UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UMA ARQUITETURA BASEADA EM CONTEXTO DE ATIVIDADES PARA GESTÃO DE CONHECIMENTO EM PROCESSOS DE TRABALHO

Leonardo Athanázio de Cerqueira Gatti

Orientadora

Flávia Maria Santoro

RIO DE JANEIRO, RJ – BRASIL.

SETEMBRO DE 2009

UMA ARQUITETURA BASEADA EM CONTEXTO DE ATIVIDADES

PARA GESTÃO DE CONHECIMENTO EM PROCESSOS DE TRABALHO

Leonardo Athanázio de Cerqueira Gatti

DISSERTAÇÃO APRESENTADA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE PELO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

Profª. Flávia Maria Santoro, DSc. (UNIRIO)

Prof. Sean Wolfgand Matsui Siqueira, DSc. (UNIRIO)

Profª. Vaninha Vieira dos Santos, DSc. (UFBA)

RIO DE JANEIRO, RJ – BRASIL.

SETEMBRO DE 2009

Gatti, Leonardo Athanázio de Cerqueira.
G263 Uma arquitetura baseada em contexto de atividades para gestão de conhecimento em processos de trabalho / Leonardo Athanázio de Cerqueira Gatti, 2009.
xv, 114f.

Orientador: Flávia Maria Santoro.
Dissertação (Mestrado em Informática) – Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2009.

1. Gestão do conhecimento. 2. Arquiteturas para sistemas sensíveis ao contexto. 3. Ontologias (Informática). 4. Web Semântica. I. Santoro, Flávia Maria. II. Universidade Federal do Estado do Rio de Janeiro (2003-). Centro de Ciências Exatas e Tecnologia. Curso de Mestrado em Informática. III. Título.

CDD – 005.5

Aos meus pais, minha irmã e

minha esposa, sempre presentes.

"O futuro pertence àqueles que acreditam na beleza de seus sonhos".

Elleanor Roosevelt.

AGRADECIMENTOS

Ainda lembro como se fosse hoje, há dez anos atrás, quando tomei a decisão que faria um mestrado acadêmico. Essa decisão norteou todos os meus caminhos até aqui. Foi assim que conheci pessoas maravilhosas, fiz amigos para toda a vida e aprendi mais do que poderia um dia sonhar. A seguir, meus agradecimentos, em ordem "quase" cronológica, para todos que, de alguma forma, me ajudaram a conquistar esse sonho.

Meus agradecimentos, ao grande amigo e orientador de iniciação científica, Professor Luiz Affonso Henderson Guedes de Oliveira, por todos os seus ensinamentos, conselhos e oportunidades de aprendizado.

A todos os grandes amigos que fiz no saudoso Laboratório de Computação Aplicada da UFPA, obrigado pela oportunidade de ter aprendido com vocês.

Aos amigos, Melina e Eduardo Guerra, por toda força, apoio, carinho e por terem participado de tantos momentos importantes da minha vida acadêmica, desde o início da graduação até aqui.

Aos primeiros amigos que fiz na cidade maravilhosa, Ermírio Coutinho, Jorge Lambert, Werner Priess e Ana Paula Mendes.

Aos amigos, José Marcos Filho, James Ribeiro, Gustavo Vergolino, Adriana Martins, Homero Fonseca, Fábio Costa, Bruna Martins, Márcio Ferreira, Leonardo Pedreira, Eliane Melo e Daniela Berger, pela amizade incondicional, mesmo com tantas ausências nos últimos anos.

Aos professores Paulo Rosa e Ricardo Choren, obrigado por terem me incentivado e encorajado a retomar o mestrado.

Ao amigo João Carlos Prates, por ter me ajudado a flexibilizar o meu horário de trabalho para conseguir retomar o mestrado.

Às professoras Renata Mendes Araújo e Ana Maria Moura, por terem acreditado e confiado em mim, dando-me as suas cartas de recomendação. Nunca esquecerei das lições de perseverança, determinação e qualidade presentes em todas as suas aulas.

A todos os meus amigos do mestrado em Informática da UNIRIO, obrigado por todos os momentos que passamos juntos. Todos os momentos ao lado de vocês foram especiais. Obrigado por ter aprendido com vocês. Em especial, aonde quer que você esteja, agradeço a querida amiga, Thalita Moraes (*in memoriam*), pelos trabalhos que fizemos juntos e por todos os momentos compartilhados de alegria e felicidade. A minha vitória, de certa forma, também é dela.

À minha inestimável orientadora e amiga, professora Flávia Maria Santoro, por todos os ensinamentos, paciência e momentos que passamos juntos nos últimos dois anos e meio. Obrigado por acreditar no meu trabalho e, principalmente, por me ajudar a não desistir e acreditar nele. Acima de tudo, obrigado por confiar e acreditar em mim, mesmo quando quase todos já não acreditavam.

À Vanessa Nunes, por ter aberto tantas frentes de pesquisa e pela disposição em ajudar e incentivar a continuidade desse trabalho.

Ao professor Sean, por todos os ensinamentos, contribuições e incentivos ao longo do curso.

À secretaria do PPGI, em especial, Alessandra Nascimento, obrigado por toda ajuda ao longo do curso.

Aos meus pais e avós, sem os quais esse trabalho não existiria. Obrigado por toda educação que vocês me deram. Obrigado por todo apoio e renúncias ao longo de toda a vida, para que eu pudesse chegar até aqui.

À minha irmã, Maíra, pela inestimável disposição em contribuir nas discussões sobre esse trabalho.

A minha amada esposa, Maria Fernanda, por todo carinho, compreensão, amor, respeito e incentivo. Por ter estado ao meu lado em todos os momentos mais difíceis e, principalmente, por ter me ajudado a seguir adiante em todos esses momentos. Sem ela, não consigo imaginar como teria sido possível.

Por fim, agradeço ao CNPq pelas bolsas que tive de iniciação científica; à CAPES pela bolsa de mestrado no IME; e à Accenture, em especial, Andréia Rodrigues, primeiro pela possibilidade de redução de carga-horária e, posteriormente, pela licença sem vencimentos, que me deu segurança e tranqüilidade, para conseguir terminar este trabalho.

GATTI, Leonardo Athanázio de Cerqueira. **Uma Arquitetura Baseada em Contexto de Atividades para Gestão de Conhecimento em Processos de Trabalho**. UNIRIO, 2009. 119 páginas. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.

RESUMO

Um dos desafios da gestão de conhecimento é capturar, armazenar e reutilizar o conhecimento contextual gerado pelos fluxos, processos e interações de trabalho que acontecem diariamente em uma organização. A criação de mecanismos que facilitem a reutilização do conhecimento contextual e a colaboração entre as pessoas, durante a execução de suas atividades, estimula a troca de informações, idéias e experiências que são realizadas continuamente dentro de um processo de trabalho. Nesta dissertação é proposta uma arquitetura baseada em contexto de atividades para suportar a gestão de conhecimento em processos de trabalho. O processamento, raciocínio e compartilhamento de conhecimento contextual envolvendo informações sobre as atividades são características necessárias para essa arquitetura. Foi realizada uma análise da literatura sobre arquiteturas para sistemas sensíveis ao contexto existentes que poderiam ser utilizadas na solução para o problema. Em seguida, foi realizada uma definição, instanciação e avaliação da arquitetura para um domínio específico. Os resultados indicam que a arquitetura proposta pode suportar as características necessárias para o problema pesquisado.

Palavras-chave: Gestão de Conhecimento, Contexto, Arquiteturas para Sistemas Sensíveis ao Contexto, Ontologias e Web Semântica.

ABSTRACT

The one of the main knowledge management challenges is to capture, store and reuse the contextual knowledge generated by the flows, processes and interactions that occur daily in an organization. The creation of mechanisms to facilitate the reuse of contextual knowledge and collaboration among people, while performing their activities, promotes the exchange of information, ideas and experiences that are continuously performed in a working process. It was proposed an activity context-aware architecture to support knowledge management in working processes. The required features for this architecture are processing, reasoning and contextual knowledge sharing involving information on the activities. It was presented a literature review on existents context-aware architectures that could be used in the solution of the problem. Afterwards, it was defined, instantiated and evaluated the architecture for a specific domain. The results indicate that the proposed architecture could support the required features of the investigated problem.

Keywords: Knowledge Management, Context, Context-Aware Architectures, Ontology and Web Semantic.

Índice

1 Introdução 1
1.1 Motivação 1
1.2 Caracterização do Problema 2
1.3 Enfoque da Solução 3
1.4 Objetivos 4
1.5 Delimitação da Abrangência do Trabalho 5
1.6 Organização da Dissertação 6
2 Gestão de Conhecimento e Contexto 7
2.1 Gestão de Conhecimento 7
2.1.1 Conceitos 7
2.1.2 Memória Organizacional 10
2.2 Conhecimento Contextual 13
2.3 Modelo de Referência para GC baseado em Contexto 14
2.4 Requisitos Arquiteturais 24
2.5 Considerações Finais do Capitulo 26
3 Arquiteturas para Sistemas Sensíveis a Contexto 28
3.1 Computação Sensível ao Contexto 28
3.2 Características de Arquiteturas de Sistemas Sensíveis ao Contexto 29
3.3 Modelagem de Contexto 35
3.3.1 Modelagem de Contexto baseado em Ontologia 36
3.4 Visão Geral de Frameworks e Arquiteturas para Sistemas Sensíveis ao Contexto 37
3.4.1 Arquiteturas 38
3.4.2 A Arquitetura CoBrA 40
3.4.3 Análise dos Requisitos para uma Arquitetura Base de Infra-Estrutura 45
3.5 Considerações Finais do Capítulo 47
4 A-CoBrA: Uma Arquitetura Baseada em Contexto de Atividades de Processos de Trabalho 49
4.1 Descrição de um Cenário 49
4.2 Descrição da Arquitetura 54
4.2.1 Fluxo de Interação 57
4.2.2 Modelo Conceitual 59
4.2.3 *Broker* de Contexto de Atividades de Processo 60
4.2.4 Representação de Contexto de Atividades de Processo 63
4.2.5 *Reasoning* de Contexto de Atividades de Processo 64
4.2.5.1 Regras de Reasoning de Contexto 67

4.3 Instanciando a Arquitetura 68
4.4 Considerações Finais do Capítulo 72
5 Avaliação da Arquitetura A-CoBrA: Uma Implementação do *Broker* de Contexto de Atividades para um Processo de Planejamento de Projetos 73
5.1 Metodologia de Avaliação 73
5.2 Descrição do Cenário e Processo de Negócio 75
5.3 Instanciação da Arquitetura 80
5.4 Instanciação e Extensão da Ontologia de Contexto para o Domínio do Processo de Negócio 81
5.4.1 Descrição de um Possível Cenário para Instâncias da Ontologia 81
5.4.2 Descrição da Ontologia de Contexto Estendida para o Domínio 82
5.4.3 Instâncias para as Classes e Atributos da Ontologia no Cenário Descrito 84
5.4.4 Consulta para Recuperar Instâncias 85
5.5 Definição de Regra de Inferência de Domínio 86
5.6 Resultados 87
5.7 Considerações Finais do Capítulo 89
6 Conclusões 90
6.1 Contribuições 91
6.2 Trabalhos futuros 92
Referências 94
Apêndice A – Ontologia estendida para o domínio de planejamento de projetos de software 104

Índice de Figuras

Figura 2.1 – Modelo de GC baseado em Contexto (NUNES, 2007). 15
Figura 2.2 – Ontologia de contexto focada em atividade (NUNES, 2007, NUNES, 2006). 18
Figura 3.1 – Arquitetura Genérica (SCHMOHL e BAUMGARTEN, 2008). 30
Figura 3.2 – Arquitetura CoBrA (CHEN, 2004). 42
Figura 4.1 – Um processo de aprovação de hipoteca simplificado. 50
Figura 4.2 – Arquitetura A-CoBrA. 56
Figura 4.3 – A-CoBrA e Atividades de Processos. 57
Figura 4.4 – A-CoBrA: Modelo Conceitual. 59
Figura 4.5 – Diagrama de Pacotes do A-CoBrA. 62
Figura 4.6 – Diagrama de seqüência para o recebimento de conhecimento contextual sobre atividades. 63
Figura 4.7 – *Handler* de *Reasoning* de Contexto de Atividades. 65
Figura 4.8 – Comportamento de recebimento de contextos de atividades até a chamada para o *reasoning* de contexto de atividades. 66
Figura 4.9 – *Handler* de *Reasoning* de Contexto de Atividades. 67
Figura 4.10 – Exemplo de instanciação do componente de *reasoning* de contexto de atividades. 69
Figura 4.11 – Diagrama de colaboração como exemplo de instanciação. 70
Figura 4.12 – Diagrama de colaboração como exemplo de instanciação com federação de *Brokers*. 71
Figura 5.1 – Processo de Planejamento de Projeto com Captura de Contexto. 77
Figura 5.2 – Exemplo de instanciação do componente de *reasoning* de contexto de atividades para a atividade de planejamento da velocidade e tamanho do projeto. 80
Figura A.1 – Ontologia estendida para o domínio de planejamento de projetos de software. 104

Índice de Tabelas

Tabela 2.1 – Descrição das classes e o relacionamento entre elas (NUNES, 2007). 19
Tabela 2.2 – Grupos de Classes vs. Mecanismos de Captura (NUNES, 2007)..... 22
Tabela 3.1 – Critérios de escolha de arquitetura base. 46
Tabela 4.1 – Conceitos adicionados à ontologia de contexto. 64
Tabela 4.2 – Regra para validar se um indivíduo tem competência para desempenhar determinado papel..... 68
Tabela 4.3 – Regra para validar se um indivíduo tem perfil para desempenhar determinado papel..... 68
Tabela 5.1 – Consulta RDQL para recuperar informações de domínio relevantes para a atividade de Planejar Velocidade do Projeto e Duração..... 86
Tabela 5.2 – Regra para análise de risco sobre a ontologia de contexto estendida para o domínio de gerenciamento de projetos de software. 87

Lista de Abreviaturas Utilizadas

API *Application Programming Interface*

GC *Gestão de Conhecimento*

MO *Memória Organizacional*

OWL *Web Ontology Language*

RDF *Resource Description Framework*

XML *eXtensible Markup Language*

W3C *World Wide Web Consortium*

1 Introdução

Este capítulo apresenta um resumo da pesquisa realizada, incluindo as motivações, descrição do problema, enfoque de solução, objetivos e a organização do texto.

1.1 Motivação

Nas últimas décadas, tornou-se notório o aumento da competitividade entre as organizações, atuantes em mercados cada vez mais globalizados, e ao mesmo tempo suscetíveis às variações e aos obstáculos de ordem política e econômica que surgem a todo instante. A eficiência no desenvolvimento e no gerenciamento do conhecimento organizacional são fatores decisivos para o sucesso, para a qualidade dos produtos e serviços oferecidos por uma organização competitiva (NONAKA e TAKEUSHI, 1991, NONAKA *et al*., 2008). No entanto, as instituições reconhecem que a gestão de conhecimento está intrinsecamente ligada aos processos e às atividades colaborativas realizadas diariamente pelos funcionários de uma organização.

Não é incomum que o conhecimento de uma organização esteja fragmentado em diversos artefatos, ou mesmo, na experiência de poucos que conhecem os processos, as atividades e o dia a dia da organização. Reter esse conhecimento em uma estrutura de memória organizacional é importante para o desenvolvimento da organização, assim como é importante integrá-lo às atividades de forma a torná-lo visível e acessível por todos.

A criação de mecanismos que facilitem a reutilização do conhecimento, e a colaboração entre as pessoas durante a execução de suas atividades, pode ser importante para a organização, ao passo que estimula a troca de informações, idéias e experiências sobre atividades, que estão inseridas e são realizadas continuamente dentro de um processo de trabalho.

1.2 Caracterização do Problema

Um dos desafios da gestão de conhecimento (GC) no contexto organizacional é capturar, armazenar e reutilizar o conhecimento gerado pelos fluxos, processos e interações de trabalho que acontecem diariamente em uma organização. No entanto, os artefatos e informações capturados ao final da execução dos processos, comumente não representam o conhecimento necessário para a execução das atividades realizadas.

A necessidade de externalização do conhecimento tácito presente nos papéis desempenhados pelas pessoas, durante a execução dos processos de negócios, esbarra na falta de tempo e motivação para compartilhar, o que muitas vezes parece óbvio ou trivial, mas que pode ser um fator chave para a execução de uma atividade.

O entendimento de como foi realizada a construção do conhecimento, quais foram os pontos de discussão e atenção durante o seu processo de concepção, quais foram as decisões tomadas e dificuldades do processo como um todo, podem ser fundamentais. O conhecimento adquirido em outras atividades pode ajudar a resolver problemas semelhantes, além de reutilizar melhores práticas, reduzir o trabalho e agilizar a execução de tarefas.

