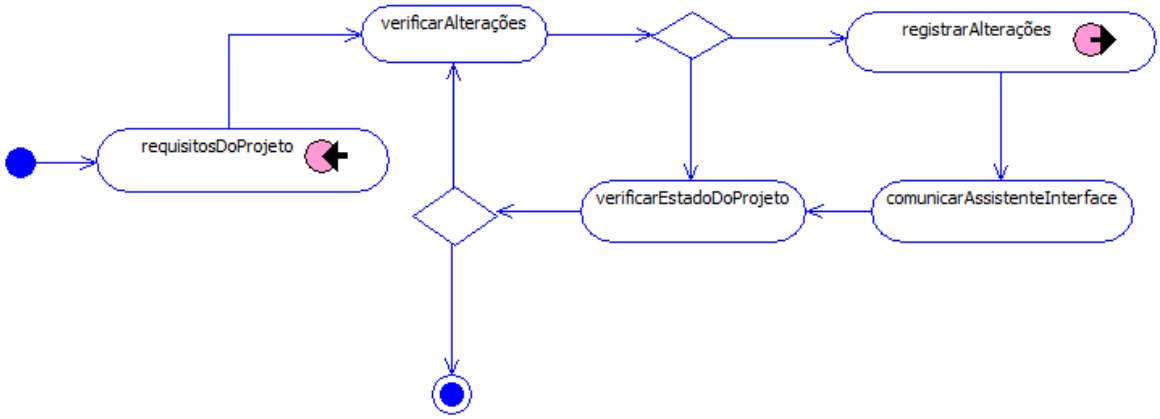


Figura 14. Modelo de Atividade - Gerenciar mudanças de requisitos (CHAGAS, 2013)

c) Manter a rastreabilidade de requisitos

A Figura 15 representa o modelo de atividade relacionado com o objetivo de manter a rastreabilidade dos requisitos. A rastreabilidade mapeia as dependências entre requisitos e assim indica qual requisito deve ser prioridade durante o desenvolvimento do projeto.

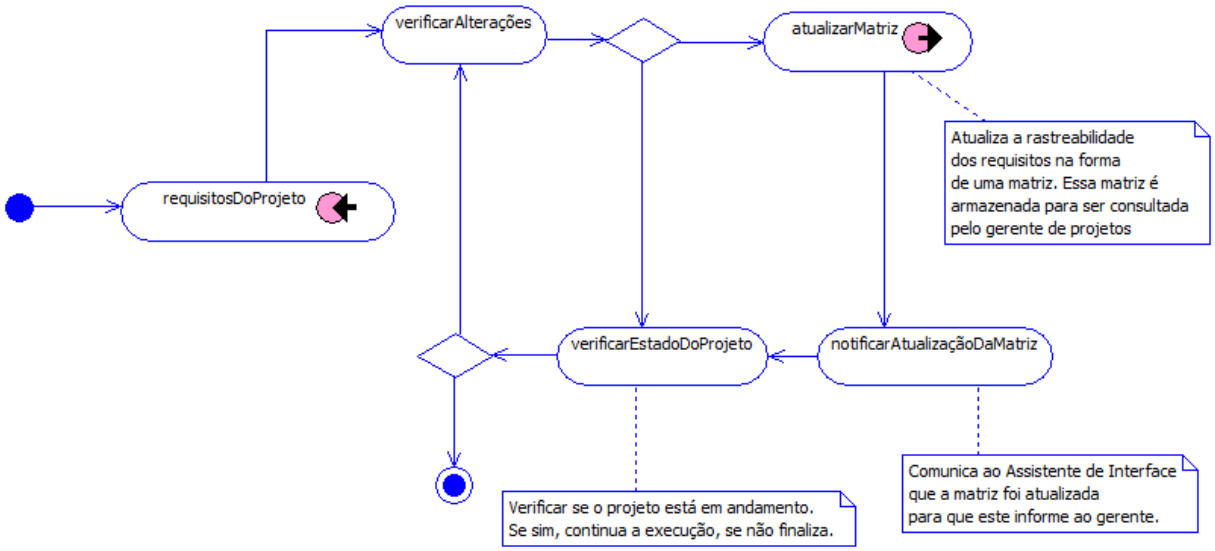


Figura 15. Modelo de Atividade - Manter Rastreabilidade (CHAGAS, 2013)

Como visto no modelo de atividade da Figura 15, o agente AR monitora alterações nos requisitos. Quando é identificada alguma alteração, o agente AR atualiza a matriz de rastreabilidade que registra a dependência entre requisitos. A matriz é armazenada no banco de dados do projeto e pode ser analisada pelo gerente de projetos a qualquer momento. Após atualizar a matriz, o agente AR comunica ao agente Assistente de Interface (AI) que, por sua vez, notifica o gerente de projetos. Após comunicar ao agente AI, ou não detectar nenhuma

alteração nos requisitos, o agente AR verifica se o projeto ainda está em andamento. Se sim, ele continua a execução, se não, ele finaliza sua atuação.

d) Garantir alinhamento entre artefatos e requisitos

O modelo de atividade da Figura 16 descreve o fluxo do assistente de requisitos para garantir o alinhamento entre artefatos e requisitos.

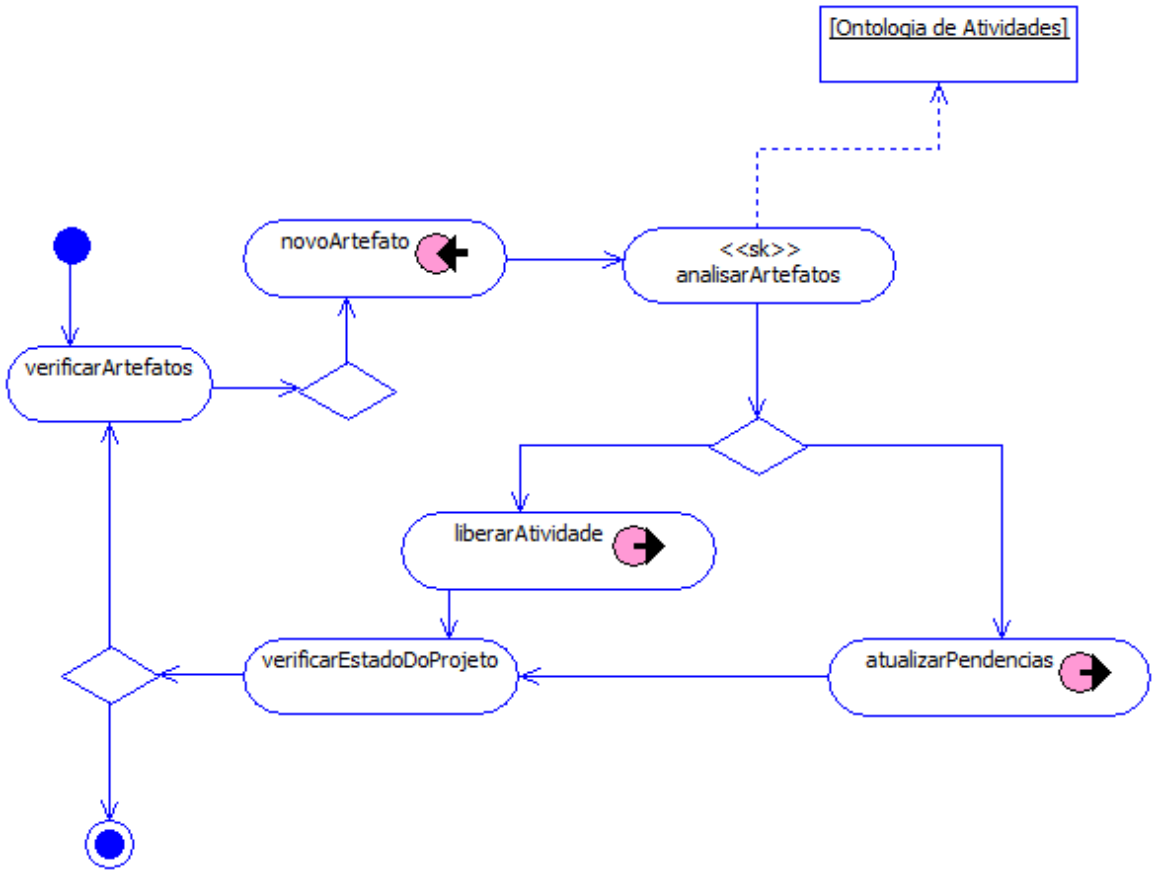


Figura 16. Modelo de Atividade - Garantir alinhamento de artefatos (CHAGAS, 2013)

Como apresentado na Figura 16, ao detectar um novo artefato no banco de dados do projeto, o agente AR analisa todos os artefatos. Para isso, ele consulta a Ontologia de Atividades (OA) para identificar quais artefatos de entrada são necessários para realizar cada atividade. Caso sejam satisfeitas todas as necessidades de artefatos de entrada para uma determinada atividade, ela será liberada para execução. Caso contrário, será atualizada a lista de artefatos pendentes para realização daquela atividade.

Agente Assistente de Plano - AP

a) Estabelecer estimativas

O modelo de atividade representado na Figura 17 mostra o fluxo de atividades do agente Assistente de Plano (AP) para realizar as estimativas de custo e tempo do projeto. As estimativas do projeto são atualizadas a cada marco do projeto.

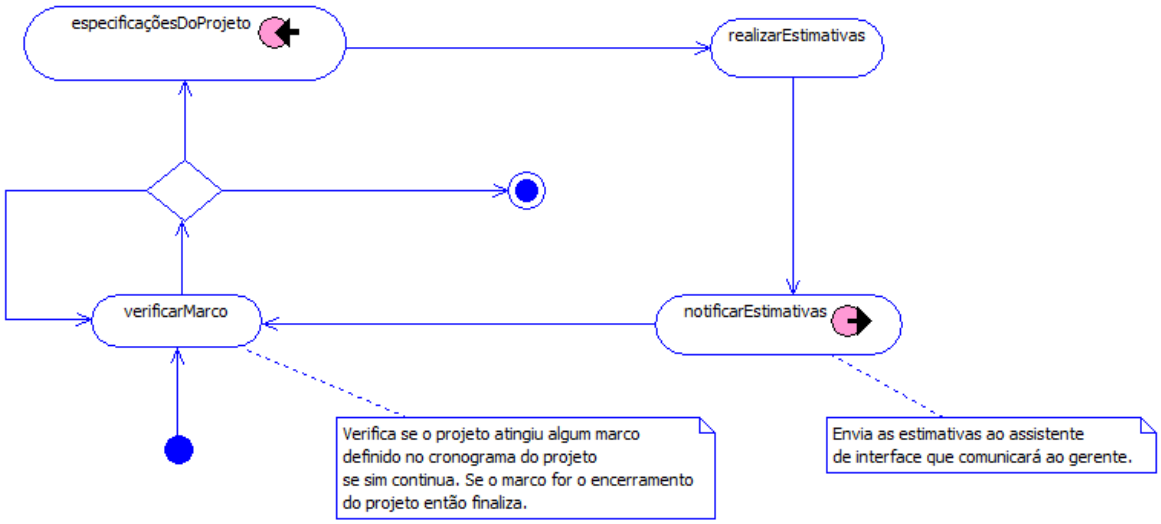


Figura 17. Modelo de Atividade - Estabelecer Estimativas (CHAGAS, 2013)

De acordo com o modelo de atividade da Figura 17, ao ser ativado, o agente AP solicita ao gerente de projetos as especificações do projeto e realiza as estimativas iniciais do projeto, considerando o marco inicial do projeto. A cada novo marco do projeto, o agente AP solicita novamente as especificações do projeto para que sejam atualizadas as estimativas. Assim, o gerente de projetos tem sempre estimativas atualizadas do projeto. Após serem realizadas as estimativas, o agente AP comunica ao agente Assistente de Interface (AI), que notifica o gerente de projetos sobre as novas estimativas.

b) Analisar riscos do projeto

Ao ser iniciado um novo projeto, o agente Assistente de Plano (AP) exibe um questionário ao gerente de projetos para saber quais são os possíveis riscos comuns a projetos de software que podem ocorrer no projeto em questão. Essas informações serão utilizadas

para monitoramento e priorização de riscos. O modelo de atividade para o objetivo Analisar Riscos pode ser observado na Figura 18.