Em (NUNES, 2007) foi proposto um modelo conceitual de Gestão de Conhecimento que é integrado ao ambiente de trabalho, de forma a tratar as situações e especificidades ocorridas durante a execução de atividades, em um processo de

trabalho. Dado o alto nível de abstração do modelo, não existe definição arquitetural para suportar a sua implementação. A partir desse ponto, surge a necessidade de se investigar como esse modelo poderia ser suportado por uma arquitetura, que possa oferecer o suporte ao reuso de conhecimento relacionado com as atividades.

Assim, o problema tratado nesta dissertação é: como deve ser definida uma arquitetura para suportar a gestão de conhecimento em processos de trabalho?

1.3 Enfoque da Solução

Neste trabalho, contexto é definido como uma informação útil para a execução das atividades e para as interações que ocorrem em um processo de trabalho. Pesquisas indicam que a captura, o armazenamento e a recuperação do conhecimento devem ser considerados juntamente com o conhecimento contextual das atividades. O conhecimento contextual pode ser encontrado em experiências, artefatos, atividades, condições, fatos e situações que acontecem durante a execução de um trabalho (BRÉZILLON, 1999).

O conhecimento contextualizado pode adicionar informações importantes para apoiar a execução de atividades, além de descrever artefatos ou explicar por que decisões foram tomadas. Quando utilizado em um modelo de gestão de conhecimento, também pode facilitar a busca por atividades correlatas ao mesmo contexto na memória organizacional.

Pesquisas propuseram um modelo para o ciclo de criação, armazenamento e reutilização de conhecimento contextual que contemplasse as questões de representação, acesso, armazenamento, comparação e recuperação no cenário da execução de uma atividade em um processo de trabalho (NUNES, 2007).

Neste trabalho, exploramos como uma arquitetura baseada em contexto de atividades pode oferecer suporte ao mesmo problema tratado pelo modelo de gestão de conhecimento proposto no trabalho de NUNES (2007). Dado que a definição de uma arquitetura baseada em contexto representa ainda um desafio e um problema em aberto, diversos trabalhos na literatura realizaram propostas que buscaram solucionar as principais questões dos seus domínios de interesse. Recentemente, foi realizada uma comparação entre as diversas abordagens disponíveis, demonstrando as características de cada abordagem (BALDAUF *et al.*, 2007). Como nenhuma das arquiteturas analisadas é diretamente voltada para tratar a gestão de conhecimento em processos de negócios, este trabalho procura propor uma arquitetura, com base nas que foram analisadas, que ofereça as características necessárias para suportar a solução do problema.

1.4 Objetivos

O objetivo geral desta dissertação é definir uma arquitetura baseada em contexto que permita processar, raciocinar, compartilhar e recuperar o conhecimento contextual de atividades intrínsecas a um processo de trabalho.

Para alcançar este objetivo, os seguintes objetivos específicos foram definidos:

- Avaliar as arquiteturas para sistemas sensíveis ao contexto propostas na literatura até o momento, extrair um padrão existente entre elas e tê-lo como base para a definição da arquitetura proposta;
- Propor uma arquitetura reutilizável e instanciável que permita às aplicações sensíveis ao contexto de atividades receberem conhecimento contextual, processá-lo, raciocinar e responder com conhecimento contextual similar ao recebido; tal arquitetura deve seguir o padrão

definido e reutilizar o máximo de API já existentes na literatura para minimizar o esforço de implementação de captura e armazenamento de contexto em geral;

- Instanciar a arquitetura a um domínio para sua avaliação, para isto é necessário:
  - ✓ Modelar o processo do domínio escolhido;
  - ✓ Estender e instanciar a ontologia definida por NUNES (2007) para o processo modelado;
  - ✓ Construir regras de inferência sobre o conhecimento contextual e de domínio como prova de conceito;
  - ✓ Avaliar a arquitetura com a sua instanciação, implementação, execução e simulação.

1.5 Delimitação da Abrangência do Trabalho

Esta seção visa delimitar a abrangência do trabalho, descrevendo algumas características que demarcam seu foco.

A definição de diferentes regras e estratégias de inferência sobre o conhecimento contextual de atividades que poderiam ser utilizadas por diferentes domínios está fora do alcance deste trabalho, dado que requer uma pesquisa profunda e validação nos diferentes domínios, quer seja por estudos de caso em ambientes organizacionais ou com experimentação utilizando dados consistentes.

A mesma justificativa é válida para a construção de mecanismos específicos para captura e apresentação de contexto. A arquitetura proposta tem como objetivo prover a integração de diversos mecanismos e componentes, porém dado que algumas camadas e componentes da arquitetura necessitariam de avaliações em aplicações dentro

de um contexto organizacional, a pesquisa foi delimitada para o núcleo central da arquitetura, e por isso nem todas as partes foram consideradas no processo de instanciação e avaliação. Além disso, NUNES (2007) já definiu padrões para especificação de tais mecanismos, que podem ser aplicados independentemente do domínio no qual o ambiente proposto será instanciado.

1.6 Organização da Dissertação

Este trabalho foi dividido em seis capítulos, sendo o primeiro a Introdução.

No **Capítulo 2** são apresentados os conceitos de Gestão de Conhecimento e Contexto, a questão da importância de modelos baseados em contexto para a Gestão de Conhecimento, e por fim o modelo referência utilizado neste trabalho.

No **Capítulo 3** discutem-se as arquiteturas para sistemas sensíveis a contexto já propostas na literatura, as suas vantagens e desvantagens, assim como qual delas foi selecionada como base para a arquitetura proposta nesta dissertação. É feita uma análise crítica do que falta para essa arquitetura suportar o modelo baseado em contexto para a Gestão de Conhecimento.

No **Capítulo 4** é apresentada a arquitetura baseada em contexto de atividades de processos de trabalho proposta, contribuição principal deste trabalho.

No **Capítulo 5** é apresentada a avaliação e instanciação da arquitetura utilizando o domínio de um processo de planejamento de projetos de software e suas possíveis informações de contexto.

Finalmente, o **Capítulo 6** faz um resumo do trabalho e realiza uma análise das contribuições, conclusões, assim como apresenta possíveis trabalhos futuros de pesquisa.

2 Gestão de Conhecimento e Contexto

Antes de avaliar as arquiteturas para sistemas sensíveis ao contexto já propostas na literatura até o momento, é importante entender as duas áreas de pesquisa bastante abrangentes: gestão de conhecimento e contexto. Nas seções deste capítulo são apresentados os conceitos que fundamentam estas áreas assim como a questão da importância de modelos baseados em contexto para a Gestão de Conhecimento e, por fim, o modelo de referência utilizado neste trabalho.

2.1 Gestão de Conhecimento

2.1.1 Conceitos

Segundo NONAKA (1991), o conhecimento é o que aumenta a capacidade de um indivíduo para tomar uma ação, de acordo com suas habilidades, competências e atividades intelectuais. Conhecimento também é informação retida na mente de um indivíduo. É personalizada ou relacionada a fatos, conceitos, interpretações, idéias, observações e julgamentos (ALAVI e LEIDNER, 1999). Logo, não necessariamente, existe uma coleção de informações disponíveis para acesso por qualquer um. Igualmente, há diferentes tipos de conhecimento: individual, parcialmente compartilhado e totalmente compartilhado (MAHÉ e RIEU, 1998), e este deve ser comunicado de forma compreensível, para que o conhecimento seja conhecido por todos.

Em um contexto organizacional, o capital intelectual de uma organização consiste em bens tangíveis e não tangíveis. Os bens tangíveis estão relacionados ao conhecimento explícito. Ele é documentado, e pode variar para os diferentes domínios de aplicação e em geral inclui manuais, correspondência de/e para clientes ou vendedores, inteligência competitiva, patentes, licenças, e conhecimento derivado de processos de trabalho, tais como artefatos de propostas e projetos. Já os bens não tangíveis estão relacionados ao conhecimento tácito e não documentado, isto é, habilidades, experiências, e conhecimento das pessoas de uma organização (NONAKA e TAKEUCHI, 1991).

Neste cenário, consideramos que Gestão de Conhecimento (GC) é a disciplina que administra o conjunto de procedimentos, práticas e ferramentas que visam capturar, armazenar e disseminar o conhecimento dentre os funcionários da organização, aproveitando os recursos tecnológicos existentes (PERRET, 2004). Os objetivos da GC em uma organização são promover o crescimento do conhecimento, a sua comunicação e preservação na organização (STEELS, 1993). Isto engloba gerenciar os recursos do conhecimento para facilitar acesso e reuso do mesmo (O'LEARY, 1998).

Embora sua função claramente signifique administrar o conhecimento, na prática o conjunto de atividades que compreende a GC é tão complexo que se encontram, na literatura, diversas definições (MALHORTA, 1998, ALAVI e LEIDNER, 1999, TERRA e GORDON, 2002) que, em sua maioria, apontam a disseminação do conhecimento como objetivo principal, o que engloba as atividades básicas, propostas por ABECKER e DECKER (1999), de identificação, aquisição, desenvolvimento, disseminação, utilização e preservação do conhecimento.

Os maiores desafios da GC não estão na criação, e sim na captura e integração do conhecimento (GRANT, 1996). De fato, o conhecimento tem pouco valor

organizacional se não for compartilhado.

Historicamente, a criação e transferência do conhecimento têm ocorrido através de interações entre as pessoas, *mentoring*, *job rotation* e capacitação de pessoal. Entretanto, com a globalização, os mercados e organizações estão se movendo cada vez mais para o plano virtual, o que tem tornado a forma tradicional de trabalho lenta e pouco efetiva. Isto tem forçado as organizações a buscarem cada vez mais as interações virtuais e eletrônicas, de forma a agilizar seus processos internos. Como resultado, companhias começaram a implementar Sistemas de Gerenciamento de Conhecimento (Knowledge Management Systems - KMS), projetados especificamente para facilitar a codificação, coleta, integração e disseminação do conhecimento organizacional (ALAVI, 1997, BARTLETT, 1996, SENSIPER, 1997).

Por outro lado, como BROWN e DUGUID (1991) notaram, o conhecimento não vai necessariamente circular livremente por toda empresa apenas porque a tecnologia está disponível para este fim. Na ausência de uma estratégia explícita para melhor criar e integrar o conhecimento na organização, sistemas de software que facilitam a comunicação e compartilhamento de informação, na melhor das hipóteses, terão somente um efeito randômico (ALAVI e LEIDNER, 1999).

Desta forma, o ciclo de transformação do conhecimento serve como base para dizer como o conhecimento é transformado, isto é, capturado, processado, armazenado e reutilizado dentro da organização. NONAKA (1991) propôs um modelo de transformação chamado "espiral do conhecimento" que é fortemente baseado em dois conceitos: conhecimento tácito e conhecimento explícito. Segundo o autor, a combinação desses dois conceitos sugere quatro padrões básicos para criação do conhecimento em uma organização:

i) Socialização: significa o processo de criar conhecimento através da conversão do conhecimento tácito de uma entidade (indivíduo, grupo ou organização) para outra entidade;

ii) Combinação: é o processo de criar novo conhecimento explícito a partir de conhecimentos explícitos existentes;

iii) Externalização: é a conversão do conhecimento tácito para o explícito;

iv) Internalização: é a conversão do conhecimento explícito para o tácito.

O conhecimento organizacional é então criado pelas interações entre esses quatros processos de conversão e através da transferência do conhecimento tácito e explícito a partir de um indivíduo ou grupo para os níveis organizacionais.

Seguindo os padrões, a criação do conhecimento começa com a socialização. Esta interação facilita o compartilhamento de experiências e perspectivas dos membros das equipes. A seguir, a externalização é acionada por sucessivas rodadas de "diálogos". Através destes diálogos, entidades articulam seu conhecimento tácito para que os outros possam entender. O conhecimento que é criado através da externalização pode ser combinado com o conhecimento existente para posteriormente ser refinado e estender a base de conhecimento ou memória organizacional. Este processo ocorre iterativamente, com o conhecimento crescendo e evoluindo de forma concreta. E através desta experimentação de aprendizado é que a internalização acontece.

2.1.2 Memória Organizacional

Na medida em que o conhecimento vai sendo transformado e armazenado, a memória organizacional cresce. Neste cenário, a Memória Organizacional (MO) é o conceito central utilizado para descrever o repositório utilizado na administração do conhecimento da organização. É importante, entretanto, notar que existem diversas

propostas de arquitetura para MO, através de estudos de técnicas em particular, tal como processamento de conhecimento (KUHN e ABECKER, 1997). Claramente, este problema complexo tem aspectos organizacionais para serem abordados e aspectos técnicos para serem resolvidos. Cientistas da Computação preocupados com o uso de TI para gestão de conhecimento, em geral, tendem a ignorar os requisitos específicos e as restrições para uma gestão de conhecimento bem sucedida na prática. Enquanto que especialistas de gestão de conhecimento, em geral, tratam os aspectos computacionais como meras ferramentas de apoio. Entretanto, para que uma MO seja criada e mantida é necessário uma abordagem multidisciplinar.

Dado que a MO pode estar retida na cultura organizacional, nas transformações organizacionais, nos arquivos externos, em manuais corporativos, nas bases de dados e nos indivíduos (ACKERMAN, 1994), ela integra técnicas básicas em um sistema computacional que – dentro das atividades de negócio da organização – continuamente coleta, atualiza, e estrutura conhecimento e informação. Além disso, ela os provê em diferentes atividades operacionais de forma sensível ao contexto, intencionada e ativa (PARRINI, 2002, ABECKER e DECKER, 1999).

Perceber as principais necessidades para construir uma MO não é uma tarefa simples. Projetistas de MO precisam aprender o máximo possível sobre quem são os usuários, quais são as tarefas que eles realizam, e em quais situações. Além disso, quais os tipos de conhecimento que precisam ser memorizados e recuperados (para a realização das tarefas), quais ferramentas usar, etc.

As organizações costumam falhar na captura de registros de processos pelos quais estes são criados e com isso, rotineiramente tendem a esquecer o que foi feito e porque foi feito. Estas organizações têm uma capacidade prejudicada para aprender, devido à inabilidade de representar aspectos críticos sobre o que elas sabem.

A falta de documentação acerca das circunstâncias através das quais os artefatos foram construídos faz com que as pessoas não sejam capazes de entender o *rationale*[1], por trás de sua criação e como conseqüência, são sobrecarregadas de informações que não contribuem para o seu trabalho, além de não confiarem nos dados que estão sendo captados.

Nos trabalhos de (CONKLIN, 1992, CHOI *et al*., 2000, MAURER e HOLZ, 1999, LAI e FAN, 2002, BERZTISS e SYSLAB, 2000) são propostas a integração de processos e GC. A idéia é que a GC serve de apoio à disseminação do conhecimento tácito entre as pessoas em um ambiente organizacional. Por exemplo, ter acesso à informação pertinente pode ser um fator crítico para execução de uma atividade em um processo de trabalho. Em adição a isto, o conhecimento organizacional que é criado e desenvolvido por cada profissional envolvido na execução de suas atividades pode ajudar a organização a aumentar seus padrões de qualidade.

Desta forma, além de armazenar e disseminar os artefatos produzidos durante a execução de uma atividade em um processo de trabalho, é necessário associar a estes artefatos gerados e ao trabalho realizado, o *rationale* para a sua elaboração e execução, enriquecendo assim a MO que será mais bem aproveitada pelos profissionais.

A função da organização, neste sentido, é fornecer o meio apropriado para facilitar as atividades de um grupo, para criar e acumular conhecimento em nível organizacional. Uma das grandes questões está no tipo, na forma e dificuldades encontradas para gestão destes elementos do processo de trabalho que podem se transformar em uma tarefa onerosa.

[1] O termo rationale está em inglês e tem sua origem no latim. Ele significa o processo de raciocínio desenvolvido para se fundamentar algo. Por não ter uma tradução direta ao português e manter a expressividade da palavra, o termo foi mantido em inglês.

2.2 Conhecimento Contextual

Pesquisas (BRÉZILLON, 1999, ARAUJO e BRÉZILLON, 2005) indicam que a captura, o gerenciamento e a disponibilização do conhecimento devem ser considerados juntamente com o contexto ao qual esse conhecimento foi capturado, gerenciado e disponibilizado. Não é objetivo deste trabalho explorar as diferentes definições de contexto existentes na literatura e que foram descritas em (NUNES, 2007), por exemplo. Porém, é importante descrever qual a definição de contexto utilizada neste trabalho. Utiliza-se como base a definição encontrada em (BRÉZILLON, 2002) que diz:

"Contexto pode ser definido como as circunstâncias sobre as quais um evento ou ação podem ocorrer".