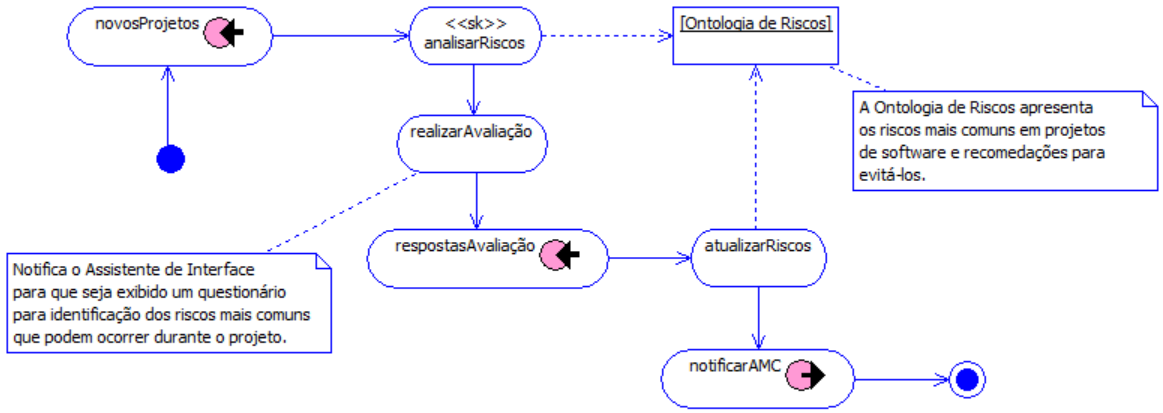


Figura 18. Modelo de Atividade - Analisar Riscos (CHAGAS, 2013)

Como pode ser visto no modelo de atividade da Figura 18, ao ser ativado, o agente AP carrega a lista de riscos da Ontologia de Riscos (OR) e exibe uma interface ao gerente de projetos com o objetivo de questioná-lo sobre os possíveis riscos do projeto. Após identificar os possíveis riscos, o agente AP atualiza a Ontologia de Riscos (OR) do projeto e notifica o agente Assistente de Monitoramento e Controle (AMC), que monitora os riscos do projeto.

c) Alocar recursos humanos

A alocação de recursos humanos é uma atividade crucial durante o desenvolvimento do projeto. O modelo apresentado na Figura 19 representa o modelo de atividade do agente Assistente de Plano (AP) para o processo de alocação de recursos humanos.

Como observado na Figura 19, ao detectar uma tentativa de alocação de recursos humanos para atividades do projeto, o agente Assistente de Plano (AP) consulta a Ontologia de Recursos Humanos (ORH) e realiza uma inferência sobre os conhecimentos necessários para realização das atividades e os conhecimentos disponíveis na equipe do projeto. Como resultado, o agente AP cria uma sugestão de alocação que é enviada ao agente Assistente de Interface, que notifica o gerente de projetos a sugestão do agente AP. A cada ciclo, o agente AP verifica se o projeto ainda está em andamento. Se sim, ele continua sua execução, caso contrário ele finaliza sua atuação.

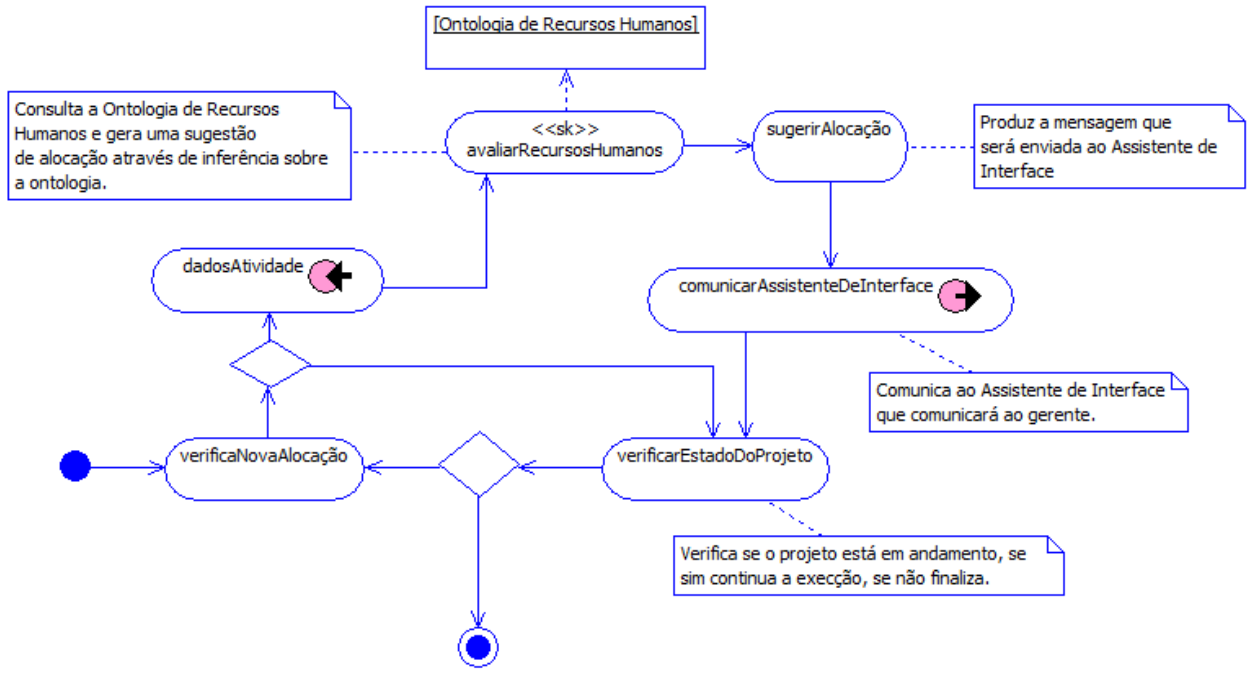


Figura 19. Modelo de Atividade - Alocar Recursos Humanos (CHAGAS, 2013)

Agente Assistente de Monitoramento e Controle - AMC

a) Monitorar riscos do projeto

O agente Assistente de Plano (AP) analisa e identifica os riscos. Porém, o agente Assistente de Monitoramento e Controle (AMC) é o responsável por monitorar os riscos do projeto. O modelo de atividades para este objetivo é apresentado na Figura 20.

Conforme o modelo de atividade da Figura 20, o agente Assistente de Monitoramento e Controle (AMC) monitora as ações do gerente de projetos durante a utilização do dotProject. A cada ação executada pelo gerente de projetos, o agente AMC notifica os possíveis riscos relacionados com aquela ação. Para isso, ele consulta a Ontologia de Riscos (OR) e realiza uma inferência sobre a priorização de riscos, para identificar quais riscos devem ser tratados em preferência a outros. O agente AMC comunica ao agente Assistente de Interface (AI), que notifica o gerente de projetos. A cada ciclo desse processo, o agente AMC verifica se o projeto ainda está em andamento. Se sim, ele continua sua execução, caso contrário, ele finaliza sua atuação.

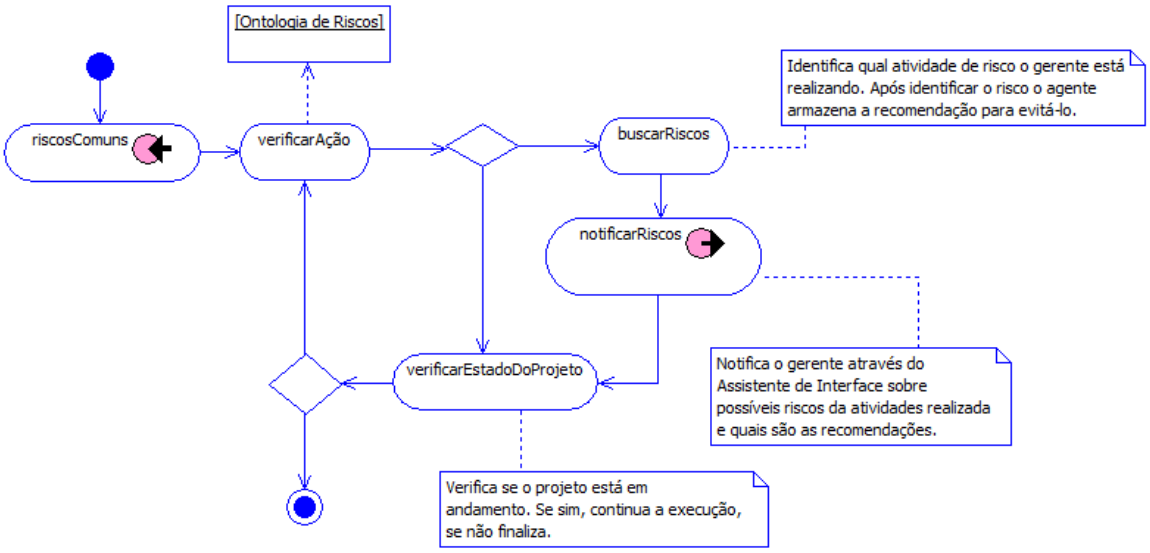


Figura 20. Modelo de Atividade - Monitorar Riscos (CHAGAS, 2013)

Agente Assistente de Interface - AI

a) Notificar gerente e membros da equipe

O agente Assistente de Interface (AI) é o responsável por tratar a comunicação entre os outros agentes e o gerente de projetos. O modelo de atividade do agente AI para o objetivo de notificar o gerente e membros da equipe é apresentado na Figura 21.

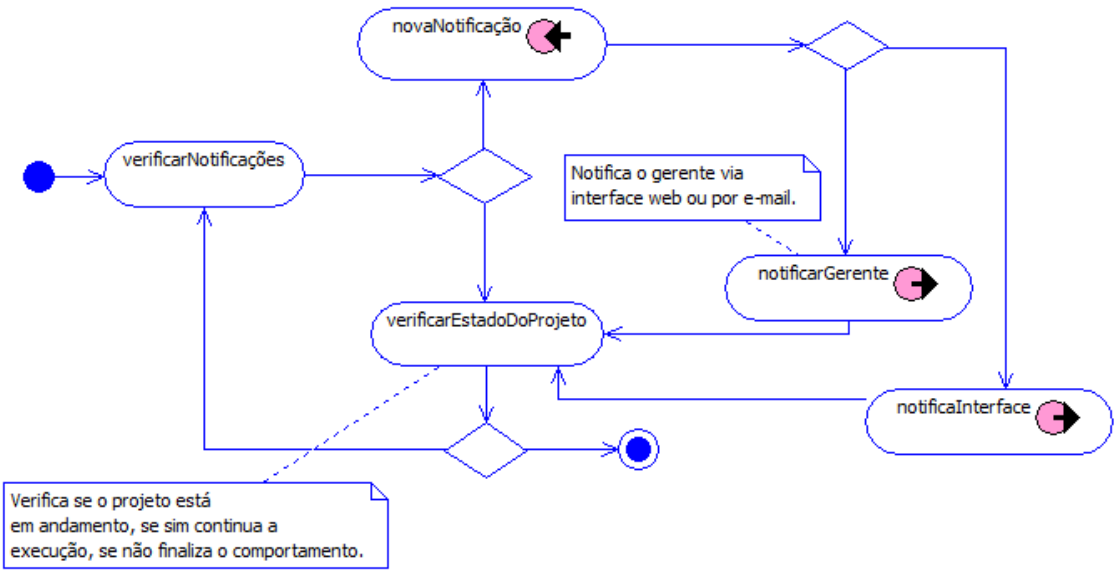


Figura 21. Modelo de Atividade – Notificar Gerente (CHAGAS, 2013)

Como pode ser visto no modelo de atividade da Figura 21, o agente AI fica constantemente verificando se existem notificações no seu repositório de mensagens. Se for encontrada uma nova mensagem, é gerada uma nova notificação, que é enviada ao gerente de projetos por e-mail (notificaGerente) e através da interface (notificaInterface) do VPM. A cada ciclo, o agente AI verifica se o projeto ainda está em andamento. Se sim, ele continua sua execução, caso contrário, ele finaliza sua atuação.

4.2.7 Modelos de Conhecimento

Os modelos de conhecimento são utilizados para definir a estrutura de armazenamento relacionada ao conhecimento interno dos agentes. Neste projeto os modelos são representados por ontologias. As ontologias foram desenvolvidas utilizando a ferramenta *Protégé* (KNUBLAUCH, FERGERSON, *et al.*, 2004). Esta é uma ferramenta de código-aberto que possui uma comunidade de milhares de usuários. Desde o seu desenvolvimento, ela tem sido direcionada para aplicações biomédicas, porém o sistema é independente de domínio e tem tido sucesso em muitas outras áreas de aplicação (KNUBLAUCH, FERGERSON, *et al.*, 2004).

A arquitetura do *Protégé* é separada em modelo e visão. O modelo é o mecanismo de representação interna para ontologias e bases de conhecimento. A visão do *Protégé* fornece uma interface de usuário para mostrar e manipular os modelos. Essa ferramenta permite carregar, editar e salvar ontologias em vários formatos (CLIPS[2], RDF, XML[3], UML[4] e bases de dados relacionais). E, recentemente, foi adicionado suporte para OWL (KNUBLAUCH, FERGERSON, *et al.*, 2004).

Ontologias são um dos principais focos da *Web* Semântica. Elas fornecem modelos formais de domínios de conhecimento que podem ser explorados por agentes inteligentes. O *plugin* OWL desenvolvido como extensão do *Protégé* pode ser utilizado para editar ontologias OWL e para acessar *reasoners* de descrição lógica (DL) (BAADER, CALVANESE, *et al.*, 2003). Como uma extensão do *Protégé*, o *plugin* OWL dispõe dos

[2] http://clipsrules.sourceforge.net/
[3] http://www.w3.org/XML/
[4] http://www.w3.org/wiki/UML_and_RDF

benefícios de uma grande comunidade de usuários, uma biblioteca de componentes reutilizáveis e uma arquitetura flexível (KNUBLAUCH, FERGERSON, *et al.*, 2004).

No sistema os modelos de conhecimento são representados através de ontologias. As ontologias do sistema são:

- Ontologia de Atividades (OA);
- Ontologia de Riscos (OR);
- Ontologia de Recursos Humanos (ORH).

a) Ontologia de Atividades (OA)

Esta ontologia representa o conhecimento que o agente possui sobre as atividades que devem ser realizadas durante o desenvolvimento do projeto de acordo com as recomendações do PMBOK. A Figura 22 apresenta a Ontologia de Atividades (OA). As classes principais desta ontologia são: Atividades, Subatividades e Artefatos. Além das classes citadas é apresentada a classe abstrata *Thing*, esta classe faz parte da representação utilizada pelo Protégé que define todas as outras classes como subclasses desta. As setas representam o relacionamento entre cada elemento da ontologia. As atividades estão agrupadas como instâncias: Acompanhar o Plano do Projeto, Elaborar / Replanejar Plano; Cancelar Projeto; Encerrar Projeto; Suspender Projeto; Estimar Parâmetros do Projeto; e Realizar Revisão Formal com Especialistas. Cada atividade possui subatividades e cada subatividade possui artefatos de entrada e de saída.

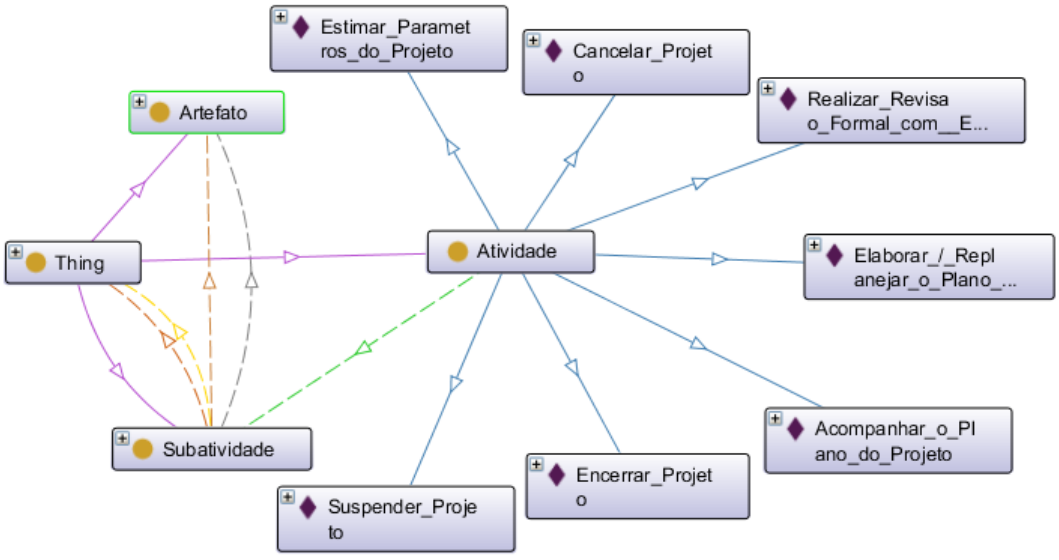


Figura 22. Ontologia de Atividades (CHAGAS, 2013)

O agente Assistente de Requisitos (AR) utiliza a ontologia apresentada na Figura 22 como base de conhecimento e realiza inferência sobre a ontologia para definir a dependência entre atividades de acordo com seus artefatos de entrada e saída. As regras para a inferência são mostradas a seguir:

```
@prefix ex: http://www.owl-ontologies.com/Ontology1308678440.owl#
[dependente:
        (?s rdf:type ex:Subatividade)
        (?v rdf:type ex:Subatividade)
        (?s ex:temEntrada ?x)
        (?v ex:temSaida ?x)
        ->
        (?s ex:dependenteDe ?v)
]
```

Nesse trecho de código é apresentada a definição da propriedade "dependente". A regra para definição dessa propriedade pode ser definida como:

Se S é uma subatividade e V é uma subatividade;
E S tem como entrada artefato X e V tem como saída artefato X;
Então S é dependente de V.

b) Ontologia de Riscos (OR)

A Ontologia de Riscos (OR) representa o conhecimento que o agente Assistente de Monitoramento e Controle (AMC) possui a respeito dos riscos mais comuns em projetos de software. Na ontologia OR, os principais conceitos representados são: Riscos, Ação, Etapa e Projeto. Os riscos estão agrupados em quatro classes: RiscosDoCliente, RiscosDeExecução, RiscosDePlanejamento e RiscosDeRequisitos. Para cada risco, existe um conjunto de recomendações que estão sumarizadas como subclasses da classe Ação. Para cada risco representado na ontologia OR existe um grupo de ações associadas para evitá-lo.

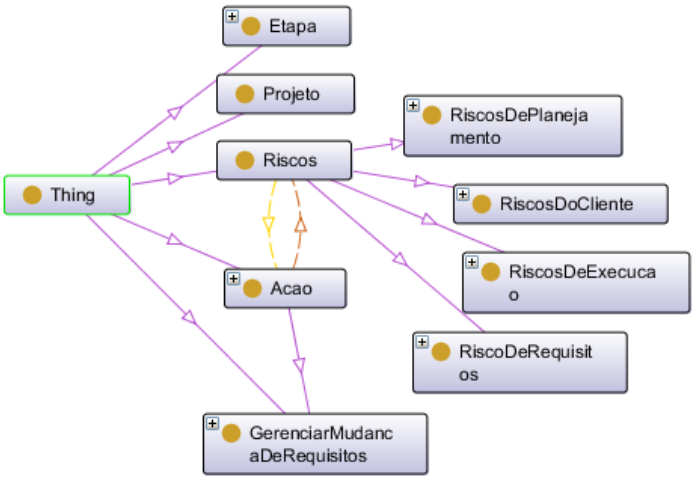


Figura 23. Ontologia de Riscos (CHAGAS, 2013)

A Ontologia de Riscos, apresentada na Figura 23, é utilizada pelo agente Assistente de Plano (AP) e pelo agente Assistente de Monitoramento e Controle (AMC). Este último agente realiza uma inferência para identificar a prioridade de riscos. As regras da inferência são apresentadas a seguir:

```
@prefix ex: http://www.owl-ontologies.com/Ontology1308678440.owl#
[priorizar:
        (?r1 rdf:type ex:Risco)
        {?r2 rdf:type ex:Risco)
        (?n1 rdf:type ex:Nivel)
        (?n2 rdf:type ex:Nivel)
        (?r1 ex:possuiNivel ?n1)
        (?r2 ex:possuiNivel ?n2)
        (?n1 > ?n2)
        ->
        (?r1 ex:temPrioridade ?r2)
]
```

Nesse trecho de código é apresentada a definição da propriedade "priorizar". A regra para definição dessa propriedade pode ser definida como:

Se R1 é um Risco e R2 é um Risco
R1 possui nível de prioridade N1
E R2 possui nível de prioridade N2
E n1 > n2
Então R1 tem prioridade sobre R2

c) Ontologia de Recursos Humanos

A Ontologia de Recursos Humanos (ORH) representa as informações sobre os conhecimentos envolvidos em atividades relacionadas ao projeto e relacionados às habilidades dos profissionais da equipe. Os conceitos chaves que a ontologia ORH representa são: Atividades, Competências, Membros da Equipe, Habilidades e Conhecimento.

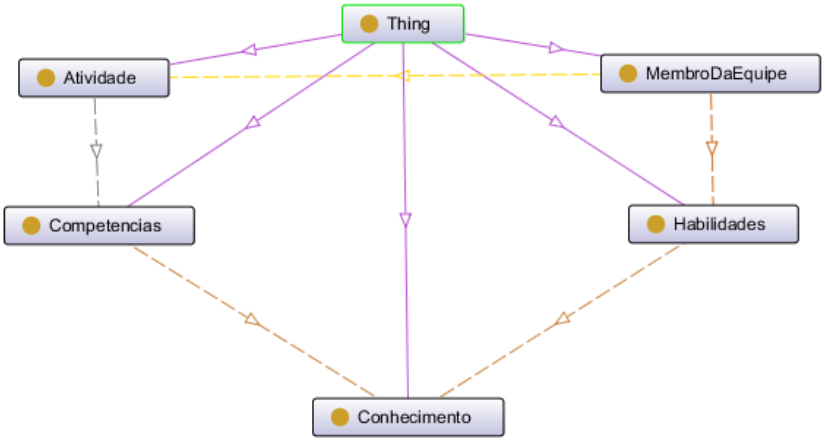


Figura 24. Ontologia de Recursos Humanos (CHAGAS, 2013)

A Ontologia de Recursos Humanos (ORH), mostrada na Figura 24, é manipulada pelo agente Assistente de Plano (AP). Ela é utilizada para gerar sugestões otimizadas de alocação de pessoal para atividades de acordo com suas habilidades e as competências necessárias para determinada atividade. As regras de inferência definidas para gerar as sugestões são mostradas a seguir:

```
@prefix ex: http://www.owl-ontologies.com/Ontology1308678440.owl#
[sugestao:
        (?m rdf:type ex:Membro)
        (?a rdf:type ex:Atividade)
        (?h rdf:type ex:Habilidade)
        (?c rdf:type ex:Conhecimento)
        (?cc rdf:type ex:Competencia)
        (?m ex:possui ?h)
        (?h ex:exige ?c)
        (?a ex:exige ?cc)
        (?cc ex:requer ?c)
        ->
        (?m ex:eAdequadoPara ?a)
]
```

Nesse trecho de código é apresentada a definição da propriedade "sugestão". A regra para definição dessa propriedade pode ser definida como:

Se M é um membro da equipe e possui habilidade H
E H está relacionado com o conhecimento C
E A é uma atividade que requer uma competência CC
E CC está relacionada com o conhecimento C
Então M é adequado para realizar atividade A

Para utilização das ontologias desenvolvidas para o sistema foi adotado o *framework* Jena (DICKINSON, 2013). O Jena[5] é um *framewok* Java para desenvolvimento e manipulação de ontologias criadas nos padrões RDF[6], OWL[7] e outros. O Jena fornece mecanismos para criar, manter e utilizar bases de conhecimento em *Web* Semântica, especificadas por ontologias utilizando a linguagem Java. Além disso, o *framework* possui uma máquina de inferência baseada em regras que permite estender as ontologias.

O *framework* trabalha com um modelo interno que é a representação em memória da base de conhecimento que será manipulada. Os modelos são criados com base nos padrões utilizados para criação de ontologias (RDF, SRDF[8], OWL, etc.). As especificações de cada padrão descrevem as características ou limitações das operações para cada padrão de

[5] http://incubator.apache.org/jena/
[6] http://www.w3.org/RDF/
[7] http://www.w3.org/OWL/
[8] http://www.w3.org/TR/rdf-schema/

representação. Além disso, disponibiliza métodos para validação das ontologias. Isso permite a verificação de modelos criados pelo usuário ou resultantes de inferência.

4.2.8 Modelo de Interação

Enquanto o modelo de atividade representa o fluxo interno de cada agente durante a sua execução, o modelo de interação é utilizado para representar o fluxo de execução no sistema, identificando todos os elementos envolvidos para que um dado objetivo seja completado. A seguir são descritos os modelos de interação para os objetivos dos agentes do VPM.