Mais além, define-se como: uma informação útil para a execução das atividades e para as interações que ocorrem em um processo de trabalho. O contexto oferece subsídios para que os profissionais da organização percebam e compreendam melhor as atividades executadas. As pessoas utilizam informações de contexto na realização de trabalhos sem se dar conta disto.

Conhecimento contextual pode ser definido como qualquer conhecimento relevante e pessoal, envolvido em uma atividade. Parte deste conhecimento contextual é proceduralizado para permitir cooperação e/ou ação, e os resultados são contextos proceduralizados[2] que podem ser elicitados e, por conseguinte, se tornarem explícitos pelos métodos usuais da engenharia do conhecimento (BRÉZILLON e POMEROL, 2001).

A importância do conhecimento contextual se deve ao fato deste possuir a capacidade de propiciar práticas mais eficientes nos negócios, promovendo a inovação e interação do grupo e da organização. Além disso, a gestão das informações de contexto,

[2] Contexto proceduralizado é a instanciação de uma parte do conhecimento contextual.

quando implementada em um ambiente computacional, pode atuar como um filtro que define, em um determinado momento, qual conhecimento deve ser levado em consideração no auxílio à execução de tarefas. Por isto, capturar e disponibilizar as circunstâncias que resultaram na tomada de uma decisão ou na criação de um artefato permitirão uma reutilização com maior qualidade e utilidade do conhecimento gerado na própria organização em situações e contextos futuros.

Considerando a gestão de conhecimento contextual, observa-se que propostas de desenvolvimento de ambientes computacionais sensíveis a contexto é um tópico de recentes pesquisas em diversas áreas (VIEIRA, 2008, CHEN, 2004, DEY *et al*., 2001, GU *et al.*, 2005). A inclusão desses mecanismos nos ambientes de trabalho muitas vezes não segue técnicas, regras ou métodos a fim de facilitar e padronizar o seu desenvolvimento.

Em particular, o contexto em um processo de trabalho tem uma natureza dinâmica pelos eventos que surgem e novas decisões que são tomadas, alterando o curso de sua execução. Se informações de contexto não são associadas às atividades executadas e artefatos gerados, a MO terá um imenso conjunto de documentos com pouca ou nenhuma conexão entre eles e acabará sendo ignorada como um recurso de informações.

2.3 Modelo de Referência para GC baseado em Contexto

Dado que o conhecimento contextual é relevante dentro do ciclo de criação e transformação de conhecimento, ele não pode ser tratado de forma independente, nem ignorado. Um modelo de GC baseado em contexto é necessário para demonstrar como a Gestão de Contexto pode ser acoplada ao ambiente de trabalho organizacional. Este modelo deve possuir uma visão geral que se inicia desde a captura das informações

contextuais, segue na sua associação à atividade em execução e posterior armazenamento na MO. Após o armazenamento, pode-se então comparar o contexto da atividade atual com aqueles armazenados na MO, através de mecanismos de inferência, que podem atuar como filtros na seleção dos contextos que podem ser úteis para execução de futuras atividades.

O modelo para gestão de conhecimento baseado em contexto proposto por NUNES (2007) trata do ciclo (Figura 2.1) de criação, armazenamento e reutilização de conhecimento contextual, o que abrange as questões de representação do conhecimento, captura, armazenamento, comparação e apresentação, no cenário da execução de uma atividade, em um processo de trabalho. Ele visa estabelecer uma memória organizacional com o resultado da execução das atividades dos processos de trabalho, e também o seu contexto, através dos quais esses resultados foram alcançados.

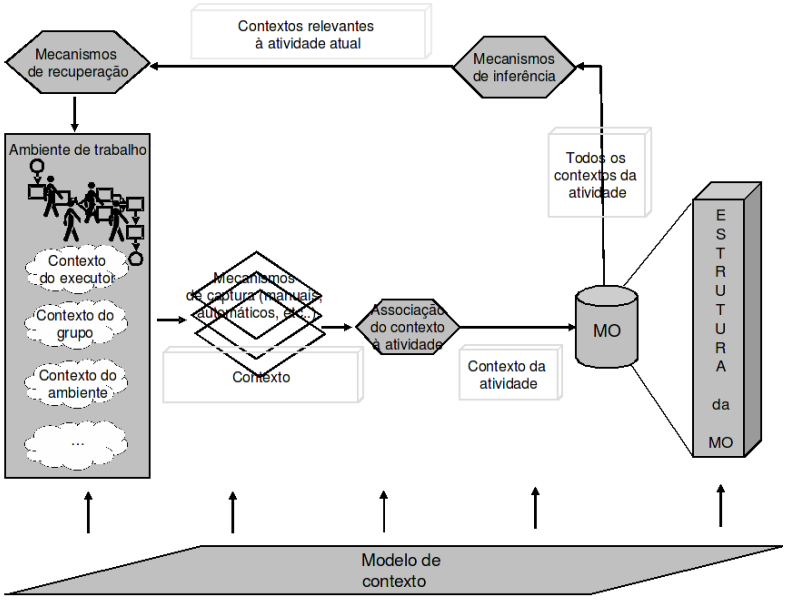

Figura 2.1 – Modelo de GC baseado em Contexto (NUNES, 2007).

Para a fase de captura de contexto, NUNES (2007) elaborou um modelo de representação de contexto formal mostrando os principais elementos que constituem as informações contextuais na execução de uma atividade e o relacionamento entre eles. Desta forma o conhecimento pode ser percebido e entendido.

A MO é organizada em uma estrutura definida que mapeia os contextos existentes para os elementos de conhecimento armazenados na MO. O resultado da captura, associação à atividade e armazenamento das informações contextuais gera o contexto da atividade atual que realimenta a MO (Figura 2.1), introduzindo conhecimentos, novos ou não, na organização que poderão ser consultados quando da execução de atividades semelhantes. Para a formalização da representação de contexto, foi elaborada uma ontologia de contexto focada nas atividades do processo de trabalho.

Ontologia é uma especificação explícita de uma conceitualização (GRUBER, 1993). Na área de Computação e Ciência da Informação, uma ontologia define um conjunto de representações primitivas para modelar um domínio de conhecimento (GRUBER, 2008).

Uma das características de ontologias é que elas são extensíveis a qualquer domínio de conhecimento. Contexto é uma construção dinâmica e está intrínseca ao domínio sobre o qual se está atuando. Além disso, ontologias representam o relacionamento entre as informações provendo maior semântica entre os contextos e potencializando a ação de máquinas de inferência na comparação entre os diversos contextos de uma mesma atividade. A utilização de uma representação formal baseada em ontologias facilita o compartilhamento de contexto entre aplicações, o raciocínio e inferência, além de possibilitar a detecção de inconsistências no conhecimento contextual.

A ontologia de contexto proposta por NUNES (2007) é centrada no conceito de Atividade e pretende ser genérica, porém extensível a qualquer domínio de conhecimento. Utilizando uma ontologia de contexto, bases de conhecimento podem ser construídas descrevendo circunstâncias específicas de um processo de trabalho. Ontologias podem ajudar a descrever fatos, crenças, hipóteses, discussões e outras

informações sobre uma atividade que está sendo executada.

As questões principais contempladas nesta ontologia estão relacionadas a: (i) informações de contexto que existem durante a execução de uma atividade como informações sobre o ambiente, tempo, artefatos, entre outros; (ii) informações sobre os indivíduos e o grupo como um todo que executa uma atividade e as interações que ocorrem entre eles; (iii) informações de contexto que explicitem a interação entre os indivíduos e a atividade em execução.

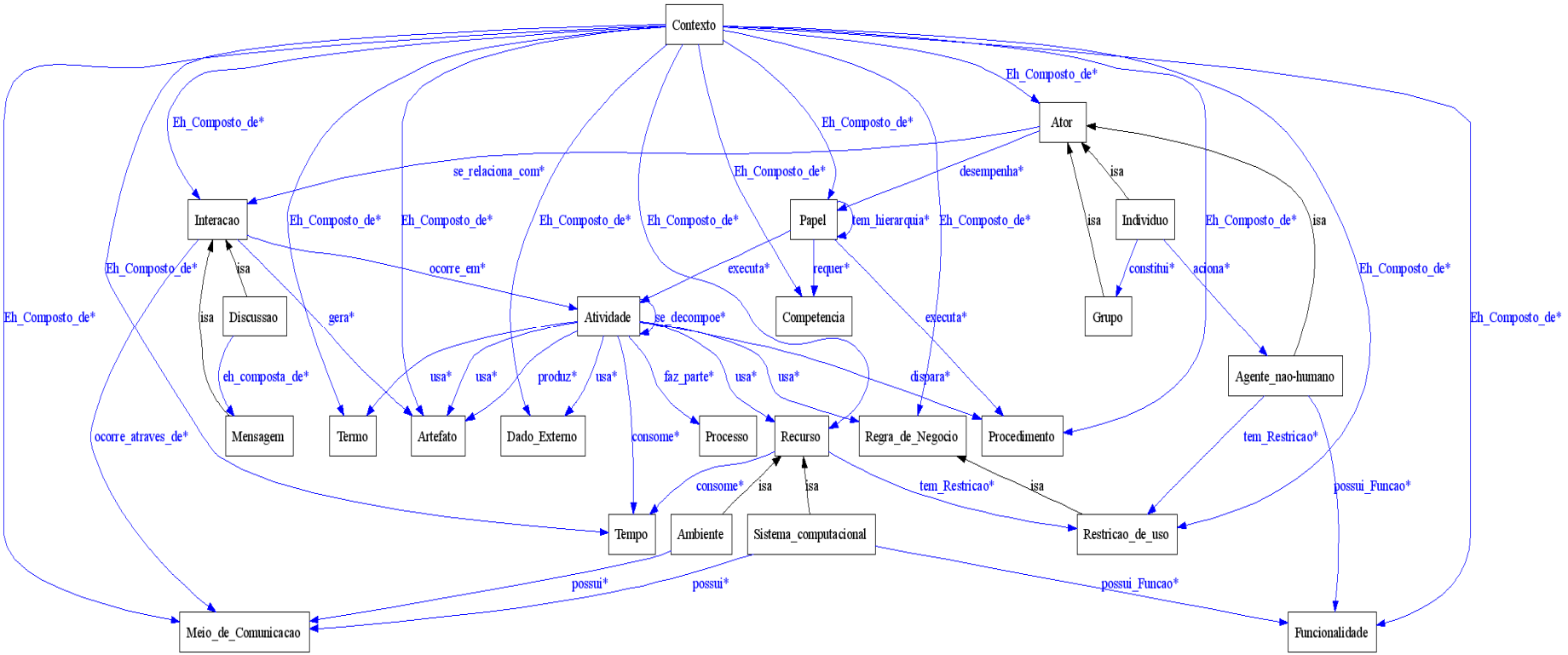

Figura 2.2 – Ontologia de contexto focada em atividade (NUNES, 2007, NUNES, 2006).

A ontologia de contexto é apresentada na Figura 2.2 e considera dois elementos, ou classes principais: Atividade e Contexto. Atividade é a referência no modelo de GC proposto, ou seja, o que se deseja representar é a atividade versus seu contexto. Atividade representa o foco e contexto representa o conjunto de informações que envolvem a sua execução.

A classe Atividade representa um conjunto de ações orientadas a um ou mais objetivos que visam atingir as metas do processo, que consomem e produzem informação e que necessitam de um ou mais agentes para executá-la. A classe Processo representa o conjunto de atividades que visam atingir uma meta. Uma atividade faz parte de um processo. A classe Contexto é composta de todas as informações ou classes de contexto que se relacionam diretamente com a atividade e descreve o relacionamento existente entre elas.

O detalhamento completo da ontologia pode ser encontrado em NUNES (2007). No entanto, para deixar este trabalho auto-contido, a Tabela 2.1 a seguir descreve as classes e relacionamentos da ontologia que foram utilizados no estudo de caso apresentado no Capitulo 5:

Tabela 2.1 – Descrição das classes e o relacionamento entre elas (NUNES, 2007).

Processo	Conjunto de atividades que visam atingir uma meta. Uma Atividade faz parte de um Processo.
Atividade	Conjunto de ações orientadas a um ou mais objetivos, que consome e produz informação e artefatos e que necessita de agentes para executá-la.
Recurso	Elemento relacionado a plataformas computacionais, equipamentos mecânicos, materiais necessários, ambientes de trabalho e que tem Restrição de Uso. Pode ser especializado em subclasses específicas ao domínio de aplicação como as classes Ambiente e Sistema Computacional.

Ambiente	Especialização da classe Recurso. Representa uma localidade da organização (física ou conceitual) que não é caracterizada apenas por suas coordenadas, mas também por características como, nome, função, capacidade. Possui Meios de Comunicação possíveis de serem utilizados. É o meio onde as interações ocorrem.
Sistema Computacional	Especialização da classe Recurso. Representa aplicações de software utilizadas como apoio na execução das atividades.
Funcionalidade	Função específica exercida por Agentes não Humanos e Sistemas Computacionais.
Restrição de Uso	É uma Regra de Negócio que representa regras de restrição apresentadas por Recursos e Agente não Humanos.
Meio de Comunicação	Forma de comunicação que pode existir durante uma Interação e que é provida pelo Ambiente ou por um Sistema Computacional.
Termo	Conceito estabelecido pela Organização e pelo domínio de aplicação.
Dado Externo	Informação externa à organização ou ao processo de trabalho que está sendo executado.
Papel	Função que é desempenhada por uma ou mais pessoas e que é atribuída a cada atividade do processo. Papéis têm hierarquia entre si e requerem algum tipo de Competência. Papéis ainda executam Procedimentos dentro de atividades.
Competência	Aptidão necessária que um papel deve possuir para ser capaz de executar uma atividade de forma correta.
Ator	Identifica os profissionais através de suas características e dados pessoais e do relacionamento entre eles. São especializados em Indivíduo, Grupo e Agente não humano que participam na execução de uma atividade.
Agente não Humano	Especialização da classe Ator que representa um sistema ou componente computacional capaz de executar uma atividade ou parte dela automaticamente ou mediante o acionamento de um Indivíduo.

Indivíduo	Especialização da classe Ator que representa pessoas que desempenha um Papel na execução de uma Atividade isoladamente.
Grupo	Especialização da classe Ator que se constitui de um ou mais indivíduos organizados em uma comunidade.
Competência	Habilidade específica requerida por um Papel para executar uma atividade de forma apropriada.
Regra de Negócio	Informação que define ou restringe alguns aspectos do negócio. Pode estar diretamente ligada à organização como um todo ou pode estar ligada a um domínio específico de atuação. Sua intenção é garantir a estrutura do negócio ou influenciar o comportamento do mesmo. Pode corresponder a: restrições internas como padrão de eficiência e atendimento às metas ou a restrições externas como leis e regulamentações.
Tempo	Contém os prazos e tempos reais de execução de uma ação, do uso de algum recurso e da atividade como um todo.
Artefato	Produto concreto resultante da execução de uma atividade que pode servir como insumo para outras atividades.
Procedimento	Norma ou padrão que é executada por atores (de acordo com o papel associado a eles no momento) ou disparado pela própria atividade através de alguma ação associada a eles.
Interação	Representa o processo de comunicação que acontece entre atores ao executar uma atividade, que ocorrem através de algum Meio de Comunicação e que também gera Artefatos. Estes dois relacionamentos são motivados pela necessidade de capturar as relações existentes entre os atores e os artefatos que são produzidos e diminuir o *gap* existente entre os objetivos que devem ser atingidos e como eles foram alcançados.
Discussão	Especialização da classe Interação que representa as discussões ocorridas entre os Atores que participam da execução do Processo.
Mensagem	Especialização da classe Interação que representa o conteúdo das mensagens trocadas durante as interações que acontecem. Uma discussão é composta de mensagens.

Sendo a classe Atividade o foco desta ontologia, seu relacionamento com as demais classes representa as informações e ligações chave para este trabalho. Uma atividade além de fazer parte de um processo, pode se decompor em outras Atividades o que permite criar uma hierarquia entre elas, além daquela já existente no processo. Uma atividade faz uso de elementos contextuais de termos, dados externos que auxiliem ou sejam imprescindíveis para sua execução, recursos que sejam computacionais, de materiais, de ambiente ou outros, e, além disso, pode utilizar também Regras de Negócio existentes na organização ou que a influenciem.

Uma atividade produz e usa como insumo Artefatos construídos em atividades anteriores ou provenientes da própria atividade e consome tempo ao ser realizada. Uma atividade ainda dispara procedimentos padrões existentes na organização. Interações ocorrem no decorrer das atividades onde atores, desempenhando papéis, estão participando da execução da atividade e se relacionam.