Agente Assistente de Requisitos - AR

a) Alertar sobre a verificação dos requisitos

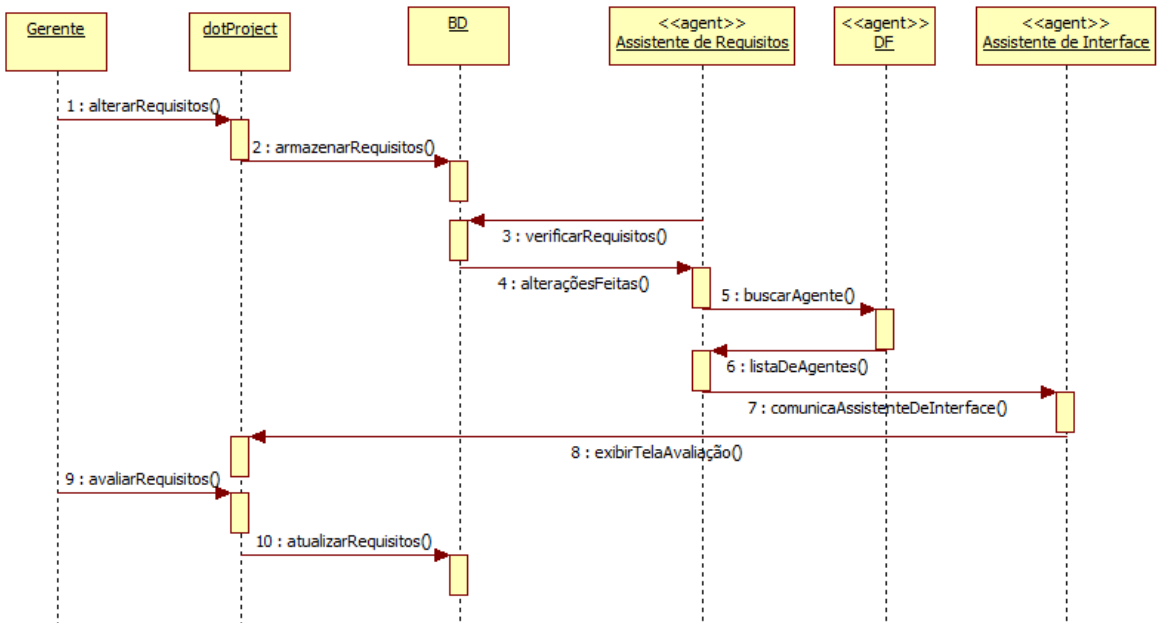


Figura 25. Modelo de Interação - Verificar Requisitos (CHAGAS, 2013)

Como pode ser visto na Figura 25, inicialmente o gerente de projetos altera algum requisito do projeto (1:alterarRequisitos()). Em seguida, o dotProject armazena o requisito no banco de dados do projeto, mas mantém seu estado "não verificado" (2:armazenarRequisitos()). O agente Assistente de Requisitos (AR) consulta então o banco de dados do dotProject em busca de requisitos com estado "não verificado" (3:verificarRequisitos). A seguir, o AR obtém do banco de dados a lista de requisitos com

estado “não verificado” (4:alteraçõesFeitas()). Após identificar os requisitos alterados, o agente AR solicita ao agente DF o endereço do agente Assistente de Interface - AI (5:buscarAgente()). O agente DF, por sua vez, retorna a lista de agentes disponíveis que fornecem o serviço de Assistente de Interface (6:listaDeAgentes()). Com o endereço do AI, o agente AR comunica ao agente AI que deve ser realizada uma verificação de requisitos e fornece os critérios para avaliação (7:comunicaAssistenteDeInterface()). O agente AI exibe a interface para avaliação de requisitos (8:exibirTelaAvaliação()) e o gerente de projetos analisa os critérios para avaliar os requisitos (9:avaliarRequisitos()). Após avaliar as alterações, o gerente de projetos finaliza as alterações e o dotProject atualiza os requisitos do projeto (10:atualizarRequisitos()).

b) Gerenciar mudanças de requisitos

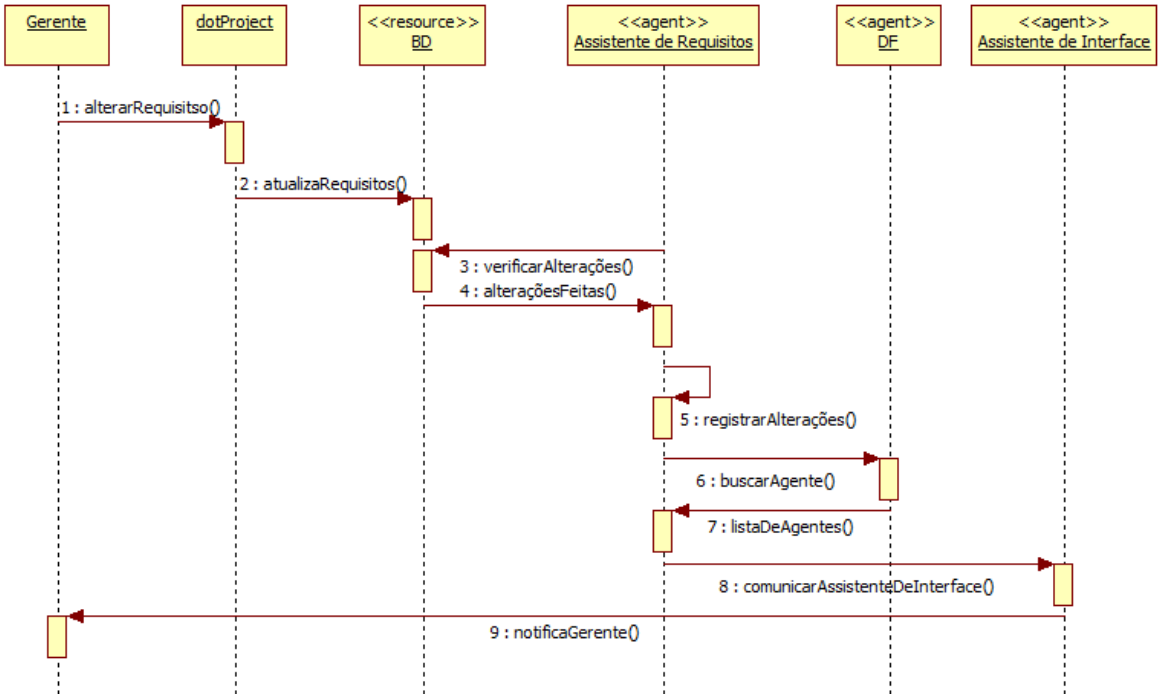


Figura 26. Modelo de Interação - Gerenciar Mudanças de Requisitos (CHAGAS, 2013)

Como pode ser visto no modelo de interação da Figura 26, quando o gerente de projetos realiza alterações nos requisitos do projeto (1:alteraRequisitos()), o dotProject atualiza os requisitos em seu banco de dados (2:atualizaRequisitos()). Percebendo isto, o agente Assistente de Requisitos (AR) consulta o banco de dados do dotProject em busca de requisitos com estado “não verificado” (3:verificarRequisitos). O AR obtém do banco de dados a lista de requisitos com estado “não verificado” (4:alteraçõesFeitas()). Em seguida, o

agente AR registra as alterações encontradas nos requisitos (5:registrarAlterações()). O agente AR solicita então ao agente DF o endereço de agentes que fornecem o serviço de Assistente de Interface (6:buscarAgente()). O agente DF retorna a lista de agentes do tipo Assistente de Interface (7:listaDeAgentes()). A seguir, o agente AR envia uma mensagem ao agente AI notificando sobre as alterações (8:notificarAlteraões()). Por fim, o agente AI notifica o gerente de projetos (10:notificaGerente()).

c) Manter a rastreabilidade de requisitos

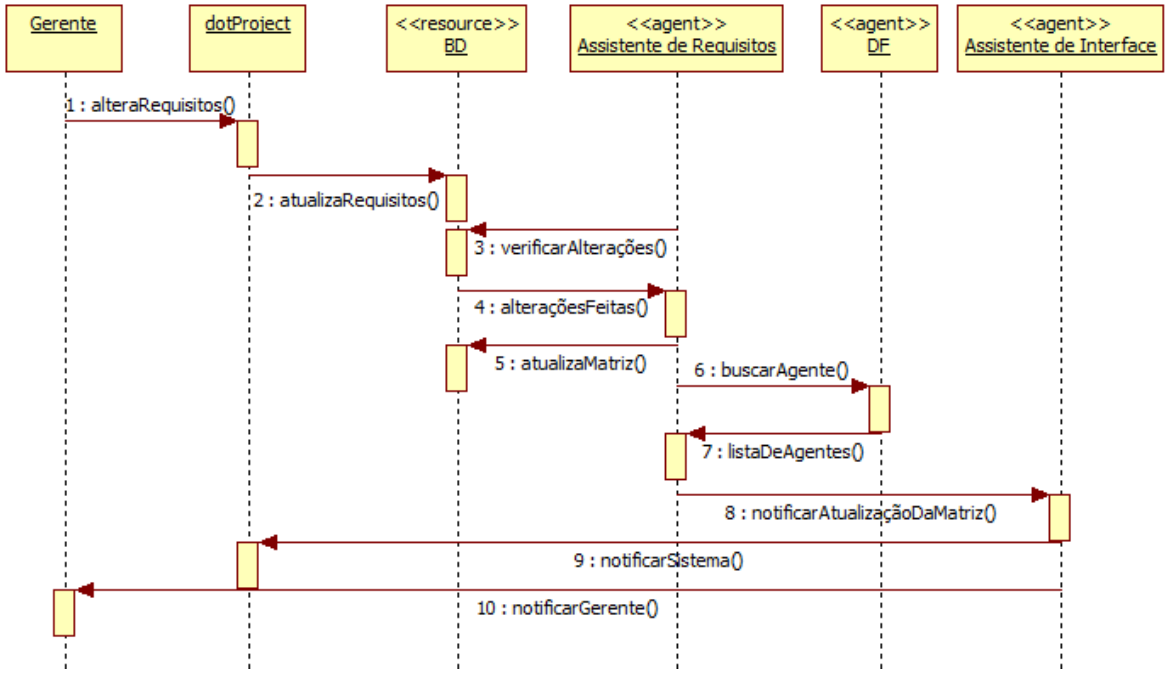


Figura 27. Modelo de Interação - Manter Rastreabilidade de Requisitos (CHAGAS, 2013)

Como apresentado na Figura 27, quando o gerente de projetos realiza alterações nos requisitos do projeto (1:alteraRequisitos()). O dotProject atualiza os requisitos em seu banco de dados (2:atualizaRequisitos()). Em seguida, o agente Assistente de Requisitos (AR) consulta o banco de dados do dotProject em busca de requisitos com estado "não verificado" (3:verificarRequisitos). O agente AR obtém do banco a lista de requisitos com estado "não verificado" (4:alteraçõesFeitas()). A seguir, o agente AR atualiza a matriz de rastreabilidade de requisitos (5:atualizaMatriz). O agente AR solicita então ao agente DF o endereço de agentes que fornecem o serviço de Assistente de Interface (6:buscarAgente()). O DF retorna a lista de agentes do tipo Assistente de Interface (7:listaDeAgentes()). O agente AR envia uma mensagem ao agente AI notificando sobre a atualização da matriz de rastreabilidade

(8:notificarAtualizaçãoDaMatriz()). O agente AI notifica então o sistema através da interface (9:notificarSistema()). Por fim, o agente AI notifica o gerente de projetos (10:notificaGerente()).

d) Garantir alinhamento entre artefatos e requisitos

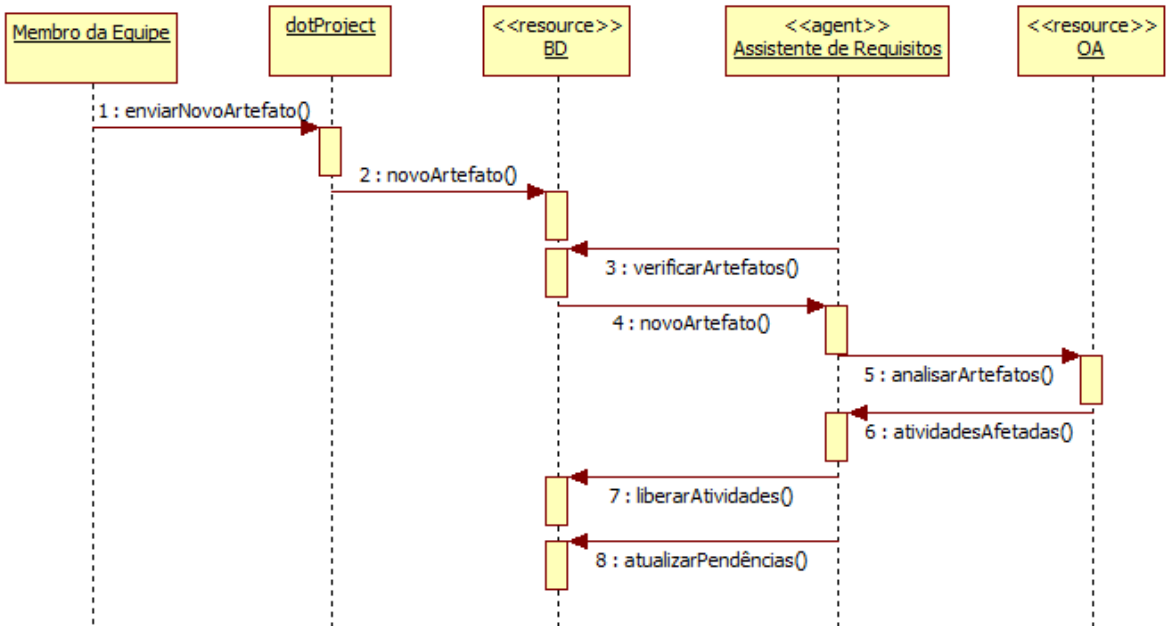


Figura 28. Modelo de Interação - Garantir alinhamento de artefatos (CHAGAS, 2013)

De acordo com o modelo de interação da Figura 28, quando um membro da equipe adiciona um novo artefato ao projeto através da interface do dotProject (1:enviarNovoArtefato()), o dotProject armazena o novo artefato no banco de dados do projeto (2:novoArtefato()). O agente Assistente de Requisitos (AR) verifica periodicamente se existem novos artefatos (3:verificarArtefatos()). O agente AR obtém do banco de dados a lista de novos artefatos associados ao projeto (4:novoArtefato()). Ao identificar novos artefatos, o agente AR consulta a Ontologia de Atividades (OA) para analisar quais atividades foram afetadas após a realização de tais artefatos (5:analisarArtefatos()). O agente AR obtém como retorno a lista de atividades afetadas pelos artefatos (6:atividadesAfetadas()). Após a análise, o agente AR identifica quais atividades podem ser liberadas para execução (7:liberarAtividades()). Além das atividades liberadas, o agente AR atualiza a lista de artefatos pendentes para realização das outras atividades.

Agente Assistente de Plano - AP

a) Estabelecer Estimativas

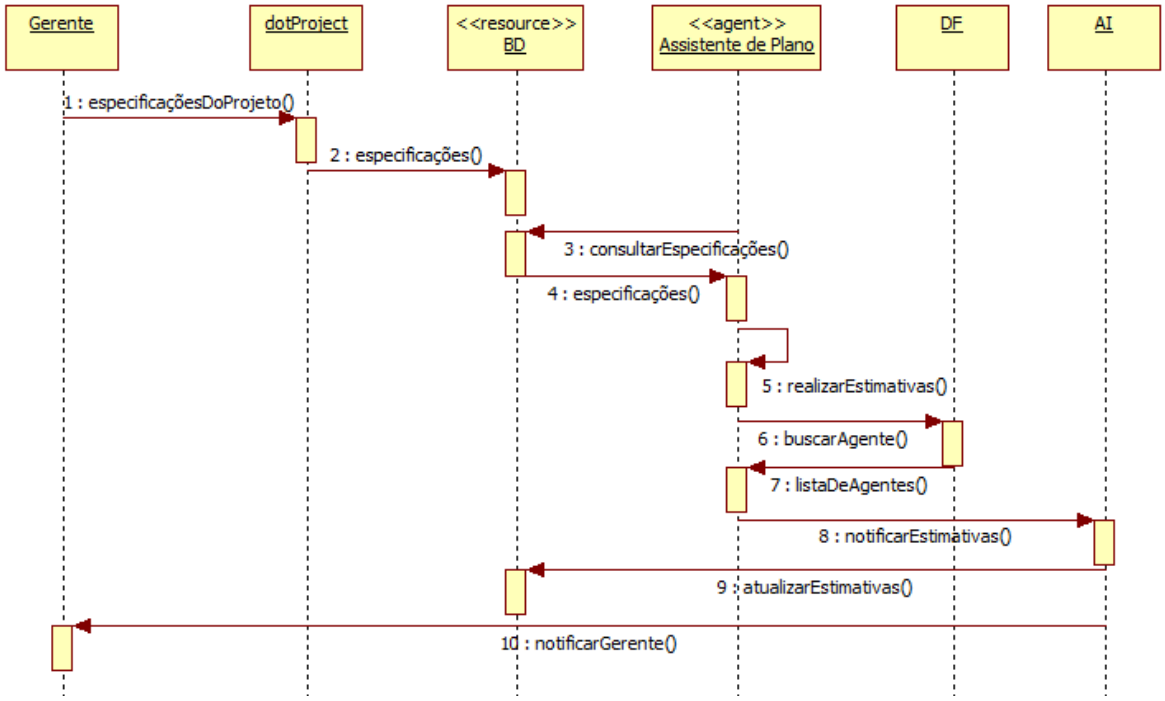


Figura 29. Modelo de Interação - Estabelecer Estimativas (CHAGAS, 2013)

Como apresentado na Figura 29, quando o gerente de projetos informa as especificações do projeto (1:especificaçõesDoProjeto()). O dotproject armazena a especificação do projeto no banco de dados (2:especificacoes()). Ao perceber isto, o agente Assistente de Plano (AP) consulta as especificações do projeto (3:consultarEspecificações()), obtendo a lista de especificações do banco de dados do dotProject (4:especificacoes). O agente AP realiza então as estimativas de custo e prazo, com base nas especificações encontradas (5:realizarEstimativas()). O agente AP contata o agente DF em busca do agente Assistente de Interface - AI (6:buscarAgente()). O agente DF retorna então a lista de agentes do tipo Assistente de Interface (7:listaDeAgentes()). O agente AP envia mensagem ao agente AI com as estimativas atualizadas (8:notificarEstimativas()). O agente AI atualiza as estimativas no banco de dados do dotProject (9:atualizarEstimativas()). Por fim, o agente AI notifica o gerente de projetos sobre a atualização das estimativas do projeto (10:notificarGerente()).

b) Analisar riscos do projeto

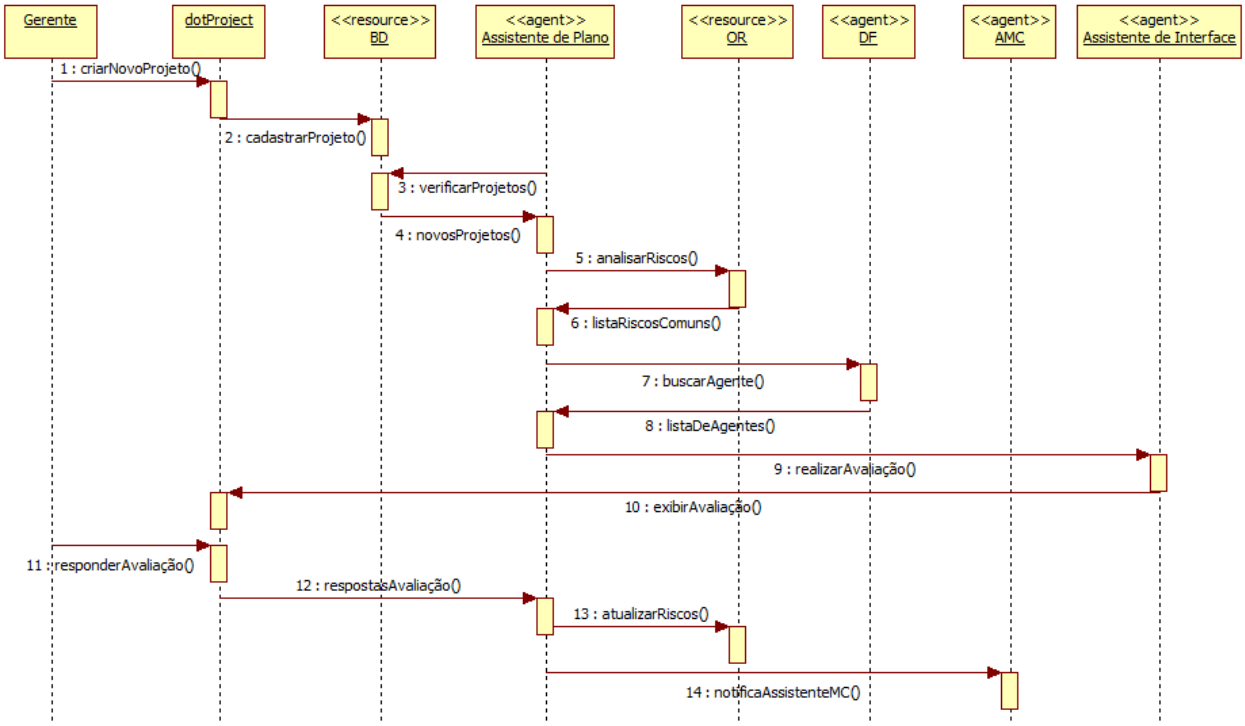


Figura 30. Modelo de Interação - Identificar Riscos (CHAGAS, 2013)

Como pode ser visto no modelo de interação da Figura 30, quando o gerente do projeto cria um novo projeto (1:criarNovoProjeto()). O dotProject armazena os dados iniciais do novo projeto (2:cadastrarProjeto()). O agente Assistente de Plano (AP) verifica, periodicamente, a inclusão de novos projetos no dotProject (3:verificarProjetos()), obtendo os novos projetos cadastrados (4:novosProjetos()). O agente AP consulta então a Ontologia de Riscos (OR) para identificar os riscos mais comuns (5:analisarRiscos()), obtendo a lista de riscos prováveis (6:listaRiscosComuns()). O agente AP solicita ao agente DF o endereço de agentes que fornecem o serviço de Assistente de Interface (7:buscarAgente()). O agente DF retorna a lista de agentes do tipo Assistente de Interface (8:listaDeAgentes()). Em seguida, o agente AP envia uma mensagem ao agente AI informando que deverá ser realizada a avaliação de riscos (9:realizarAvaliação()). O agente AI exibe um questionário ao gerente do projeto para a identificação dos possíveis riscos (10:exibirAvaliação()). O gerente do projeto responde ao questionário para identificação de riscos (11:responderAvaliação). O dotproject envia ao agente AP as respostas da avaliação para análise (12:respostasAvaliacao()). O agente AP atualiza a Ontologia de Riscos (OR) do projeto (13:atualizarRiscos()). Por fim, o agente AP notifica o agente AMC sobre a identificação dos possíveis riscos do projeto para monitoramento (14:notificaAssistenteMC()).

c) Alocar Recursos Humanos

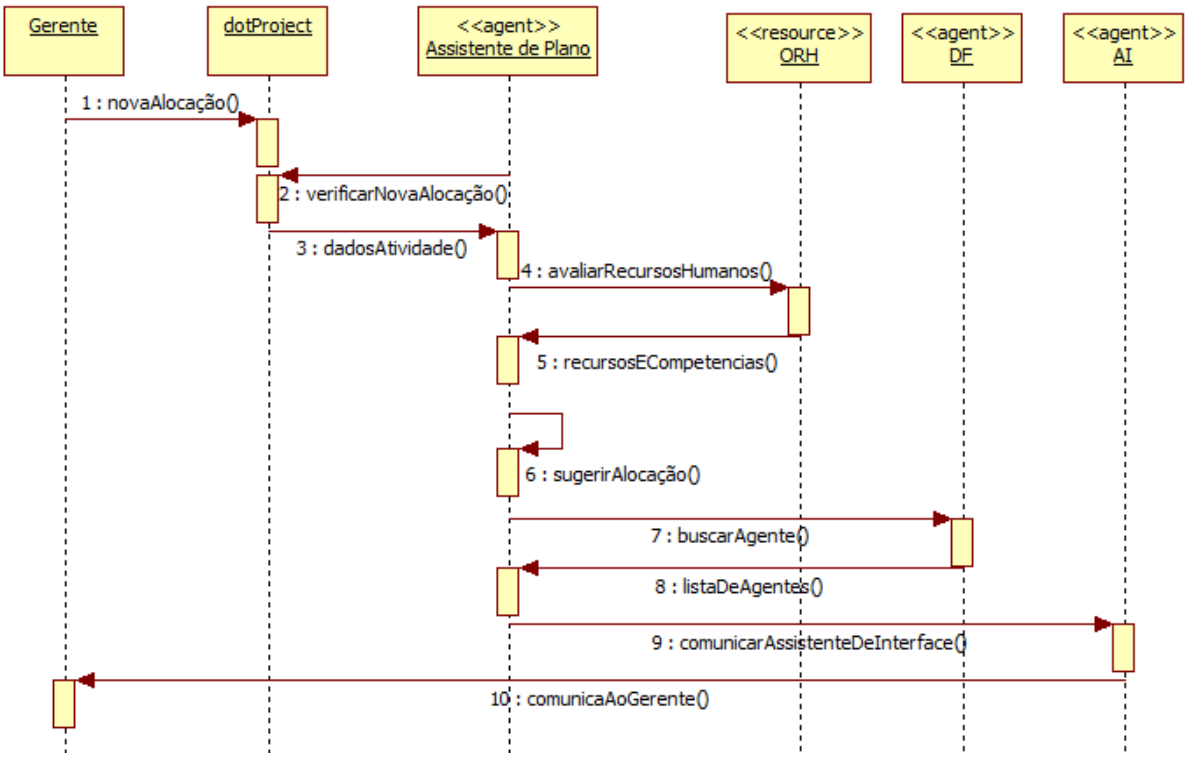


Figura 31. Modelo Interação - Alocar Recursos Humanos (CHAGAS, 2013)