Com base na ontologia de contexto, tipos de mecanismos de captura de contexto foram propostos, assim como padrões para sua implementação. NUNES (2007) apresentou uma matriz (Tabela 2.2) que relaciona os grupos de classes a cada tipo de mecanismo de captura.

Tabela 2.2 – Grupos de Classes vs. Mecanismos de Captura (NUNES, 2007).

	Mundo físico	Atividade	Pessoas e Agentes	Interação
Humanos		×	×	
Agentes automáticos	×	×	×	×
Sensores físicos	×			×
Agentes semi-automáticos		×	×	×

Os mecanismos humanos possuem interfaces para captura de informação que são apresentadas ao profissional com o objetivo dele inserir o que julgar necessário e relevante. Agentes automáticos são componentes lógicos que são construídos para monitorar o ambiente, tanto física quanto operacionalmente. Também se inserem aqui mecanismos capazes de fazer extração automática de informação contextual existente nos artefatos elaborados pelos profissionais. O mecanismo baseado em sensores físicos são, em sua maioria, componentes eletromecânicos que monitoram e captam informações sobre o ambiente físico em que os profissionais atuam e a interação que ocorre entre eles. E, por fim, agentes semi-automáticos são componentes lógicos que requerem certo nível de interação entre o profissional e o sistema, ou seja, dependem parcialmente de algum tipo de comando externo para serem acionados.

De acordo com a matriz, mecanismos humanos são principalmente utilizados para capturar informações contextuais relacionadas à atividade e aos agentes que participam na execução da atividade. O registro humano destes conhecimentos tenta capturar os passos seguidos pelos agentes e também o seu ponto de vista em relação às dificuldades, artefatos produzidos e conhecimento adquirido e utilizado neste meio.

Os agentes automáticos são utilizados para capturar informações contextuais de qualquer um dos grupos de informação. São componentes ou módulos de software que buscam também na MO informações já existentes sobre os profissionais e sobre as atividades.

Os sensores físicos permitem a captura: de informações de contextos referentes ao mundo físico; e as interações, que ocorrem entre os profissionais que se encontram no mesmo ambiente de trabalho. Em sua maioria, as informações capturadas devem ser tratadas por serem armazenadas em mídias complexas e de difícil busca por informações.

Finalmente, os agentes semi-automáticos podem ser utilizados para capturar informações relacionadas às atividades, as pessoas e agentes e as interações. São softwares que a partir de regras criadas pela própria organização ou regras que surgem do aprendizado proveniente das execuções de uma determinada atividade, são capazes de capturar informações de contexto que possam estar relacionadas à situação, mas que necessitam ser validadas ou acionadas opcionalmente pelo profissional em exercício.

Quatro padrões para especificação e inserção destes mecanismos no ambiente de trabalho foram elaborados por (NUNES, 2007, NUNES *et al*., 2007) cobrindo as possíveis situações existentes:

(i) Interromper para registrar, isto é, atividades podem ou não ser apoiadas por software. É necessário coletar o contexto manualmente através de mecanismos humanos;

(ii) Registrar sem interromper, ou seja, atividades são necessariamente apoiadas por software ou groupware. O contexto está disponível na MO e é automaticamente capturado;

(iii) Registrar através de sensor físico, o que significa que atividades, ou parte delas, usam sensores físicos que capturam automaticamente o contexto sobre o ambiente;

(iv) Acionar registro, ou seja, atividades são necessariamente apoiadas por software ou groupware. O contexto é capturado automaticamente mediante a permissão de um ou mais executores.

2.4 Requisitos Arquiteturais

A criação e o uso do conhecimento ocorrem de forma distribuída em uma organização por natureza. Apesar de algumas organizações possuírem limites físicos, o ambiente geralmente é aberto e distribuído entre diversos departamentos, unidades e localidades.

As contínuas interações, que ocorrem no decorrer da execução dos processos organizacionais, tem como necessidade a integração entre diversas fontes de informações. A heterogeneidade e diversidade de informações disponíveis dificultam a tarefa da integração. Assim, a solução de um sistema ou conjunto de aplicações para gestão do conhecimento, que atenda os diversos processos de uma organização, tende para um sistema complexo e distribuído, com suporte para um ambiente aberto e integrado.

Uma abordagem natural para a construção de softwares complexos, em ambientes distribuídos, é a abordagem orientada a agentes, pois estes oferecem: autonomia; pró-atividade; comportamento inteligente; controle descentralizado; mobilidade, aprendizado e adaptação (WOOLDRIDGE e JEENINGS, 1995).

Com base no modelo proposto em NUNES (2007), os seguintes requisitos podem ser extraídos: suportar a troca de representações de contexto utilizando ontologias; fornecer meios para captura de contexto de atividades e envio dessas informações para uma estrutura de MO; armazenar informações de contexto de atividades em uma estrutura de MO que torne possível consulta e recuperação; suportar recuperação e inferência de contextos de atividades a partir da MO; suportar apresentação de contextos de atividades existentes.

Neste trabalho, procura-se identificar uma arquitetura genérica para direcionar o suporte para gestão do contexto, bem como uma arquitetura base de infra-estrutura para extensão e alinhamento com a arquitetura genérica.

A partir dos requisitos identificados, a escolha da arquitetura base deve ser influenciada pelos seguintes critérios: suporte para a arquitetura genérica escolhida; suporte aos mecanismos de *reasoning*; orientada a agentes; uso de ontologias como forma de representação do conhecimento contextual.

2.5 Considerações Finais do Capitulo

Neste capítulo foram apresentados os conceitos fundamentais de Gestão de Conhecimento e Contexto, bem como foram apresentadas às contribuições de NUNES (2007) que serviram de base para o desenvolvimento deste trabalho.

Como visto, o conhecimento organizacional é de fundamental importância para que as organizações busquem a inovação e possam competir em mercados cada vez mais globalizados. Na busca pelo acúmulo de conhecimento e na construção da memória organizacional, não é suficiente armazenar e disseminar artefatos. Visto que a correta associação dos artefatos gerados com o raciocínio utilizado para a sua elaboração e execução pode enriquecer de forma substancial a memória organizacional.

Também vimos que o raciocínio pode ser representado como Contexto, ou seja, como as informações que caracterizam a execução de uma atividade em um determinado processo de trabalho. Neste caso, a Gestão do Contexto se torna peça importante para a Gestão de Conhecimento e pode contribuir para o melhor entendimento das situações ocorridas e, por conseguinte, para a reutilização deste conhecimento.

Para suportar a Gestão de Contexto, NUNES (2007) propôs uma forma de representação de contexto baseada em ontologias e um modelo de GC baseado em Contexto. Seu modelo prevê captura, representação, armazenamento, recuperação e apresentação de conhecimento contextual em uma estrutura de memória organizacional. No entanto, o modelo proposto é conceitual e possui um alto nível de abstração, sem se preocupar com a questão arquitetural e aspectos de implementação de um ambiente que possa oferecer suporte à Gestão de Contexto.

Este trabalho busca preencher essa lacuna, assumindo que: o conhecimento contextual foi capturado utilizando-se um dos padrões propostos; deve existir uma

arquitetura que ofereça suporte para associação do conhecimento contextual com as atividades de um processo; deve existir suporte para persistência do conhecimento contextual em uma estrutura de MO; deve existir suporte para a inferência e recuperação de conhecimento contextual.

Para definir uma proposta de arquitetura, este trabalho utiliza como base as pesquisas já realizadas na literatura sobre arquiteturas para sistemas sensíveis a contexto, que serão apresentadas no próximo capítulo.

3 Arquiteturas para Sistemas Sensíveis a Contexto

Neste capítulo, é apresentado o estado da arte de arquiteturas para sistemas sensíveis a contexto existentes na literatura. Primeiramente, conceitos de computação sensível ao contexto são apresentados, seguidos das principais características de arquiteturas para sistemas sensíveis ao contexto. Também são apresentadas as abordagens de modelagem de contexto. Posteriormente, discutem-se as abordagens utilizadas em cada arquitetura citada, assim como suas vantagens e desvantagens. Por fim, são apresentados os critérios para escolha da arquitetura de infra-estrutura base.

3.1 Computação Sensível ao Contexto

Sistemas sensíveis ao contexto são aqueles capazes de adaptar suas operações ao contexto corrente sem intervenção explícita do usuário (BALDAUF, 2007). A informação contextual pode ser coletada de várias formas, tais como o uso de sensores, informações da rede, estado de um dispositivo, perfis de usuários de browser, dentre outros dependendo do tipo de informação de contexto desejada.

A história de sistemas sensíveis ao contexto data de quando WANT *et al*. (1992) apresentaram seu Sistema de Localização de Distintivos Ativos, considerado a primeira aplicação sensível ao contexto e que se tratava de um sistema de localização de pessoas em um ambiente de trabalho. Funcionários usavam distintivos que transmitiam sinais com informação sobre a sua localização para um serviço de localização central, através de uma rede de sensores.

De forma geral, um sistema sensível ao contexto é capaz de perceber e entender a resposta para as seguintes questões: Quem é você, e com quem você está? O que você está fazendo, por que está fazendo e para quê? Se não está fazendo agora, quando estará? Quais são suas motivações e crenças para fazer isto? (SINGH, 2006)

Sensibilidade ao contexto é o estado onde um dispositivo ou um programa de software está consciente do ambiente, isto é, consegue percebê-lo e atuar em funções produtivas automaticamente, sem a necessidade de um controle externo explícito ou notificação.

É importante notar que por não ser necessário um controle explícito, os processos de captura do contexto, armazenamento e processamento devem ocorrer em background de forma que o usuário e suas tarefas principais sejam o foco e não as questões técnicas da sensibilidade ao contexto.

Como se pode perceber, em sistemas sensíveis ao contexto, dispositivos e programas não são mais entidades passivas esperando por instruções ou comandos. Ao invés disto, agem como se estivessem vivos e capazes de realizar comportamento inteligente. Eles são projetados para entenderem uns aos outros, bem como os conceitos do mundo real. Portanto, um dos objetivos de um sistema sensível ao contexto é alcançar este tipo de comportamento autônomo em um software usando informação contextual.

3.2 Características de Arquiteturas de Sistemas Sensíveis ao Contexto

Nesta seção, são apresentadas as principais características de arquiteturas de sistemas sensíveis ao contexto. Mesmo sabendo que algumas abordagens existentes na literatura utilizam a sensibilidade ao contexto para domínios específicos, e que os

conceitos e sistemas utilizados são heterogêneos, existem similaridades entre estas abordagens.

O objetivo desta seção é abstrair as características de arquiteturas de sistemas sensíveis ao contexto necessárias para prover fundamentos para a arquitetura proposta neste trabalho.

Como base de explicação das características, é apresentada na Figura 3.1 a arquitetura conceitual genérica para aplicações sensíveis ao contexto proposta por SCHMOHL e BAUMGARTEN (2008). Consideramos os conceitos apresentados por essa arquitetura os que melhor se aproximam da arquitetura proposta neste trabalho, muito embora o seu trabalho não esteja relacionado com a gestão de contexto de atividades de um processo. De certa forma, isto demonstra a generalidade da sua proposta.

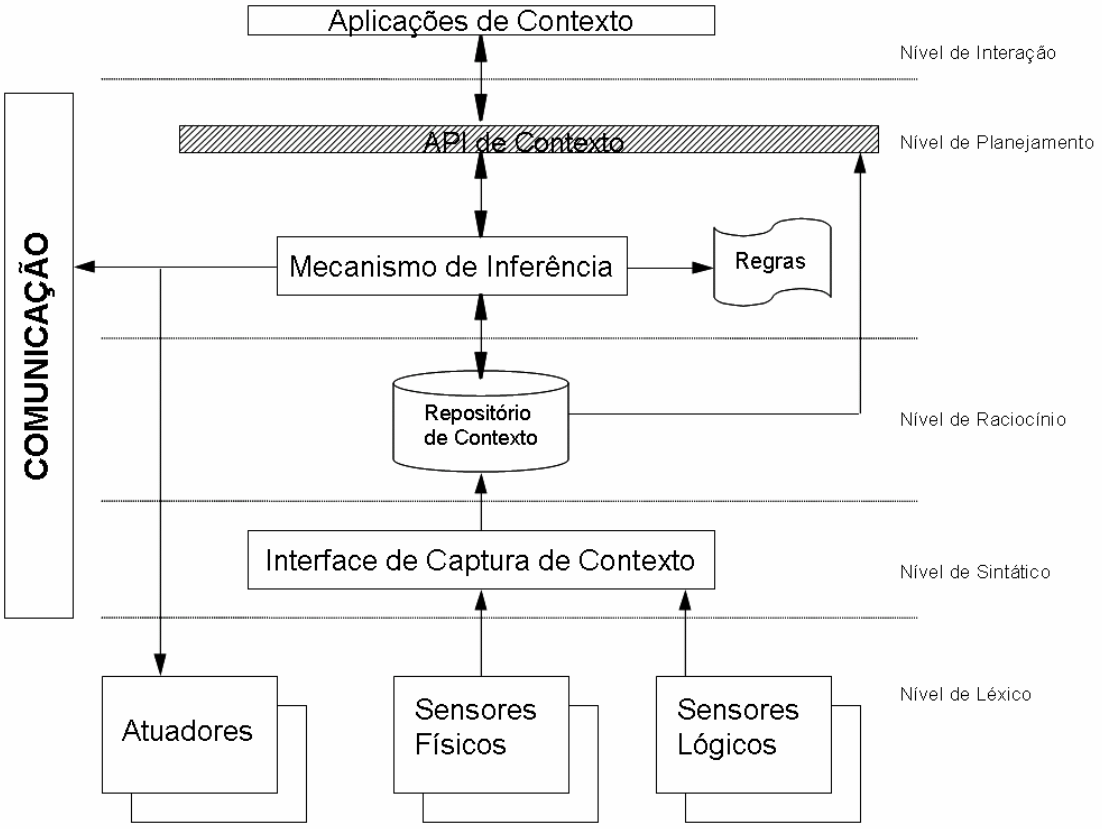

Figura 3.1 – Arquitetura Genérica (SCHMOHL e BAUMGARTEN, 2008).

Em (SCHMOHL e BAUMGARTEN, 2008) foi feita uma análise de diferentes arquiteturas de sistemas sensíveis a contexto apresentadas em várias publicações, tais como (GELLERSEN *et al*., 2002), (JACOB *et al.*, 2007), (BIEGEL e CAHILL, 2004), (CHRISTOPOULOU *et al*., 2005), (ROCHA E ENDLER, 2005), (YAU *et al*., 2002), onde foi possível identificar diversos componentes que são compartilhados pela maioria das abordagens.

SCHMOHL e BAUMGARTEN (2008) se baseou no trabalho de CHRISTOPOULOU *et al*. (2005) para descrever os diversos níveis da arquitetura: Léxico, Sintático, Raciocínio, Planejamento e Interação. No nível Léxico são examinados como sinais do ambiente, que são capturados por sensores e podem ser traduzidos para eventos de contexto. A princípio, qualquer sensor pode ser considerado como uma fonte de contexto. O objetivo deste nível é que o sensor possa ser modelado de forma a abstrair as tecnologias de sensores ou características particulares de um sensor. Os eventos de contexto devem ser analisados ou combinados no próximo nível, que é o Sintático ou de Representação.

No nível Sintático, o objetivo é traduzir os eventos de contexto em informação de contexto relevante, através da associação dos valores obtidos pelos sensores com propriedades ou entidades do mundo real, como pessoa, lugar ou atividade, de acordo com uma ontologia previamente definida.

No nível de Raciocínio, o objetivo é prover modelos para o refinamento e organização das informações de contexto em uma representação que possa sofrer processamento e raciocínio. A inferência de contexto também pode ser vista como um problema de classificação e em casos mais simples o *Reasoning* pode ser baseado em: definição de uma ontologia, que pode usar lógica de descrição; ou *Reasoning* definido pelo usuário, utilizando lógica de primeira ordem.

No nível de Planejamento, o objetivo é definir estratégias e ações que possam ser tomadas em resposta para a variação ou mudanças nas informações de contexto. Assim que mudanças são detectadas, reações são planejadas e agendadas.

Já no nível de Interação, o objetivo é prover um modelo para as interações pessoais e coletivas nos ambientes. Um modelo conceitual de alto nível separa a aquisição de contexto (nível léxico) da sua caracterização (nível de representação ou *reasoning*), da sua entrega e reação ao contexto (nível de planejamento) e da sua composição em diversas aplicações sensíveis a contexto.

Após o detalhamento de cada um dos níveis apresentados na Figura 3.1, a seguir temos os componentes abstratos que podem ser identificados e também são compartilhados pela maioria das abordagens:

Sensores de contexto: os sensores podem existir tanto como dispositivos, isto é, hardware percebendo o ambiente, como componentes de software provendo dados de fontes de contexto. Independente de qual tipo seja o sensor, a sua primeira tarefa é a aquisição do dado cru, para posterior refinamento com a informação contextual.

Interface de captura de contexto: dados coletados por sensores, que são componentes de baixo nível, são geralmente incertos e difíceis de interpretar por componentes de alto-nível, que são as aplicações do domínio. Por esta razão, estes dados precisam ser refinados e transformados em estruturas de dados para posterior processamento. Este processo também é chamado de "agregação" ou "composição".

Repositório de contexto: ele armazena o contexto atual e contém as estruturas de dados usadas para representar o modelo do contexto.

Raciocínio de contexto: o componente de raciocínio de contexto é responsável pela inferência do novo contexto baseado na informação contextual atual no repositório de contexto e a nova informação de contexto adquirida através da interface de captura de

contexto. Este processo pode ser colocado em prática por máquinas de inferência baseadas em regras (BIEGEL e CAHILL, 2004). As regras podem ser tanto de escopo local, quanto global. Regras globais impactam o modelo contextual inteiro, enquanto que as regras locais somente alteram entidades locais (CHRISTOPOULOU *et al*., 2005).

API de contexto: a API de contexto provê uma interface para aplicações sensíveis ao contexto efetivamente utilizarem informação contextual (ROCHA E ENDLER, 2005). O uso de tal interface implementa o paradigma comumente aceito de separação da implementação de gerenciamento de contexto, da implementação da lógica da aplicação (CHRISTOPOULOU *et al*., 2005).

Aplicação de contexto: são as aplicações acessando a API de contexto que podem de fato usar a informação contextual para seus propósitos específicos. O que inclui a execução automática de serviços ativados por condições especiais de contexto, assim como a descoberta e alocação de recursos de um contexto corrente (MANTORO e JOHNSON, 2003).

Interface de comunicação: considerando que um sistema sensível ao contexto é distribuído por natureza, a comunicação precisa ser manipulada de forma apropriada. Isto afeta em particular os mecanismos de *reasoning*, que podem necessitar atualizar informações de contexto para diversos clientes ao mesmo tempo, enquanto que componentes de captura de contexto de outros clientes podem requisitar informações.

Atuadores: atuadores[3] são o complemento dos sensores, pois realizam comportamento inverso e podem ser utilizados para apresentar o resultado de uma atualização contextual ao ator ativo na execução do processo (BIEGEL e CAHILL, 2004).

[3] O termo atuador foi traduzido do inglês, actuator, e tem a sua origem na engenharia mecânica, porém é amplamente utilizado em sistemas autônomos. Atuadores modificam o ambiente através de ações.

Cada componente está representado em uma camada da arquitetura (Figura 3.1). O fluxo de informações de contexto na arquitetura genérica funciona da seguinte maneira: sensores capturam dados brutos e não processados do ambiente, os quais são refinados pela interface de captura de contexto em informação contextual. Os dados brutos são abstraídos em estruturas de dados, de forma que possam ser processados por software de alto nível. A informação de contexto é atualizada no repositório de contexto, que é responsável pelo armazenamento persistente de contexto usando um modelo de contexto apropriado. O repositório de contexto controla o acesso aos dados de contexto e, portanto, funciona como uma interface para o resto do sistema sensível ao contexto e quaisquer aplicações de usuários utilizando o contexto. A transformação de contexto é codificada em regras de inferência, as quais são validadas pelas máquinas de inferência. Tanto as regras, quanto a informação contextual é carregada pela máquina de inferência, que continuamente atualiza o repositório de contexto de acordo com as regras de inferência. Já que se trata de um sistema distribuído, atualizações de contexto podem ser propagadas para outros nós via interface de comunicação, a qual está atrelada ao hardware de comunicação.

A máquina de inferência pode também disparar qualquer atuador, que são acionados por atualização de contexto. A API de Contexto provém o contexto para aplicações de usuário e representa o corte arquitetural entre a gestão do contexto e a sua utilização. A máquina de inferência pode também notificar aplicações de usuários através da API de contexto, se a inferência de novo contexto requisitar.

Assim, é possível distinguir três tipos de fontes de atualização de contexto: ambiente - informação contextual que é capturada por sensores; máquina de inferência - novo contexto que é derivado independentemente por uma máquina de inferência

utilizando as suas próprias regras; usuário - o usuário atualiza dados de contexto através da API.

3.3 Modelagem de Contexto

Como visto na Figura 3.1, o ponto central de uma arquitetura sensível a contexto é o repositório de contexto. Além de ele armazenar o contexto atual, ele contém as estruturas de dados usadas para representar o modelo do contexto de forma que o contexto possa ser manipulado, processado e atualizado pelos outros componentes. Os modelos de contexto específicos podem tratar da modelagem do contexto em relação a uma aplicação ou uma família de aplicações, enquanto que os modelos de contexto mais genéricos podem ser usados em diferentes tipos de aplicação.

Alguns modelos consideram o estado corrente dos usuários, tal como "em uma reunião". Outros consideram o ambiente físico, ou seja, localidades. Outros usam os dois, além de considerar o tempo. Em (STRANG e LINNHOFF-POPIEN, 2004) é feita uma análise de diversas abordagens para modelos de contexto. Nesta subseção, citamos as abordagens mais relevantes e descrevemos em mais detalhes uma delas baseada em ontologias, que foi utilizada para a realização deste trabalho.

As abordagens de modelagem de contexto são classificadas pelos tipos de estruturas de dados que são usados enquanto a informação contextual é trocada. Então, por exemplo, o modelo chave-valor (*KeyValue*) é o de estrutura de dados mais simples disponível. Em particular, pares chave-valor são fáceis de gerenciar, porém perdem quanto à capacidade de representação de estruturas sofisticadas necessárias para que os algoritmos de busca de contexto sejam mais eficientes.

Na linha de abordagens com enfoque forte em modelagem gráfica, pode-se utilizar a UML (2009). Além de ser muito conhecida e utilizada, possui uma estrutura

genérica para modelar contexto. BAUER (2003), por exemplo, modelou aspectos de contexto relevantes existentes no gerenciamento de tráfego aéreo como extensões de UML.

Existem ainda os modelos orientados a objetos (BOUZY e CAZENAVE, 1997). Nesta abordagem, o objetivo é empregar os benefícios de uma abordagem orientada a objetos, tais como encapsulamento e reuso, para cobrir alguns problemas que surgem na dinâmica de contexto em ambientes ubíquos. Desta forma, os detalhes do processamento de contexto são encapsulados no nível do objeto e consequentemente escondidos dos outros componentes. O acesso à informação contextual é provido somente através de interfaces especificadas *a priori*. Logo, novos tipos de informação contextual (classes), assim como instâncias (objetos), podem ser manipulados no sistema de forma distribuída.

Por fim, antes de descrevermos em detalhes o modelo baseado em ontologias, temos os modelos baseados em lógica. Como exemplo, temos o trabalho de (GRAY e SAUBER, 2001). Neste tipo de abordagem, uma expressão lógica define as condições nas quais uma sentença ou fatos pode derivar (*reasoning*) a partir de um conjunto de outras sentenças ou fatos. Para descrever estas condições em um conjunto de regras, um sistema formal é usado. O contexto é consequentemente definido como fatos, sentenças e regras. Em geral, a informação contextual é adicionada ao modelo e atualizada em termos de fatos ou apagada se inferida a partir das regras.

3.3.1 Modelagem de Contexto baseado em Ontologia

O uso de ontologias permite um modo uniforme de representar conceitos e fatos, além de facilitar o compartilhamento e reuso do conhecimento contextual entre diversas aplicações. Outra característica interessante é a possibilidade de utilização de máquinas de inferência e raciocínio para validar ou avaliar o conhecimento contextual.

ÖTZTÜRK e AAMODT (1997) realizaram uma das primeiras abordagens de modelagem de contexto baseada em ontologias. A sua proposta foi analisar diversos estudos psicológicos sobre a diferença entre a habilidade de lembrar e o ato de reconhecer em relação a problemas de combinações de informações contextuais. Como resultado deste trabalho eles concluíram a necessidade de normalizar e combinar o conhecimento de diversos domínios.

WANG *et al*. (2004) criou uma ontologia de topo, isto é, uma ontologia que descreve conceitos bastante genéricos, independentes de domínio. Na ontologia criada, características básicas de entidades de contexto são capturadas. Uma coleção de ontologias específicas e suas características para cada subdomínio também foram criadas. As ontologias foram serializadas em OWL-DL com uma semântica equivalente à lógica de descrição, o que permite checagem de consistência e o *reasoning* de contexto usando mecanismos de inferência.

Padrões de metadados semânticos como *Resource Description Framework* (RDF) e *Web Ontology Language* (OWL) provêm um padrão para representação de dados de contexto. OWL adiciona semântica à representação em RDF para representar conceitos através de classes, conjuntos, restrições e outros. Máquinas de inferência podem atuar em documentos OWL, utilizando essa semântica, para derivar fatos ou responder perguntas sobre entidades semânticas.

3.4 Visão Geral de Frameworks e Arquiteturas para Sistemas Sensíveis ao Contexto

SINGH e CONWAY (2006) escreveram um *survey*, do ponto de vista arquitetural, sobre diversos frameworks sensíveis a contexto, incluindo: CONTEXT TOOLKIT (DEY *et al*., 2001), CoBrA (CHEN, 2004), CMF (KORPIPÄÄ *et al*., 2003) e

SOCAM (GU *et al*.,2005). No *survey* foram considerados: framework/software, representação de contexto e inteligência computacional.

Outro *survey* recente sobre sistemas sensíveis a contexto foi realizado por (BALDAUF *et al*., 2007). Neste trabalho, foram considerados princípios de projeto, incluindo arquitetura e tipos de contexto. Também foram analisados os trabalhos de GAIA (ROMAN *et al*, 2002), CONTEXT TOOLKIT, SOCAM e CoBrA.

Outro trabalho interessante do ponto de vista de arquitetura é o JCAF (*Java Context Awareness Framework*) que oferece suporte para infra-estrutura e um framework para desenvolvimento de aplicações sensíveis a contexto em JAVA (BARDRAM, 2005). Nas seções a seguir são descritas brevemente as características de cada um dos frameworks e/ou arquiteturas supracitadas, e é descrita em mais detalhes a arquitetura CoBrA, que foi a utilizada como base neste trabalho.

3.4.1 Arquiteturas

Context Toolkit

O Context Toolkit (DEY *et al*., 2001) foi proposto como uma ferramenta para apoiar os desenvolvedores de sistemas sensíveis a contexto. É baseado em dispositivos de contexto que podem operar de forma semelhante a uma interface gráfica, onde o objetivo é esconder a complexidade dos sensores físicos. Ele possui uma arquitetura em camadas que permite a separação da aquisição de contexto, representação e o processo de adaptação. Os dispositivos oferecem abstração de contexto e facilitam a reutilização do contexto capturado. Em termos gerais, oferece comunicação distribuída entre diversos dispositivos e possui mecanismo de descoberta centralizado. A arquitetura trabalha com eventos para notificar as atualizações de contexto através do uso de uma *thread* para cada evento, o que pode sobrecarregar o sistema e comprometer a sua escalabilidade. Não existe nessa arquitetura um módulo para *reasoning*, já que o modelo

utilizado para representação de contexto é chave/valor (*key/value*), o que dificulta o *reasoning*. Como restrição de segurança, o acesso às informações de contexto só é possível ao proprietário das informações.

Context Management Framework

KORPIPÄÄ *et al*. (2003) propôs o Framework de Gerenciamento de Contexto cuja arquitetura é composta por quatro entidades funcionais principais: o gerente do contexto, os servidores de recursos, os serviços de reconhecimento de contexto e a aplicação. Enquanto os servidores de recurso e os serviços de reconhecimento de contexto são componentes distribuídos, o gerente de contexto funciona como um servidor centralizado gerenciando um *blackboard*. Ele armazena os dados de contexto e provê esta informação para aplicações clientes. A estrutura da ontologia e o vocabulário aplicado na ferramenta de Gerenciamento de Contexto são descritos em RDF.

GAIA

O projeto GAIA é um *middleware* baseado em CORBA que apóia aplicações ubíquas ativas (ROMAN *et al*., 2002). O *middleware* GAIA provê um serviço de contexto para ajudar aplicações a buscarem e publicarem informações contextuais. Seu contexto é representado de forma diferente, com predicados de 4-aridades da seguinte forma Context (<ContextType>, <Subject>, <Relater>, <Object>), e escritos em DAML+OIL. O <Context Type> está relacionado ao tipo de contexto do predicado sendo descrito. O <Subject> pode ser uma pessoa, um lugar ou algo no qual o contexto esteja relacionado. O <Object> é o valor associado com o <Subject>. O <Relater> relaciona o <Subject> com o <Object> tal como um operador de comparação (=, >, ou <), um verbo, ou preposição. Por exemplo, considere a seguinte instância de contexto Context (temperature, room 3231, is, 98 F). A sintaxe é usada tanto para representar o contexto como para a formação de regras de inferência. Considerando a questão

segurança, GAIA utiliza diversos mecanismos para definir restrições de privacidade e comunicação segura na busca de localidade de pessoas.

JCAF

No JCAF, a informação contextual é tratada por serviços separadamente. Existe um serviço que permite publicar e outro recuperar informações de contexto (Bardram, 2005). JCAF é baseado em um modelo *peer-to-peer*. No entanto, não existe suporte para descoberta automática de *peers* ou um *super-peer* em seu framework. A comunicação é baseada em Java RMI (*Remote Method Invocation*), há suporte para diversos tipos de sensores de monitoramento de localização e classes básicas para descrever entidades relevantes que são utilizadas em aplicações sensíveis a contexto.

SOCAM

SOCAM é um *middleware* para construção de serviços sensíveis a contexto (GU *et al*., 2008). Ele oferece suporte para modelagem de contexto e *reasoning* baseado em OWL. A sua implementação é fortemente baseada em RMI. A representação é baseada em ontologias generalizadas e de domínio, que são divididas em subdomínios com o objetivo de reduzir a complexidade do processamento das informações contextuais. A arquitetura usa o modelo cliente/servidor e tem como componente central o Interpretador de Contexto. Este interpretador é responsável por interagir com as fontes de contexto e a base de dados contextual para fornecer informações para os serviços móveis e para o seu serviço de localização de serviços. Outro componente importante é o *Reasoner* de Contexto que é responsável por manipular a ontologia de contexto para realizar o *reasoning*. Neste componente, também existe a possibilidade de adição de outros mecanismos de *reasoning* de modo a diversificar a capacidade para algum caso específico.

3.4.2 A Arquitetura CoBrA

Context Broker Architecture (CoBrA) (CHEN, 2004) é uma arquitetura baseada em agentes para o suporte a sistemas sensíveis a contexto em ambientes inteligentes. O componente central dessa arquitetura é chamado de *Context Broker*. Este componente é responsável por manter um modelo compartilhado de contexto para agentes, serviços e dispositivos em um determinado lugar. Uma de suas características é prover proteções para manter a privacidade dos usuários, de acordo com as políticas de segurança definidas. A inteligência computacional usa inferência baseada em regras para o *reasoning* de contexto e manutenção do conhecimento.

Na figura 3.2, é possível ver que, na arquitetura CoBrA, o *Context Broker* é composto por quatro componentes principais: Base de Conhecimento de Contexto, Mecanismo de *Reasoning* de Contexto, Módulo de Aquisição de Contexto e Módulo de Gestão de Privacidade.

Como CoBrA utiliza um componente central, existe a possibilidade de criação de federações de *brokers* para evitar que uma falha no componente central derrube todo o sistema. Desta forma, a existência de diferentes *Context Brokers* pode significar conhecimento contextual distinto entre os *Brokers*. No entanto, também existe a possibilidade de um *broker* se registrar em outros para que seja possível fundir e compartilhar o conhecimento contextual de forma a mante-lo consistente com o contexto

global.