Como pode ser observado na Figura 31, no momento em que o gerente de projetos inicia uma nova alocação de atividades (1:novaAlocação()). O agente Assistente de Plano (AP) verifica se existem novas alocações em processo (2:verificarNovaAlocação()), obtendo, a partir do dotProject, as informações iniciais sobre uma nova alocação (3:dadosAtividade()). O agente AP consulta então a Ontologia de Recursos Humanos (ORH) para identificar os conhecimentos necessários para uma atividade disponíveis na organização (4:avaliarRecursosHumanos()). O agente AP obtém da ontologia ORH os recursos e competências úteis para alocação (5:recursosECompetencias()). O agente AP gera então uma sugestão de alocação com base em uma inferência realizada sobre a ontologia (6:sugerirAlocação()). O agente AP contata o agente DF em busca do agente AI (7:buscarAgente()). O agente DF retorna a lista de agentes do tipo Assistente de Interface (8:listaDeAgentes()). O agente AP envia a sugestão de alocação ao agente AI (9:comunicarAssistenteDeInterface()). Por fim, o agente AI notifica o gerente de projetos sobre a sugestão de alocação (10:comunicaAoGerente()).

Agente Assistente de Monitoramento e Controle - AMC

a) Monitorar riscos do projeto

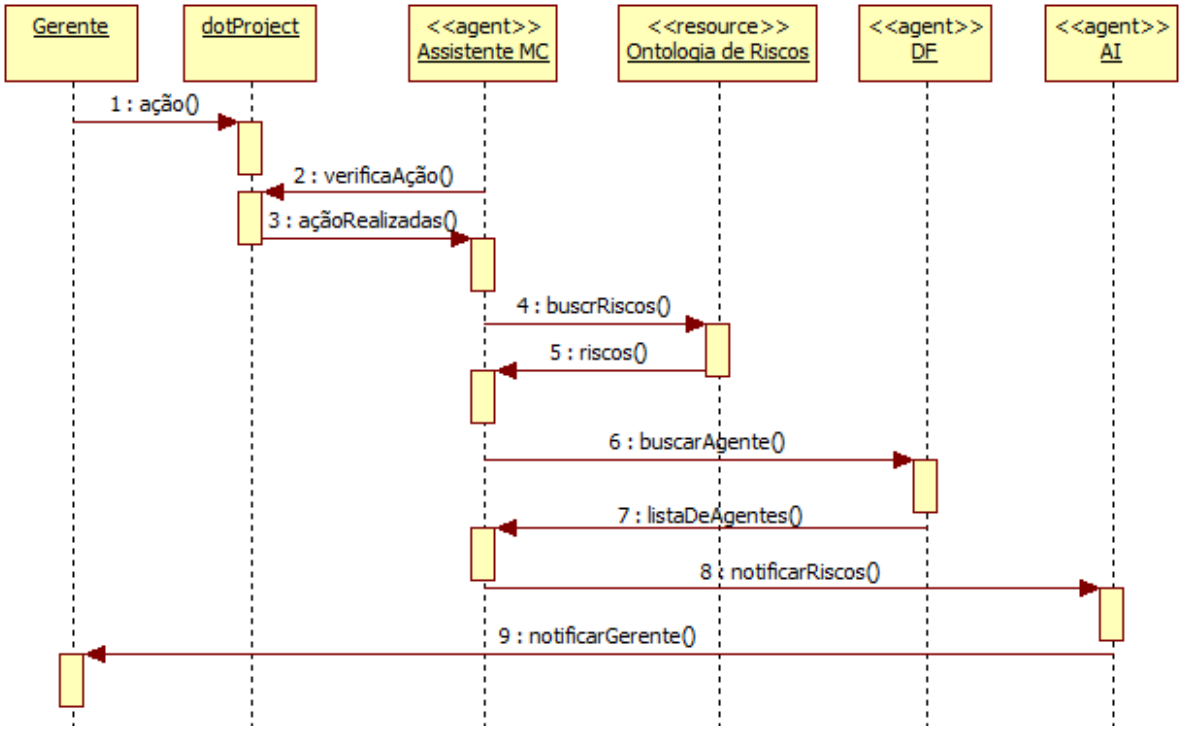


Figura 32. Modelo de Interação - Monitorar Riscos (CHAGAS, 2013)

Como pode ser visto no modelo de interação da Figura 32, quando o gerente de projeto realiza uma ação no dotProject (1:ação()), o agente Assistente de Monitoramento e Controle (AMC) percebe estas ações (2:verificaAção()), obtendo, do dotProject, a ação realizada (3:açãoRealizada()). O agente AMC busca na Ontologia de Riscos os possíveis riscos relacionados aquela ação (4:buscarRiscos()). O agente AMC obtém da ontologia os possíveis riscos (5:riscos()). O agente AMC contata o agente DF em busca do agente Assistente de Interface - AI (6:buscarAgente()). O agente DF retorna a lista de agentes do tipo AI (7:listaDeAgentes()). O agente AMC envia a notificação de risco ao agente AI (8:notificarRiscos()). Por fim, o agente AI notifica o gerente do projeto sobre os possíveis riscos, com as respectivas recomendações (9:notificarGerente()).

Agente Assistente de Interface - AI

a) Notificar gerente e membros da equipe

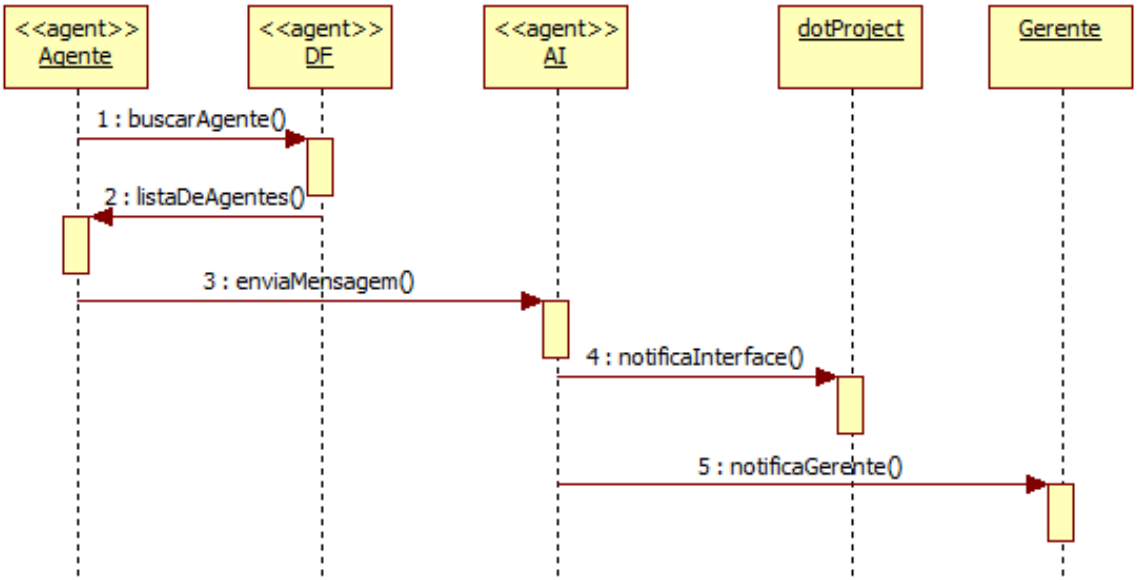


Figura 33. Modelo de Interação - Notificar Gerente (CHAGAS, 2013)

Como ilustrado no modelo de interação da Figura 33, quando um agente do VPM contata o agente DF em busca do agente Assistente de Interface – AI (1:buscarAgente()). O agente DF retorna a lista de agentes do tipo AI (2:listaDeAgentes()). O agente que iniciou o processo envia uma mensagem ao agente AI (3:enviaMensagem()). O agente AI notifica então os membros da equipe através da interface do VPM no dotProject (4:notificaInterface()). Por fim, o agente AI notifica o gerente de projetos por e-mail (5:notificaGerente()).

Para que esse processo de busca de agentes seja possível, todos os agentes devem registrar seus serviços no agente DF, que é o agente responsável pelo serviço de páginas amarelas. O modelo de interação para o processo de registro de agentes é apresentado na Figura 34.

Como pode ser observado na Figura 34, um agente, ao ser ativado, registra seu serviço no agente DF (1:registrarServiço()). Quando precisa dos serviços de outros agentes, ele aciona o agente DF buscando agentes identificados pelo serviço que eles oferecem (2:buscarAgente()). O agente DF, em resposta, retorna a lista de agentes disponíveis que fornecem o serviço solicitado (3:lista de Agentes).

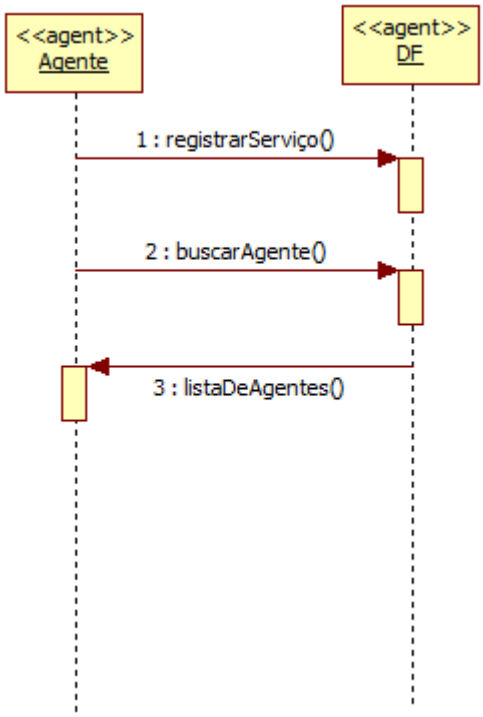


Figura 34. Modelo de Interação - Registro de Agentes no DF (CHAGAS, 2013)

4.3 CENÁRIOS DE USO

Nesta seção será apresentada uma descrição de cenários para a funcionalidade de gerenciamento de atividades da ferramenta VMP. O objetivo é apresentar, de forma contextualizada, a utilização das tecnologias e das abordagens do projeto.

As funcionalidades representadas pelos modelos de sequência nas Figura 35 e Figura 40 contemplam uma das recomendações do CMMI. Essa recomendação visa o correto alinhamento entre artefatos e requisitos. Segundo a recomendação, as atividades e artefatos devem ser revisados para que sigam o plano do projeto. O responsável por cumprir com esse objetivo é o agente Assistente de Requisitos (AR).

O modelo de sequência é baseado em um conjunto de passos realizados para cumprir determinada tarefa, além de mostrar quais atores e recursos estão envolvidos.

4.3.1 Cenário 1 - Atualizar Projetos

No cenário, descrito na Figura 35, é apresentado o processo que ocorre quando um novo projeto passa a ser acompanhado pela VPM.