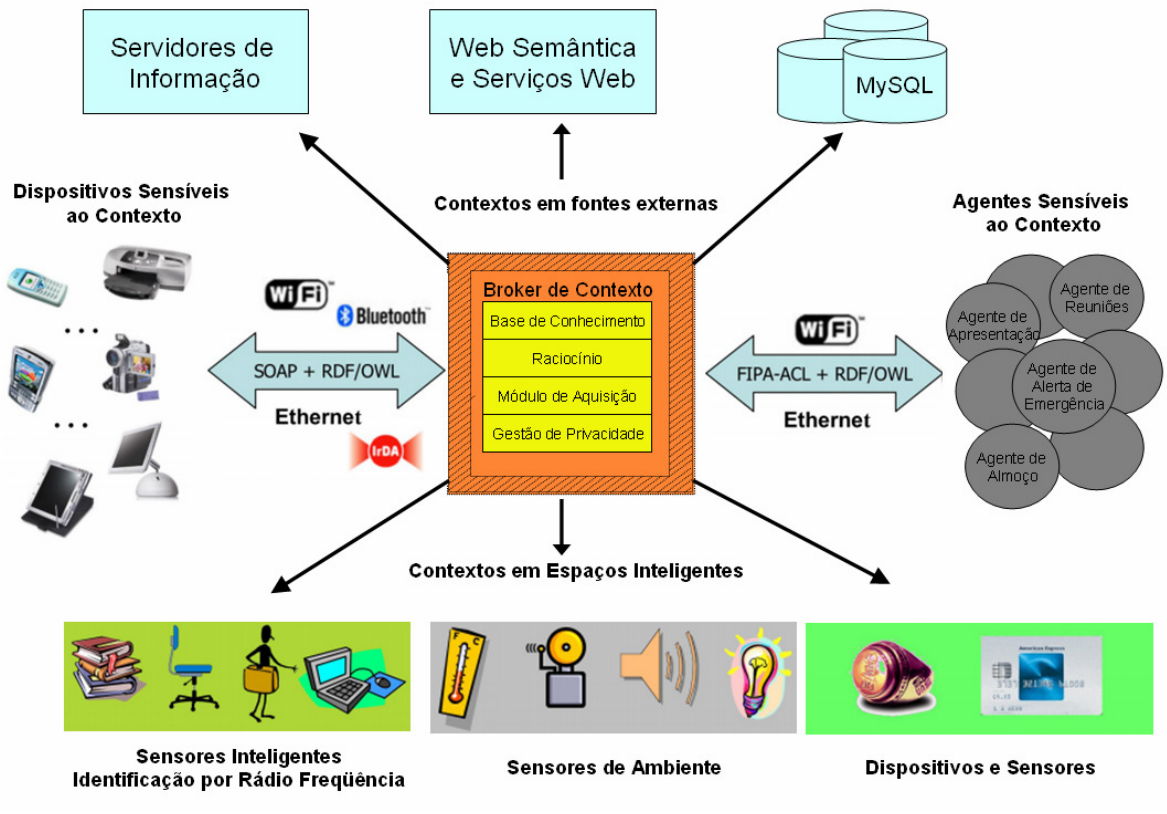

Figura 3.2 – Arquitetura CoBrA (CHEN, 2004).

CoBrA usa uma abordagem de ontologias baseada em OWL, conhecida como CoBrA-Ont (CHEN *et al.*, 2004), que é um conjunto de ontologias para agentes que descrevem informação contextual e compartilham conhecimento contextual.

CoBrA inclui uma linguagem flexível para controlar o acesso ao contexto chamada Rei (KAGAL *et al.*, 2003). Esta linguagem consiste em uma ontologia que pode modelar conceitos de direitos, proibições, obrigações e acesso através de regras que podem ser dependentes de domínio. Ontologias em Rei são expressas em linguagem RDF.

O framework de *reasoning* de contexto do CoBrA é composto de um conjunto de estados. Quando um *Context Broker* captura informação de contexto de fontes externas, essa informação é armazenada em uma base persistente de conhecimento.

A inclusão de novo conhecimento dispara o comportamento de *reasoning* do *Context Broker*. Dependendo do tipo de conhecimento declarado, inferências lógicas baseadas em regras diferentes são executadas para interpretar o contexto. Antes que qualquer conhecimento possa ser atualizado na base, validações sobre inconsistências precisam ser realizadas. Se algum conflito é identificado, os comportamentos de *reasoning* apropriados são disparados para tentar resolver a inconsistência.

Com base nesse framework de *reasoning*, CHEN (2004) desenvolveu um protótipo para demonstrar a aplicabilidade do *reasoning* de contexto. Neste protótipo, o *Context Broker* é implementado como um agente em acordo com o padrão FIPA (2005) usando o *middleware* JADE (2009). JADE é um *middleware* para o desenvolvimento de sistemas multi-agentes. Ele provê funcionalidades tais como distribuição, mobilidade, entre outros.

O comportamento funcional de um *Context Broker* é implementado como um conjunto de classes de comportamentos de agentes do JADE. Estas classes de comportamentos compartilham um conjunto comum de recursos de agentes. Isto inclui uma base de conhecimento, uma coleção de *reasoners* de contexto e os objetos essenciais para as comunicações entre os agentes e para a descoberta de serviços na plataforma FIPA. Em tempo de execução, esses comportamentos são gerenciados por um escalonador interno de uma super classe do JADE, ou seja, a classe do *Context Broker* é uma subclasse da classe agente do JADE.

A comunicação entre o *Context Broker* e outros agentes segue o padrão FIPA de comunicação entre agentes. A implementação define diversos protocolos de interações para os agentes se comunicarem com o *Context Broker*. Estes protocolos são protocolos de comunicação para informar a um *Context Broker* sobre certas condições, consultar o conhecimento contextual e convencer o *Context Broker* a acreditar que certo

conhecimento é verdadeiro através de argumentação (MCBURNEY e PARSONS, 2004).

A implementação do *Context Broker* suporta dois tipos diferentes de representações de conteúdo de mensagens. Dependendo do propósito da comunicação, mensagens de conteúdo são representadas usando diferentes linguagens. Para compartilhar informação com outros agentes, uma representação XML/RDF da linguagem OWL é utilizada. Para responder consultas sobre certo conhecimento contextual, a linguagem RDQL (SEABORNE, 2004) é utilizada. Os vocabulários usados nesta representação são importados das ontologias SOUPA e CoBrA-Ont. Estas ontologias também são utilizadas para expressar conhecimento interno do *Context Broker*. O conhecimento do *Context Broker* é representado como comandos RDF e é armazenado em uma base persistente de conhecimento. Esta base é mantida por um banco de dados MySQL que é persistido pelo framework para web semântica JENA 2 (2009).

O *reasoning* de contexto do CoBrA usa um número diferente de sistemas baseados em regras, o que inclui os *reasoners* baseados em regras (REYNOLDS, 2009) do JENA 2, JESS (FRIEDMAN-HILL, 1997) e THEORIST (um reasoner baseado em Prolog) (POOLE, 91). Diferentes sistemas baseados em regras provêm diferentes apoios para inferência lógica de *reasoning* de contexto. Os *reasoners* baseados em regras são usados para inferências de ontologias OWL; o mecanismo de regras do JESS é usado para interpretar o contexto com regras específicas de domínio. Já o sistema Theorist, que é um programa Prolog executado como um processo separado usando uma interface nativa de Java com Prolog, é usado para suportar as inferências lógicas que resolvem inconsistências no conhecimento. Nem todos os *reasoners* são implementados em Java. Portanto, o *Context Broker* usa interfaces de programação diferentes para interagir com

sistemas baseados em regras. Ele diretamente interage com os mecanismos do JESS e do JENA usando a sua API Java existente.

3.4.3 Análise dos Requisitos para uma Arquitetura Base de Infra-Estrutura

Neste trabalho procuramos comparar as arquiteturas de acordo com alguns requisitos e características que consideramos necessários para o problema:

- **Suporte a representação de contexto baseado em ontologia, em particular OWL ou RDF** - como RDF/XML é uma das sintaxes normativas de OWL e existem muitas ferramentas de apoio para editar, gerenciar e armazenar dados OWL em RDF/XML, o esforço para manutenção e desenvolvimento é facilitado. Além disso, o uso de linguagens como OWL e RDF adicionam expressividade na representação e na modelagem do conhecimento contextual, em comparação com outras representações.
- **Suporte a mecanismos de *reasoning*** - a representação explícita das regras de *reasoning* de contexto ajuda a separar a lógica de *reasoning* de alto nível, da lógica de implementação funcional de baixo nível. Com isto, desenvolvedores podem efetuar manutenção nos componentes de *reasoning* de contexto sem muito esforço de reimplementação. Outra vantagem desta abordagem é que quando regras de interpretação de contexto são representadas de forma explícita, técnicas de meta-*reasoning* podem ser criadas para detectar e resolver interpretações de contexto inconsistentes. Ou seja, podem ser criadas técnicas de raciocínio sobre o próprio raciocínio, utilizando a combinação com técnicas de planejamento ou aprendizado;
- **Robustez** – uma arquitetura com suporte à distribuição para o caso de falhas e escalabilidade para o problema de desempenho é mais robusta em relação às que têm pouco ou nenhum suporte;
- **Uso de padronização** – integração e interoperabilidade são requisitos essenciais no sentido que permitem que componentes desenvolvidos de forma independente sejam

reutilizados, sendo assim, uma arquitetura que permite padronização de comunicação entre os componentes é mais vantajosa que uma que limita o desenvolvimento de futuros componentes, assim como a integração com outros já existentes;

- **Arquitetura orientada a agentes** – uma arquitetura orientada a agentes fornece abstrações necessárias para o desenvolvimento de inteligência distribuída, permite interatividade entre sensores e atuadores de contexto, alem de permitir decomposição de comportamento em um sistema complexo. Uma arquitetura orientada a agentes é mais vantajosa que uma orientada a serviços, pois um agente além de encapsular o comportamento de um serviço, possui comportamento autônomo e pró-ativo;

Tabela 3.1 – Critérios de escolha de arquitetura base.

	Representação de Contexto em OWL/RDF	Suporte a Mecanismos de *Reasoning*	Robustez	Uso de padronização	Orientada a agentes
Context Toolkit	×	×	×	×	×
FGC	×	✓	✓	×	×
Gaia	×	✓	✓	✓	×
JCAF	×	×	✓	✓	×
SOCAM	✓	✓	✓	×	×
CoBrA	✓	✓	✓	✓	✓

FGC: Framework de Gerenciamento de Contexto

A Tabela 3.1 apresenta as arquiteturas descritas na seção anterior e quais critérios são satisfeitos por cada uma delas. Considerando a representação de contexto, somente as arquiteturas SOCAM e CoBrA suportam OWL ou RDF. Como já visto, o Context Toolkit permite representação de ontologia através de atributos chave-valor, Gaia permite representação através de DAML+OIL, e o JCAF não permite nenhuma representação de contexto. Considerando o critério de suporte a mecanismos de *reasoning*, como já apresentado anteriormente, as arquiteturas FGC, Gaia, SOCAM e CoBrA provêm componentes para tal fim. Quanto ao critério robustez, somente o Context Toolkit não apresenta evidências na literatura de acordo com os atributos

distribuição ou escalabilidade relacionados à robustez. Todos os outros, seja através da distribuição, da orientação a serviços ou orientação a agentes, provêm maior robustez. Considerando o uso de padronização, a arquitetura Gaia é a que segue a especificação CoBrA, o JCAF usa RMI e CoBrA é FIPA-*compliant*. E, por fim, quanto ao critério arquitetura orientada a agentes, somente a arquitetura CoBrA satisfaz o critério.

Desta forma, CoBrA foi selecionada a arquitetura de infra-estrutura base. Porém, é importante ressaltar quais são seus pontos fracos e que deverão ser minimizados pela arquitetura proposta neste trabalho. Um dos pontos fracos do CoBrA, é que a implementação de seu *Context Broker* foi construída fortemente acoplada aos comportamentos específicos de domínio. No caso do seu protótipo, o domínio de reuniões (o chamado *EasyMeeting*). *EasyMeeting* é um sistema de ambiente de reunião inteligente cujo objetivo é prover serviços e informações relevantes para os participantes de uma reunião baseado no contexto de suas necessidades locais. O forte acoplamento dos comportamentos com o *Context Broker* limita a sua extensão para soluções em domínios diferentes, além de dificultar a implementação de soluções mais generalizadas. Outra característica importante que poderia ter sido considerada no projeto da arquitetura é a capacidade de reuso do *reasoning* de contexto. Ele foi implementado de forma totalmente acoplada ao comportamento (*behaviors* do JADE) de um agente, o que limita a sua reutilização para outras soluções que não sejam baseadas em agentes, ou que sejam, mas não utilizem o JADE como API de desenvolvimento e comunicação de agentes.

3.5 Considerações Finais do Capítulo

Neste capítulo, foram discutidas as arquiteturas para sistemas sensíveis a contexto já propostas na literatura, as suas diferentes abordagens e características,

vantagens e desvantagens, assim como a arquitetura referência de SCHMOHL e BAUMGARTEN (2008). A partir de critérios estabelecidos, a arquitetura CoBrA foi selecionada como base para a arquitetura proposta nesta dissertação.

Para que CoBrA possa ser utilizada na gestão de contexto de atividades de processo, foi identificada a necessidade de refatoração, desacoplamento e extensão da sua arquitetura, como será visto na arquitetura proposta no próximo capítulo.

4 A-CoBrA: Uma Arquitetura Baseada em Contexto de Atividades de Processos de Trabalho

Neste capítulo é apresentada A-CoBrA, uma arquitetura baseada em contexto de atividades de processos de trabalho, contribuição principal deste trabalho.

4.1 Descrição de um Cenário

Nesta seção, a descrição do cenário utiliza um exemplo de um processo fictício, de fácil entendimento, do domínio bancário, a fim de apoiar a demonstração dos requisitos e componentes da arquitetura proposta.

A Figura 4.1 mostra um processo de aprovação de hipoteca simplificado. Quando um cliente solicita uma hipoteca ao banco, uma atividade de análise de risco (Estimar Risco) é realizada. Em seguida, as parcelas do empréstimo são criadas e limitadas tendo como base o risco (atividade Criar Empréstimo), ou o empréstimo é rejeitado (atividade Rejeitar).

Como já visto, de forma geral, um sistema sensível ao contexto é capaz de perceber e entender a resposta para as seguintes questões: quem é você, e com quem você está? O que você está fazendo, por que está fazendo e para quê? Se não está fazendo agora, quando estará? Quais são suas motivações e crenças para fazer isto? No entanto, neste trabalho, também estamos preocupados em saber: alguém já fez algo parecido? Se fez, como fez? Quais informações estão disponíveis sobre o que foi feito?

Na execução da atividade Estimar Risco, um hipotético Analista de Risco irá utilizar o seu conhecimento tácito e todas as informações disponíveis (conhecimento explícito) para realizar a sua análise. Neste cenário, a gestão de conhecimento contextual da atividade de Análise de Risco pode ser relevante para apoiar a atividade e mitigar os riscos associados à concessão de empréstimos e hipotecas.

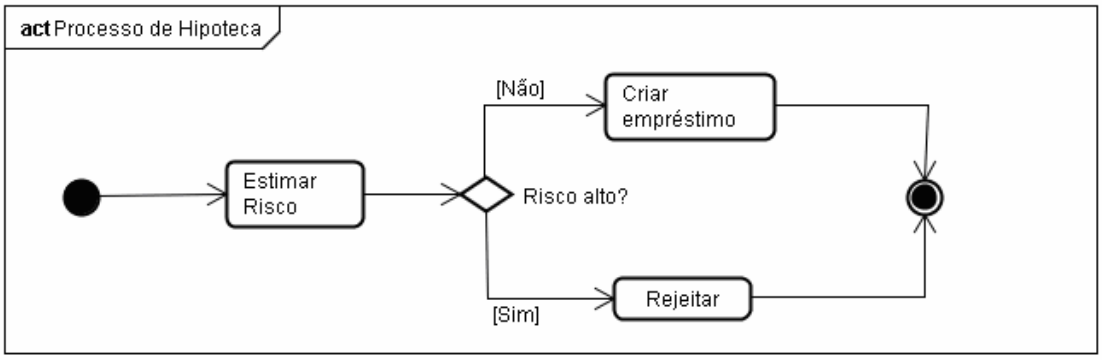

Figura 4.1 – Um processo de aprovação de hipoteca simplificado.

Voltando às perguntas que um sistema sensível ao contexto de atividades é capaz de perceber, teríamos para a atividade Estimar Risco:

a) Quem é você: Analista de Risco;

b) O que você está fazendo: estimando o risco para aprovação de uma hipoteca;

c) Porque está fazendo: porque a atividade do processo foi iniciada;

d) Para quê está fazendo: para minimizar o risco de aprovar uma hipoteca que o solicitante não poderá pagar;

e) Quais as motivações: garantir que o banco aprove empréstimos seguros;

f) Com o que/quem você interage: com o solicitante da hipoteca;

g) Alguém já realizou a mesma atividade: outros Analistas de Risco podem ter feito análises de riscos de pessoas com o mesmo perfil do solicitante;

h) Se a atividade já foi realizada previamente, como fez o executor que a realizou: perguntas sobre a vida pessoal foram realizadas, conversou com as referências indicadas pela pessoa, etc.;

i) Quais informações estão disponíveis sobre as tarefas realizadas durante a atividade: está disponível o áudio das iterações telefônicas do Analista de Risco com as referências do solicitante, de modo que se possa perceber como deve ser feita a abordagem com as referências de um solicitante.

Após a associação entre a atividade e o contexto capturado, o sistema sensível ao contexto de atividades irá raciocinar para consultar a memória organizacional e tentar descobrir informações de contexto relevantes para o executor da atividade atual. Como já visto no modelo de GC com suporte à execução de um processo de trabalho de NUNES (2007), esta seria a fase de recuperação de contexto, para posterior apresentação, que são problemas em aberto e que não foram explorados por NUNES (2007).

Nesta fase de recuperação, um sistema sensível ao contexto de atividades poderia entender e responder às seguintes perguntas:

f.1) Quem é o solicitante da hipoteca:

f.1.1) O solicitante é conhecido, temos um histórico do seu perfil? Caso afirmativo, seu perfil é positivo? Tem histórico de dívidas? Se casado, seu (sua) companheiro (a) tem perfil positivo?

f.1.2) Caso o solicitante não seja conhecido, qual é o perfil em média de pessoas com a mesma faixa etária, e/ ou profissão, e/ ou estado civil? Qual é o histórico de operações bancárias destas pessoas, caso exista?

g.1) Alguém já realizou a mesma atividade:

g.1.1) Outros Analistas de Risco realizaram análises de riscos de pessoas com o mesmo perfil do solicitante? Que recursos e artefatos foram necessários? Algum sistema computacional foi utilizado? Que informações foram importantes para determinar o risco?

É necessária, então, uma arquitetura que dê suporte à reutilização do conhecimento contextual das atividades de um processo, tal como o ilustrado. Tal arquitetura tem como requisitos a disponibilização dos componentes previstos na arquitetura conceitual genérica proposta por SCHMOHL e BAUMGARTEN (2008), que foi detalhada na seção 3.1. A seguir, os requisitos são relacionados com o cenário descrito anteriormente:

I) Disponibilizar sensores de contexto de atividades de processo

Por exemplo, quando o Analista de Risco inicia a atividade, o sensor de contexto desta atividade deve estar ativado. Agentes inteligentes devem ser configurados para monitorar o contexto de atividades de um processo e, uma vez que a atividade tenha sido iniciada, eles são capazes de perceber o contexto e capturar informações.

II) Disponibilizar atuadores de contexto de atividades de processo

Como já visto no capítulo anterior atuadores são o complemento dos sensores e podem ser utilizados para apresentar o resultado de uma atualização contextual. Deverão existir agentes que atuarão na aplicação de contexto, apresentando os resultados para os executores da atividade corrente, se encontrados.

III) Disponibilizar uma interface de captura de contexto de atividades de processo

Todas as informações relacionadas com a atividade de Estimar Risco devem ser coletadas e transformadas em uma representação que permita processamento, transferência ou armazenamento. No caso dessa arquitetura, as informações de contexto são representadas utilizando ontologias.

IV) Disponibilizar repositório de contexto de atividades de processo

O contexto capturado na atividade Estimar Risco deve ser armazenado em um repositório de contexto com suporte para a representação ontológica do modelo de contexto.

V) Disponibilizar *reasoning* sobre contexto de atividades de processo

No momento em que o contexto da atividade Estimar Risco é capturado, ao tentar inferir conhecimento sobre o contexto da atividade executada pelo Analista de Riscos, é necessário utilizar máquinas de inferência que vão, por exemplo, inferir sobre o contexto atual da atividade de Estimar Risco e outros contextos armazenados sobre a mesma atividade. Também deve ser possível realizar o *reasoning* sobre as informações de contexto com as informações do solicitante da hipoteca, como por exemplo, verificando se o mesmo possui um perfil positivo ou não a partir do contexto ao qual está inserido (tipo de hipoteca, valor, e seu histórico ou comparativo de histórico com perfis semelhantes).

VI) Disponibilizar uma API de contexto

Tanto na percepção do conhecimento tácito no contexto de uma atividade de processo (por exemplo, Estimar Risco), como na apresentação do conhecimento explícito recuperado da memória organizacional, (por exemplo, conjunto de mensagens trocadas no contexto da atividade Estimar Risco) é preciso uma API de contexto que irá prover uma interface para aplicações sensíveis ao contexto efetivamente utilizarem a informação contextual. Esta API deve prover a separação da lógica de gestão de conhecimento contextual, da lógica da aplicação sensível ao contexto. A API de contexto é representada pelos comportamentos oferecidos pelo agente *Broker* de contexto de atividades. O *Broker* de Contexto de Atividades deve ser responsável por isolar o tratamento do conhecimento contextual das aplicações sensíveis a contexto de atividade, que por sua vez devem ser construídas com base nos agentes sensores de contexto (como visto no item I) e agentes atuadores de contexto (como visto no item II).

VII) Prover uma interface de comunicação adequada

Considerando que um sistema sensível ao contexto tende a ser distribuído por natureza, o tratamento da comunicação é essencial para o seu bom funcionamento. A comunicação entre o *Broker* e outros agentes (sensores e atuadores) deve seguir o padrão FIPA de comunicação. Diversos protocolos de interações para os agentes se comunicarem com o *Broker* devem existir. Isto permite que os componentes da arquitetura trabalhem de forma assíncrona e paralela. Então, por exemplo, os mecanismos de *reasoning* podem necessitar atualizar informações de contexto para diversos clientes ao mesmo tempo, enquanto que componentes de captura de contexto de outros clientes podem requisitar informações.

VIII) Prover aplicações sensíveis ao contexto de atividades de processo

Seguindo a arquitetura genérica de SCHMOHL e BAUMGARTEN (2008), uma aplicação sensível ao contexto deve prover aplicações de contexto, que na arquitetura proposta são aplicações baseadas em agentes sensores e atuadores acessando a API de contexto, como visto no item VI, o *Broker* sensível ao Contexto de Atividades. Como visto nos itens VI e VII, o *Broker* sensível ao Contexto de Atividades encapsula a API de contexto e tem como interface de comunicação a troca de mensagens FIPA-*complient*. A aplicação sensível ao contexto de atividades construída com base nos agentes sensores e atuadores será sempre específica do domínio de negócio, e deve ser facilmente integrada com a API fornecida pelo *Broker*, o que se dá através da alta interatividade que os agentes são capazes de atingir através da troca de mensagens e do protocolo de compartilhamento de contexto.

4.2 Descrição da Arquitetura

Nesta seção, é descrita em detalhes a arquitetura A-CoBrA proposta. A-CoBrA (*Activity Context Broker Architecture*) trata-se de uma arquitetura projetada para a

construção de sistemas sensíveis ao contexto de atividades que tornem possível a gestão de contexto e do conhecimento em um ambiente organizacional. Ela instancia a arquitetura de referência proposta por SCHMOHL e BAUMGARTEN (2008) e tem como infra-estrutura base a arquitetura CoBrA.

A Figura 4.2 ilustra como a arquitetura de referência de SCHMOHL e BAUMGARTEN (2009) foi instanciada. Nesta instanciação, é possível visualizar como cada componente de A-CoBrA é posicionado em cada um dos níveis da arquitetura. No nível mais básico, o léxico, os Agentes Sensores são responsáveis por capturar informações de contexto de baixo nível e transformá-las em uma representação ontológica de contexto através de uma interface de captura de contexto. Já os Agentes Atuadores são responsáveis por atualizar o ambiente ou os possíveis interessados em um caminho contrário ao realizado pelos Agentes Sensores, ou seja, na transformação do contexto de alto nível em contexto que possa ser inteligível e/ou visível aos interessados. Portanto, na arquitetura proposta por esse trabalho, os Agentes Sensores e Atuadores atuam nos níveis Léxico e Sintático, ultrapassando os limites dos Sensores e Atuadores propostos por SCHMOHL e BAUMGARTEN (2008).

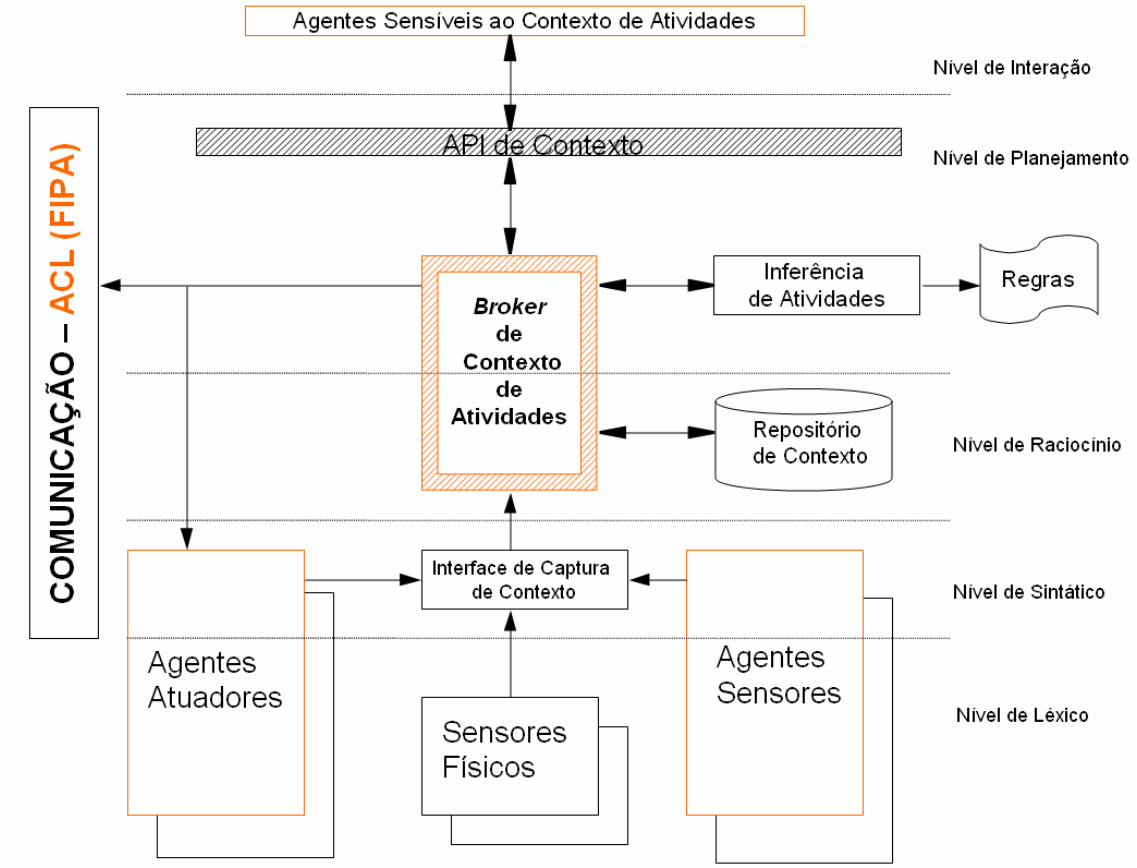

Figura 4.2 – Arquitetura A-CoBrA.

No nível de Raciocínio encontra-se o componente central de A-CoBrA, o *Broker* de Contexto de Atividades, responsável por receber os contextos de atividades dos Agentes Sensores e utilizar os seus Mecanismos de Inferência de Atividades para raciocinar em cima do conhecimento contextual existente no Repositório de Contexto. O Mecanismo de Inferência de Atividades pode se utilizar de regras pré-definidas específicas ao domínio e que podem ser aplicadas a uma máquina de inferência. Segundo a arquitetura de referência de SCHMOHL e BAUMGARTEN (2008), os mecanismos de inferência e regras encontram-se no nível de Planejamento.

No nível de interação, temos os Agentes Sensíveis ao Contexto de Atividades que são responsáveis pela coordenação dos Agentes Sensores e Atuadores. Seu objetivo é facilitar a interação com o usuário ou seu ambiente durante o desenvolvimento de suas

atividades. Todos os Agentes interagem com o Agente *Broker* de Contexto de Atividades por meio de trocas de mensagens padronizadas por um protocolo de compartilhamento de contexto.

4.2.1 Fluxo de Interação

Na Figura 4.3, é possível visualizar A-CoBrA sob o ângulo do trabalho proposto por NUNES (2007), onde foi proposto um modelo para Gestão de Conhecimento baseado fortemente na Gestão de Conhecimento Contextual de Atividades. Neste trabalho, o modelo foi evoluído para uma arquitetura baseada em contexto de atividades de processos.

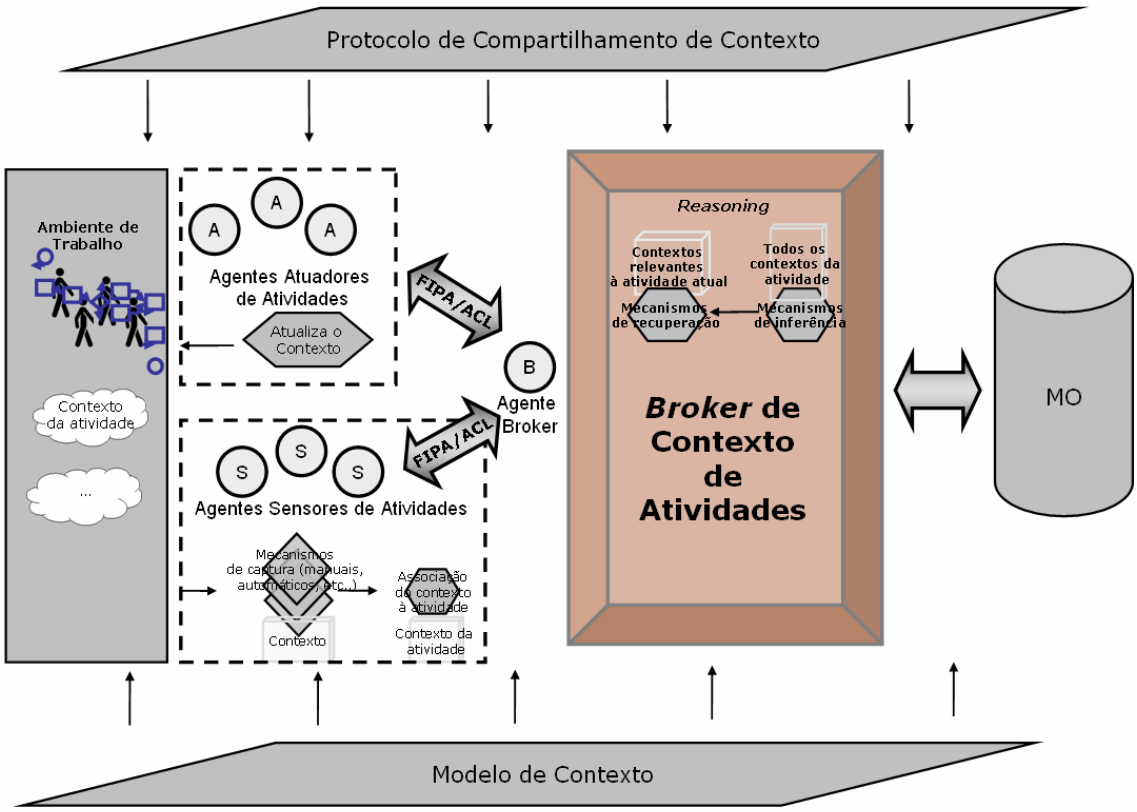

Figura 4.3 – A-CoBrA e Atividades de Processos.

Nesta visão da arquitetura, é possível observar como os componentes interagem no modelo proposto por NUNES (2007). O ambiente de trabalho organizacional é monitorado por Agentes Sensores de Atividades que devem ser construídos com base

em mecanismos de captura manuais ou automáticos, definidos a partir do prévio conhecimento do processo existente na organização e da escolha do padrão de captura apropriado para cada atividade (NUNES, 2007).

Os Agentes Sensores de Atividades também são responsáveis por transformar o contexto de baixo nível em contexto de alto nível utilizando o modelo de contexto conhecido por todos os componentes da arquitetura. Neste caso, é utilizada uma representação ontológica de contexto, proposta no trabalho de NUNES (2007), em OWL.

A comunicação entre os Agentes Sensores e o *Broker* de Contexto de Atividades é feita por um Protocolo de Compartilhamento de Conhecimento Contextual que define os padrões para recebimento e envio de mensagens entre os agentes da arquitetura. Este protocolo é FIPA-*complient* (FIPA, 2005) e segue o padrão internacional de interação entre Agentes.

O *Broker* de Contexto de Atividades possui mecanismos de inferência e recuperação que podem ser desacoplados da arquitetura e são implementados com a utilização do framework JENA (2009) que suporta o desenvolvimento de aplicações, as quais utilizam ontologias e máquinas de inferência. O JENA é responsável por toda a abstração da camada de persistência no repositório de contexto, que é responsável por compor a memória organizacional de conhecimento contextual sobre atividades.