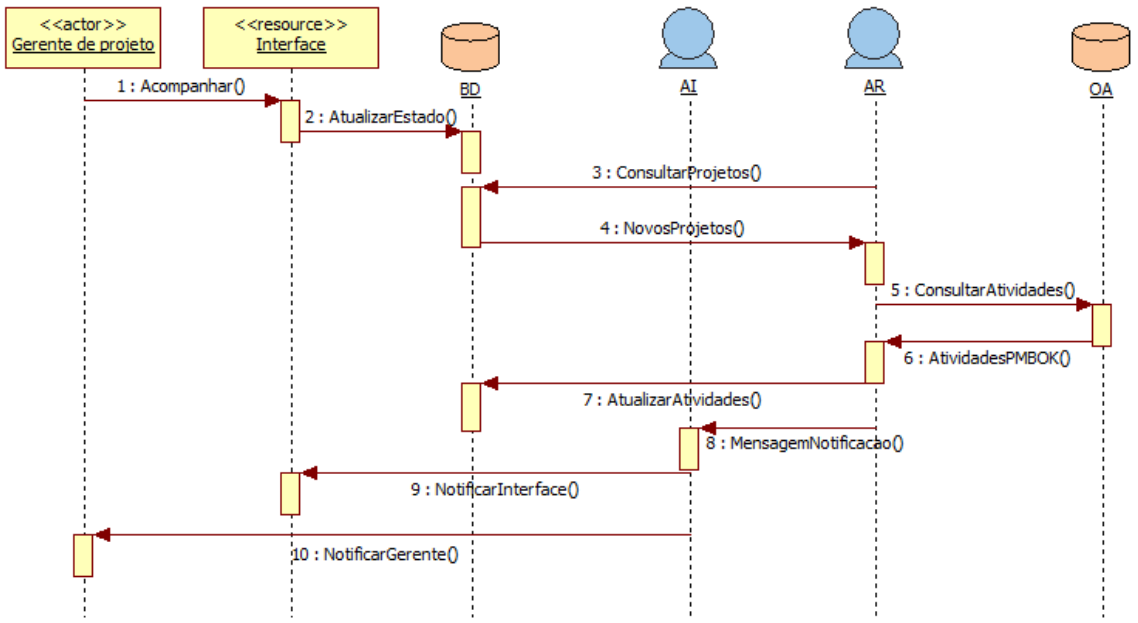


Figura 35. Modelo de Sequência para o Cenário 1 (CHAGAS, 2013)

Como pode ser observado no modelo apresentado na Figura 35, quando o gerente de projetos, utilizando a ferramenta através da interface *web* (Figura 36), define que um determinado projeto será acompanhado pela VPM (*1: Acompanhar()*). A interface, após a ação do gerente de projetos, atualiza o estado do projeto no banco de dados (*2: AtualizarEstado()*). Constantemente o agente AR irá consultar o banco de dados do dotProject em busca de novos projetos acompanhados (*3: ConsultarProjetos()*), obtendo a lista de todos os projetos atualmente acompanhados (*4: NovosProjetos()*). Quando o agente AR identifica um novo projeto acompanhado, ele consulta a Ontologia de Atividades - OA (*5: ConsultarAtividades()*). Essa consulta resulta na lista atualizada das atividades e subatividades que devem ser associadas aos projetos de acordo com o PMBOK (*6: AtividadesPMBOK()*). Quando o agente AR carrega as atividades da Ontologia de Atividades (OA), ele atualiza a lista de atividades dos projetos em acompanhamento pela VPM (*7: AtualizarAtividades()*). Após atualizar as atividades, o agente AR envia uma mensagem de notificação ao agente AI

(*8: MensagemNotificacao()*). O agente AI, ao receber a notificação do agente AR, notifica o gerente de projetos e a equipe via interface do sistema (*9: NotificarInterface()*) como mostrado na Figura 38. Em seguida, o agente AI notifica o gerente de projetos via e-mail (*10: NotificarGerente()*).

Figura 36. Interface para acompanhamento de projetos (CHAGAS, 2013)

A seguir, é apresentando o trecho de código das consultas realizadas pelo agente AR. O código foi desenvolvido em Java utilizando a API Jena (DICKINSON, 2013), que é uma API Java desenvolvida para manipulação de ontologias.

```
1. //Comando para consultar indivíduos da classe atividade na ontologia
2. String consultarAtividades = prefix + "SELECT ?atividade"
3. + "WHERE {?atividade rdf:type vcard:Atividade}";
4. /*Comando para consultar indivíduos no conjunto imagem da propriedade
5. possuiSubatividade da classe Atividade*/
6. String consultarSubatividades = prefix + "SELECT ?subatividade WHERE {
7. ?atividade rdf:type vcard:Atividade ."
8.    + "?atividade vcard:possuiSubatividade"
9.    + "?subatividade}";
10.
11. /*Realizando a consulta*/
12. Query quer1 = QueryFactory.create(consultarAtividades);
13. Query quer2 = QueryFatory.create(consultarSubatividades);
14. QueryExecution qe = QueryExecutionFactory.create(quer1, OA);
15.
16. //Conjunto de resultados para a primeira consulta
17. ResultSet results = qe.execSelect();
18. while(results.hasNext()){
19.   QuerySolution r = results.next();
20.   atividades.add(r.getResource("atividade")
21.   .getLocalName());
22. }
23. //Conjunto de resultados para a segunda consulta
24. qe = QueryExecutionFactory.create(quer2,OA);
25. results = qe.execSelect();
26. while(results.hasNext()){
27.   QuerySolution r = results.next();
28.   subatividades.add(r.getResource("subatividade")
29.   .getLocalName()):
30.   }
```

Figura 37. Consulta Ontologia de Atividades do PMBOK (CHAGAS, 2013)

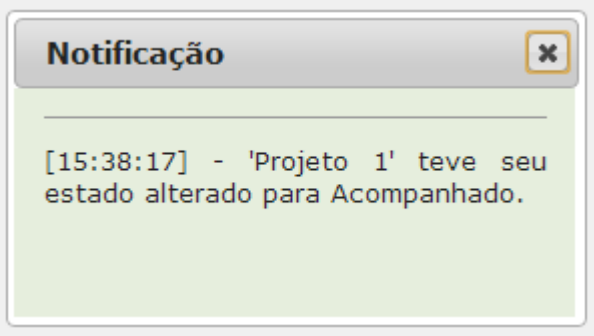


Figura 38. Mensagem exibida na área de notificação (CHAGAS, 2013)

4.3.2 Cenário 2 - Analisar Artefatos

No cenário ilustrado na Figura 40, é apresentado o processo que ocorre quando novos artefatos são associados aos projetos monitorados. São comparados os artefatos contemplados e os artefatos necessários para realização das tarefas.

Inicialmente um projeto em acompanhamento possui um conjunto de atividades bloqueadas que somente serão liberadas após a entrada dos artefatos iniciais do projeto. A interface referente a este estado do projeto é apresentada na Figura 39.

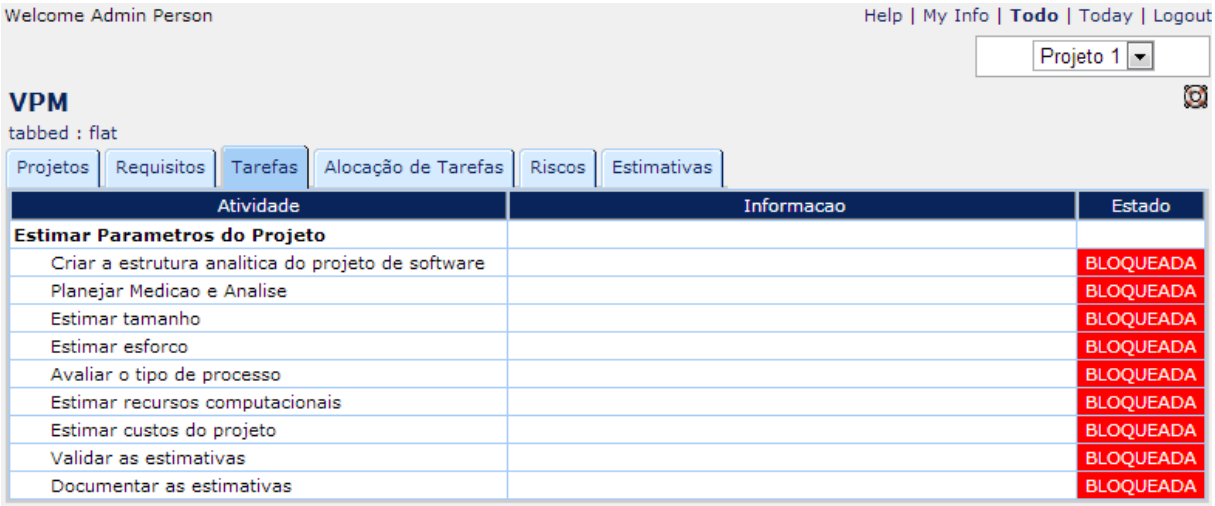


Figura 39. Interface apresentado a lista de atividades bloqueadas para o projeto (CHAGAS, 2013)

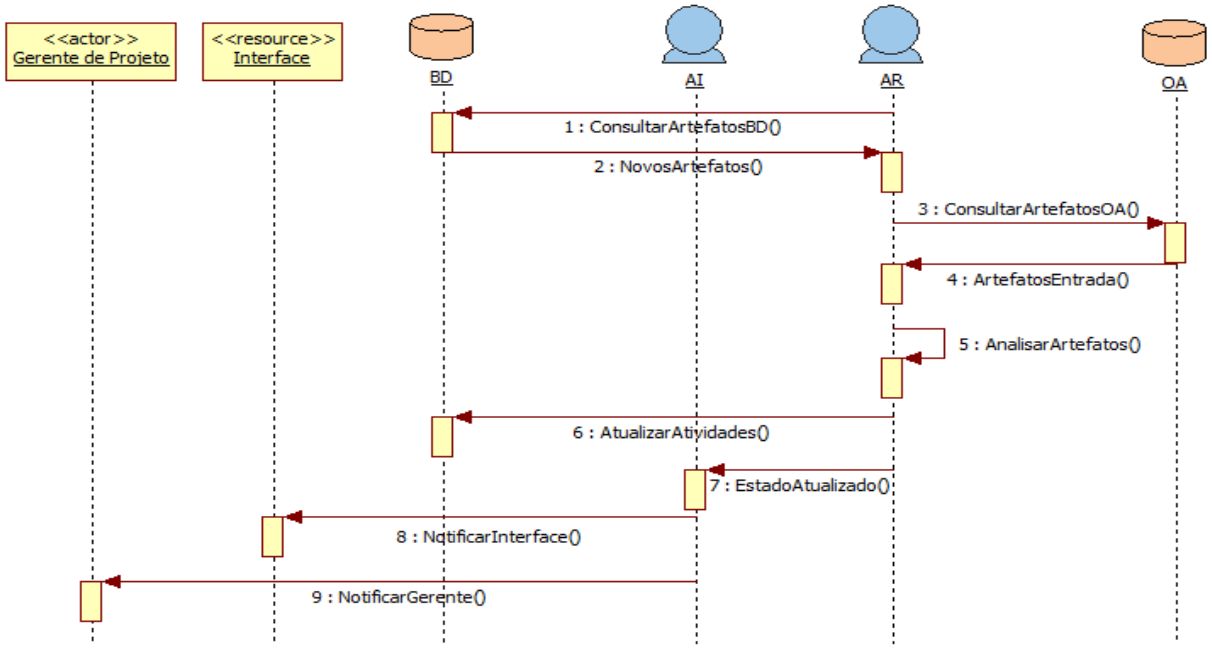


Figura 40. Modelo de Sequência para o Cenário 2 (CHAGAS, 2013)

Como pode ser visto na Figura 40, o agente AR monitora o banco de dados do dotProject em busca de novos artefatos associados aos projetos monitorados (*1: ConsultarArtefatosBD()*), obtendo a lista completa de artefatos associados aos projetos (*2: NovosArtefatos()*). Ao verificar que novos artefatos foram associados aos projetos monitorados, o agente AR consulta, na Ontologia de Atividades (OA), todos os artefatos necessários para entrada das atividades associadas aos projetos (*3: ConsultarArtefatosOA()*). A consulta aos artefatos é realizada utilizando o código a seguir.

```
1. String prefix = "PREFIX vcard: <"+str+">\n"
2.  + "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>";
3.
4. /*Comando para consultar indivíduos na ontologia que satisfazem a imagem da
5. propriedade temEntrada da classe Subatividade*/
6. String queryString = prefix + "SELECT ?artefato WHERE "
7.  + "{ ?atividade rdf:type vcard:Atividade ."
8.  +  ?atividade vcard:possuiSubatividade ?subatividade ."
9.  +  ?subatividade vcard:temEntrada ?artefato}";
10.
11. //Realizando a consulta
12. Query quer = QueryFactory.create(queryString);
13. QueryExecution qe = QueryExecutionFactory.create
14. (quer, OA);
15.
16. //Coletando os resultados
17. ResultSet results = qe.execSelect();
18. while(results.hasNext()){
19.   QuerySolution r = results.next();
20.   artefatos.add(r.getResource("artefato")
21.   .getLocalName());
22. }
```

Figura 41 Consulta de Artefatos na Ontologia de Atividades (CHAGAS, 2013)

A consulta realizada pelo agente AR retorna a lista de artefatos necessários para entrada das atividades (*4: ArtefatosEntrada()*). Com a lista de novos artefatos associados aos projetos e a lista de artefatos carregados da ontologia de atividades, o agente AR compara as duas listas. A partir dessas informações, o agente AR identifica quais atividades possuem o conjunto completo de artefatos de entrada e quais não possuem (*5: AnalisarArtefatos()*). Após realizar a análise dos artefatos, o agente AR atualiza o estado das atividades dos projetos monitorados. As atividades cujos artefatos de entrada já foram contemplados, são liberadas para execução, e as pendências das que não estão são informadas (*6: AtualizarAtividades()*). Quando o agente AR atualiza as atividades dos projetos, ele notifica o agente AI (*7: EstadoAtualizado()*). Ao perceber que os estados das atividades foram atualizados, o agente AI notifica via interface (*8: NotificaInterface()*). Além de notificar os usuários através da interface, o agente AI notifica o gerente de projetos via e-mail (*9: NotificarGerente()*).

Caso todos os artefatos para as atividades sejam contemplados a interface exibirá a lista de atividades com o estado LIBERADA para execução. Este estado das atividades é exemplificado na Figura 42.

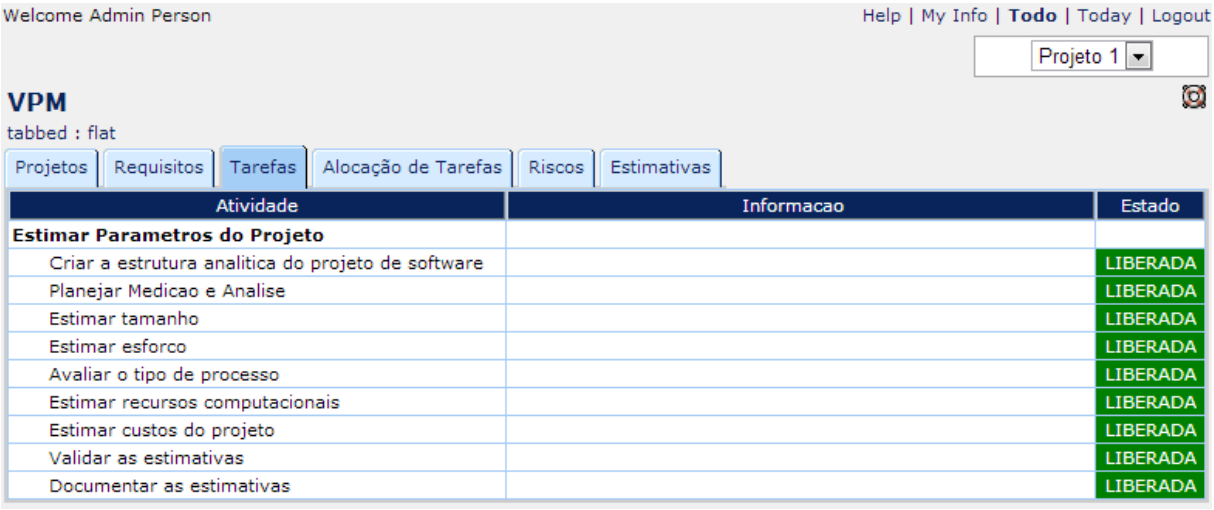


Figura 42. Interface exibindo o estado das atividades como LIBERADA (CHAGAS, 2013)

5 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Neste trabalho foi apresentada uma abordagem baseada em agentes e ontologias para apoiar o processo de desenvolvimento de software. A aplicação é um sistema multiagente de apoio à gestão de conhecimento em projetos de software. O propósito é diminuir a sobrecarga

cognitiva sobre os gerentes de projeto de ODS durante a implantação, ou manutenção, do CMMI-Nível 2. O CMMI é um modelo de melhoria de processo de software que propõe um conjunto de recomendações de melhores práticas para desenvolvimento de sistemas. Implantar modelos de melhoria como esse reflete diretamente nos custos, desempenho e qualidade dos produtos gerados. Então, a criação de tecnologias, ferramentas ou recursos que apoiem o processo de implantação ou manutenção desses modelos poderá auxiliar as ODS, principalmente aquelas que possuem limitações de recursos financeiros e humanos. Através do estudo realizado foi identificado que ainda são poucas as iniciativas para o desenvolvimento e/ou divulgação de ferramentas que auxiliem esse processo.

A modelagem do sistema foi realizada utilizando a metodologia MAS-CommonKADS+ para modelagem de agentes que também permite a modelagem de outras funcionalidades de sistemas de agentes. A implementação foi realizada utilizando o framework JADE para desenvolvimento de agentes e a API Jena para manipulação de ontologias. Todo esse processo poderá ser reutilizado por outros pesquisadores da área e nos nossos trabalhos futuros.

A solução apresentada neste trabalho se diferencia de algumas abordagens citadas na literatura por não criar passos adicionais ao processo de gestão, além dos passos recomendados pelo CMMI. Além disso, ainda não é muito explorado o uso de agentes de software e ontologias como forma de apoiar o trabalho dos gerentes de projetos nesse processo. Os agentes podem trabalhar de forma proativa durante a execução das tarefas de gerenciamento. As ontologias podem armazenar e gerar conhecimento durante a execução do projeto, além de tornar o sistema flexível quanto ao domínio do negócio. O uso dessas tecnologias torna a ferramenta mais flexível, visto que as algumas dificuldades do processo de gestão de projetos geralmente são muito subjetivas.

Como trabalhos futuros, pretende-se melhorar a arquitetura do SMA, adicionando novos agentes para contemplar outros níveis do CMMI, além de melhorar os agentes existentes em relação ao uso de outras técnicas inteligentes visando o aprimoramento de suas atividades. Pretende-se ainda tornar a arquitetura proposta genérica o suficiente para ser utilizada com outras ferramentas de gerenciamento de projetos, como a ferramenta TRAC (TEAM, 2012), e realizar um estudo de caso em uma ODS para validar na prática a eficácia do sistema.

6 REFERÊNCIAS

ALI, R.; IBRAHIM, S. **An application tool to support the implementation of integrated software process improvement for malaysia's sme.** 5th Malaysian Conference in Software Engineering. [S.l.]: IEEE. 2011. p. 177-182.

ALMEIDA, M. B.; BAX, M. P. Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. **Ciência da Informação**, Brasília, v. 32, p. 7-20, 2003.

ANSI; PMI. **A Guide to the Project Management Body of Knowledge:** PMBOK Guide. [S.l.]: Project Management Institute, 2008.

BAADER, F. et al. **The description logic handbook:** theory, implementation, and applications. Cambridge: Cambridge University Press, 2003.

BELLIFEMINE, F.; CAIRE, G.; GREENWOOD, D. **Developing multi-agent systems with JADE**. [S.l.]: Wiley, v. V, 2007.

BERNERS-LEE, T.; HENDLER , J.; LASSILA, O. The semantic web. **Scientific american**, v. 284, n. 5, p. 28-37, Maio 2001.

BRAGA, B.; PEREIRA, J. L. A. **Agentes Inteligentes - Conceitos , Características e Aplicações**. Belém: UNIVERSIDADE DA AMAZÓNIA, 2001.

BRIAND, L. C.; WÜST, J.; LOUNIS, H. Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs. **Empirical Software Engineering**, Março 2001. 11-58.

CHRISSIS, M.; KONRAD, M.; SHRUM, S. **CMMI® for Development:** Guidelines for Process Integration and Product Improvement. [S.l.]: Addison-Wesley Professional, 2011.

COURTNEY, A. **Phantom:** An interpreted language for distributed programming. Proceedings of the USENIX Conference on Object-Oriented Technologies on USENIX Conference on Object-Oriented Technologies (COOTS). [S.l.]: USENIX Association. 1995. p. 7-7.

DAVENPORT, T.; PRUSAK, L. **Conhecimento Empresarial**. Rio de Janeiro: Campos, 1998.

DICKINSON, I. The Jena Ontology API. **Apache Jena - The Jena Ontology API**, Janeiro 2013. Disponivel em: <http://jena.apache.org/documentation/ontology/>.

EHSAN, N. et al. **Comparative study for PMBOK & CMMI frameworks and identifying possibilities for integrations ITIL for addressing needs of it service industry**. International COnference on Management of Innovation and Technology. [S.l.]: IEEE. 2010. p. 113-116.

FIPA. Agent Management Specification. **The Foundation for Intelligent Physical Agents**, Janeiro 2004. Disponivel em: <http://www.fipa.org/specs/fipa00023/SC00023K.html>. Acesso em: 18 Outubro 2012.

FURUCHO, R.; AGUIAR, M. A. A importância do CMMI pra o sucesso das organizações. **Engenharia de Software Magazine**, p. 17-25, 2012.

GARCIA, I.; PACHECO, C.; CALVO-MANZANO, J. Using a web-based tool to define and implement software process improvement initives in a small industrial setting. **IET Software**, 2010. 237-251.

GRUBER, T. A translation approach to portable ontology specifications. **Knowledge Acquisition**, 1993. 199-220.

GUARINO, N. Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration. **Information Extraction A Multidisciplinary Approach to an Emerging Information Technology**, Frascati, Julho 1997. 14-18.

HUANG, D.; ZHANG, W. **CMMI in medium & small enterprises:** Problems and Solutions. The 2nd IEEE International Conference on Information Management and Engineering. [S.l.]: IEEE. 2010. p. 171-174.

JORDAN, L. **Gerenciamento de Projetos com dotProject:** guia de instalação, configuração, customização e administração do dotProject. [S.l.]: Pearson Prentice Hall, 2008.

KNUBLAUCH, H. et al. **The Protégé OWL Plugin:** An Open Development Environment for Semantic Web Applications. Third International Semantic Web Conference. Hiroshima: Springer. 2004. p. p. 229-243.

KOVACHEVA, T.; TODOROV, N. **Optimizing software development process:** A case study for integrated agile-cmmi process model. EUROCON - International Conference on Computer as a Tool. [S.l.]: IEEE. 2011. p. 1-2.

NELSON, G. **Systems programming with Modula-3**. [S.l.]: Prentice-Hall, Inc., 1991.

NOVELLO, T. C. Ontologias, Sistemas Baseados Em Conhecimento e Modelos de Banco de Dados, 2002. Disponivel em: <http://tools.assembla.com/svn/PFKaue/parteEscrita/Documentos/apoio/artigo_taisa.pdf>. Acesso em: 29 Janeiro 2013.

OLIVEIRA, M. J. **MAS-CommonKADS+: Uma extensão a metodologia MAS-CommonKADS para suporte ao projeto detalhado de sistemas multiagentes racionais**. [S.l.]. 2010.

RUMBAUGH, J.; JACOBSON, I.; BOOCH, G. **The Unified Modeling Language Reference Manual**. [S.l.]: Pearson Higher Education, 2004.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence:** A modern approach. [S.l.]: Prentice Hall, 2010.

SCHWABER, K.; BEEDLE, M. **Agile software development with Scrum**. [S.l.]: Prentice Hall, 2002.

SEI. Published CMMI appraisal results. **Published CMMI appraisal results**, Abril 2012. Disponivel em: <http://sas.sei.cmu.edu/pars>.

TAKEUCHI, H.; NONAKA, I. **Gestão do Conhecimento**. [S.l.]: Bookman, 2008.

TEAM, S. Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.3: Method Definition Document. **Software Engineering Institute**, 2011. Disponivel em: <http://www.sei.cmu.edu/library/abstracts/reports/11hb001.cfm>. Acesso em: 2012.

TEAM, T. The TRAC user and administration guide. **TRAC**, Dezembro 2012. Disponivel em: <http://trac.edgewall.org/wiki/TracGuide>.

ZAMITH, F. J. Ontologia. **PUC-RIO. Certificação Digital Nº 0024134/CA**, 1998. Disponivel em: <http://www2.dbd.puc-rio.br/pergamum/tesesabertas/0024134_02_cap_04.pdf>. Acesso em: 28 Janeiro 2013.

ZHANG, L.; SHAO, D. **Software process improvement for small and medium organizations based on CMMI**. 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC). [S.l.]: IEEE. 2011. p. 2402-2405.