Utilizando-se dos mecanismos de inferência e recuperação, o *Broker* de Contexto de Atividades pode descobrir conhecimento contextual de atividades relevante para uma atividade em execução em um cenário organizacional e apresentá-lo ao possível interessado. Para tanto, ele deve utilizar-se dos Agentes Atuadores, responsáveis por apresentar e atualizar o contexto no ambiente do usuário.

4.2.2 Modelo Conceitual

Um modelo conceitual representa "conceitos" (entidades) e relacionamentos entre eles. O modelo conceitual é explicitamente definido para ser independente de detalhes de implementação, tais como o sistema de gerência de banco de dados a ser utilizado. Nele são apresentados relacionamentos entre os componentes da arquitetura através de uma abstração de classes, dependências e relacionamentos, o que auxilia a desfazer mal entendidos sobre os termos e garantir que não ocorram diferentes interpretações, o que poderia ter como conseqüência um projeto de software mal sucedido. Uma vez que a arquitetura tenha um modelo conceitual bem definido, o modelo se torna uma base estável (FOWLER, 1997), para os subseqüentes desenvolvimentos de aplicações em ambientes de GC, que reutilizem a arquitetura. Com este objetivo, é apresentado o modelo conceitual de A-CoBrA na Figura 4.4.

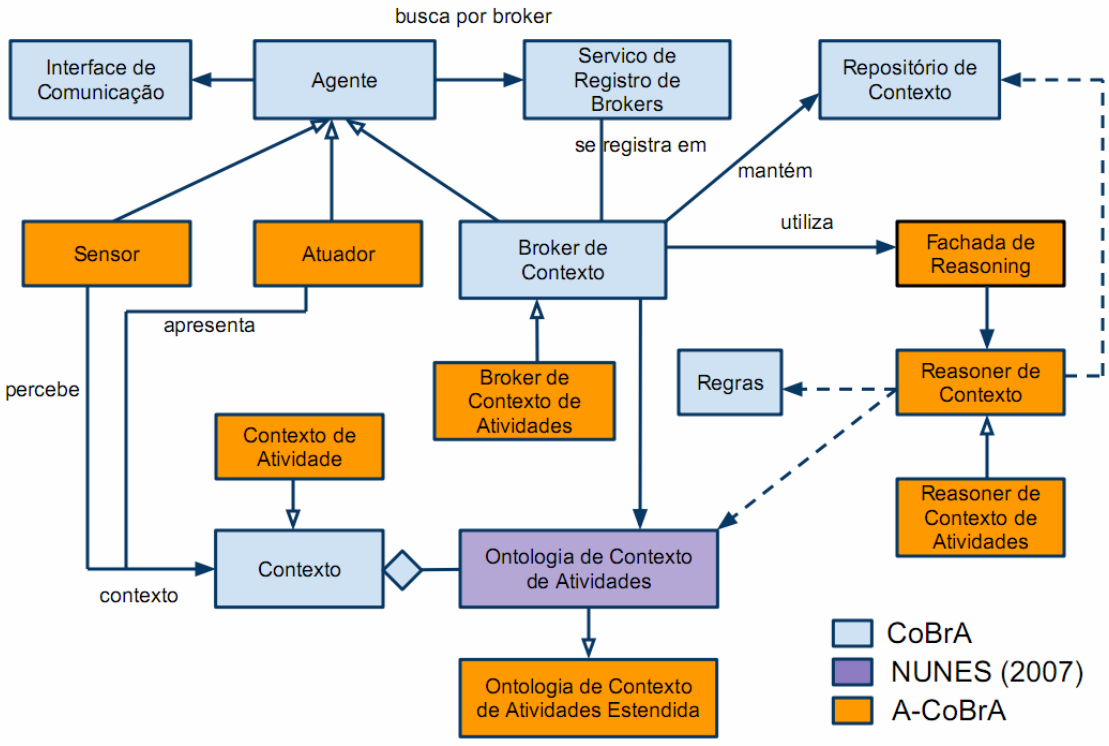

Figura 4.4 – A-CoBrA: Modelo Conceitual.

Começando pelo componente central, o *Broker* de Contexto de Atividades é uma refatoração e posterior especialização do *Broker* de Contexto da arquitetura CoBrA (CHEN, 2004). O principal motivo para isso é herdar as suas funcionalidades de registro no Serviço de Registro de *Brokers* e manutenção do Repositório de Contexto. Outro relacionamento herdado é o de especialização de um Agente que é a sua característica mais básica.

Na arquitetura, um agente pode ser um Sensor, Atuador ou um *Broker* de Contexto. Agentes Sensores e Atuadores têm as capacidades de perceber e apresentar Contexto, respectivamente. Tal Contexto pode ser uma especialização para Contexto de Atividade e é composto por uma ou mais instâncias de Ontologias de Contexto.

Como contribuição principal desta arquitetura, o *Broker* de Contexto de Atividades utiliza uma Fachada de *Reasoning* responsável por isolar e diminuir o acoplamento do Reasoner de Contexto, ou de sua especialização, o Reasoner de Contexto de Atividade. Este Reasoner é responsável por realizar as inferências sobre contextos de atividades e possui dependências com as Regras e com o Repositório de Contexto.

Não faz parte do escopo deste trabalho oferecer soluções customizadas para os agentes sensores e atuadores, pois são específicos e dependentes do ambiente de execução de atividades de um determinado processo. Desta forma, as próximas subseções descrevem somente os componentes centrais da arquitetura, ou seja, o *Reasoner* e *Broker* de Contexto de Atividades e suas dinâmicas de execução que são aplicáveis a qualquer atividade ou processo.

4.2.3 *Broker* de Contexto de Atividades de Processo

O componente central da arquitetura A-CoBrA é chamado de *Broker* de Contexto de Atividades e a sua principal função é permitir o compartilhamento de

conhecimento contextual de atividades. Em um ambiente de gestão de conhecimento, o *Broker* de Contexto de Atividades tem as seguintes responsabilidades:

(i) prover um modelo centralizado de contexto de atividades que pode ser compartilhado por todos os dispositivos, serviços e agentes;

(ii) raciocinar sobre o conhecimento contextual de atividades;

(iii) perceber e resolver inconsistências no conhecimento contextual que é armazenado no modelo compartilhado de contexto;

Na Figura 4.5 é possível visualizar o diagrama de pacotes que ilustra a arquitetura A-CoBrA, onde foram selecionadas as classes mais significativas e seus inter-relacionamentos.

O *ContextActivityBroker* é uma refatoração e especialização do *ContexBroker* da arquitetura CoBrA. Isto permite reutilizar as propriedades e comportamentos necessários para o funcionamento do *Broker* em um ambiente inteligente como o projetado por CHEN (2004). Como pode ser visto na Figura 4.5, o *ContextActivityBroker* pode reutilizar a classe *KBManager*, responsável pela gestão da base de conhecimento, que na arquitetura A-CoBrA é o repositório de conhecimento contextual sobre as atividades. A classe *KBManager* é responsável por manter a persistência e consistência do modelo de contexto através do framework JENA. No diagrama de pacotes, é possível ver o relacionamento de dependência entre o *KBManager* e a classe Model do JENA.

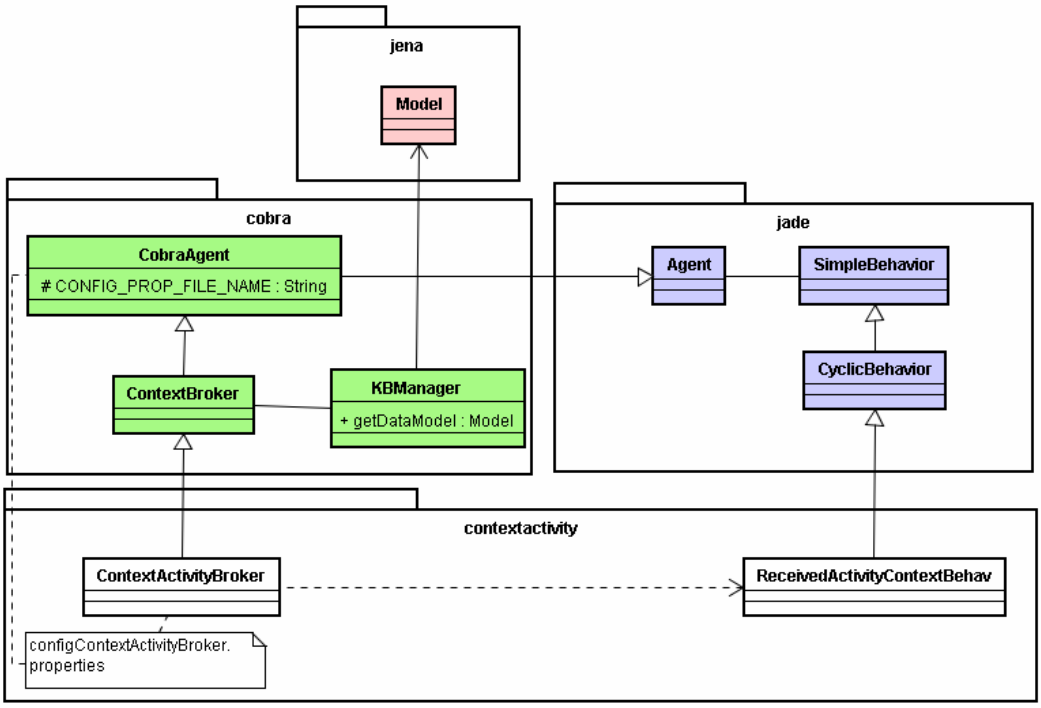

Figura 4.5 – Diagrama de Pacotes do A-CoBrA.

Como pode ser visto na figura 4.5, devido às sucessivas especializações entre *Agent*, *CobraAgent* e *ContextBroker*, o *ContextActivityBroker* também herda as características básicas de um Agente do framework JADE, o que permite a implementação de comportamentos, que no caso do A-CoBrA são cíclicos, à espera de mensagens dos agentes sensores. Com este propósito, foi projetado um comportamento chamado *ReceivedActivityContextBehav* responsável por receber, processar e responder às mensagens dos demais agentes sobre o conhecimento contextual de atividades.

Na Figura 4.6, é possível visualizar o diagrama de seqüência com a troca de mensagens entre um agente sensível ao contexto de atividades e o *Broker* de Contexto de Atividades. Neste caso, o *Broker* recebe uma mensagem com instâncias da ontologia de contexto e identifica que deve executar o comportamento cíclico responsável pelo recebimento de conhecimento contextual de atividades.

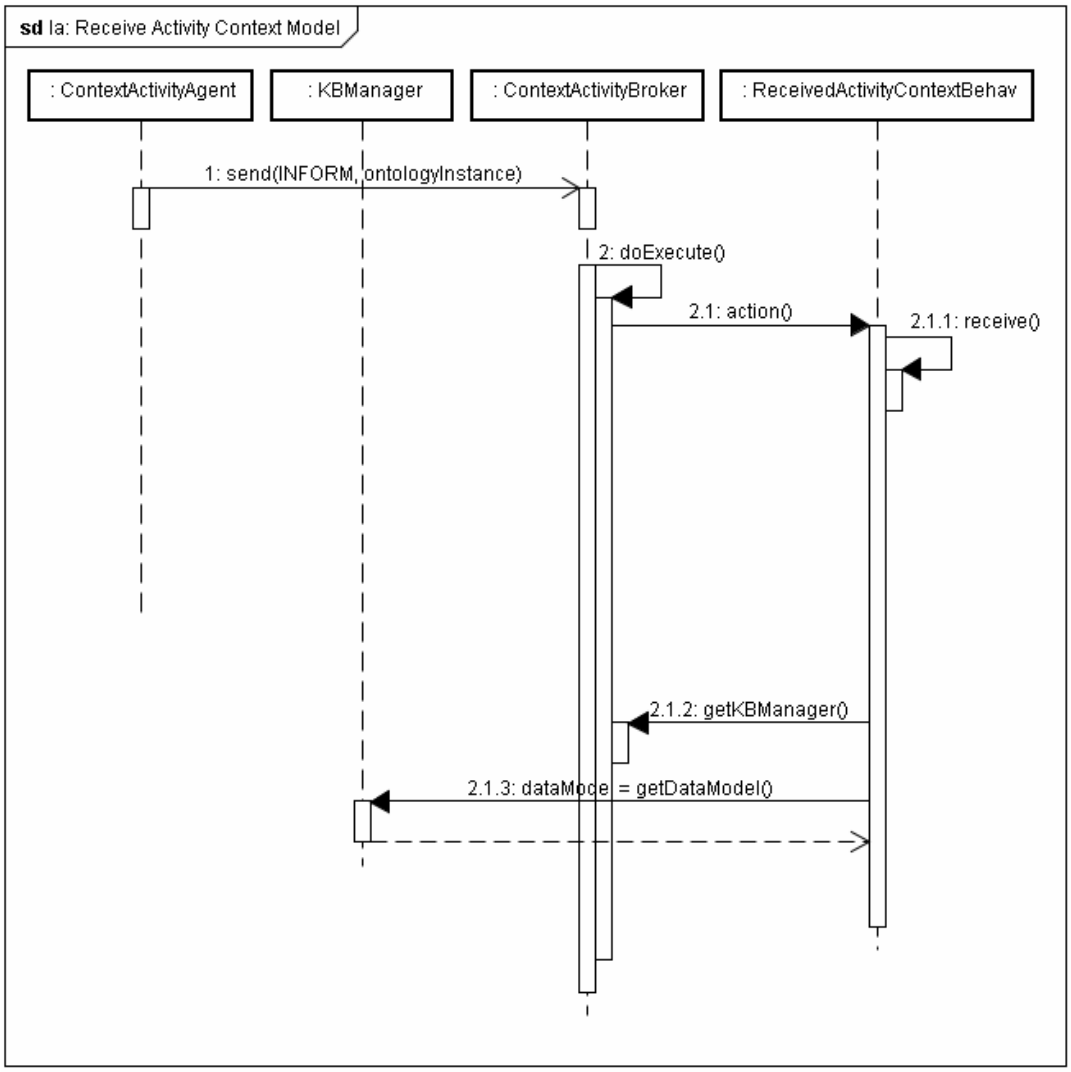

Figura 4.6 – Diagrama de seqüência para o recebimento de conhecimento contextual sobre atividades.

4.2.4 Representação de Contexto de Atividades de Processo

Como representação de contexto, é utilizada a ontologia de contexto proposta por NUNES (2007). Esta ontologia foi evoluída no sentido de incorporar novas classes que podem adicionar informações relevantes sobre as atividades que são realizadas em um processo de trabalho. No Apêndice A, é demonstrada a ontologia evoluída e adaptada, enquanto que na Tabela 4.1, são detalhadas as classes adicionadas que são independentes de domínio:

Tabela 4.1 – Conceitos adicionados à ontologia de contexto.

Conhecimento	Grau de saber sobre determinado assunto. Pode ser especializado em Conhecimento Básico, Intermediário ou Avançado.
Complexidade	Grau de dificuldade em determinado assunto. Pode ser especializado em Complexidade Baixa, Média ou Alta.
Raciocínio	Explicação racional para a forma como conduzida à execução de determinada atividade.

A classe Conhecimento tem relacionamento com a classe Individuo, pois todo Individuo pode ter um Conhecimento sobre determinado assunto que pode ser básico, intermediário ou avançado.

A classe Complexidade tem relacionamento com a classe Atividade, pois toda a atividade pode ter uma complexidade baixa, média ou alta.

Já a classe Raciocínio tem relacionamento com as classes Atividade e Contexto. Sendo que toda atividade pode produzir raciocínio para a sua execução e este pode ser de extrema relevância para posterior recuperação e entendimento da atividade. Como tem forte ligação com a atividade, que é o foco principal, e descreve informação relevante, Raciocínio também compõe o Contexto.

4.2.5 *Reasoning* de Contexto de Atividades de Processo

Reasoning de contexto é fundamental para o funcionamento de um sistema sensível a contexto. Segundo CHEN (2004), a implementação de *reasoning* utilizando uma abordagem baseada em regras é melhor em diversos aspectos em relação a uma abordagem em que o *reasoning* é implementado proceduralmente em uma linguagem de programação. As principais vantagens são:

i) Uma representação explícita de regras de *reasoning* pode ajudar a separar a lógica de *reasoning*, da lógica de implementação funcional, o que facilita a manutenção dos componentes de *reasoning* de contexto;

ii) Uma abordagem baseada em regras facilita a adição de modelos lógicos de conceitos gerais como tempo e espaço, o que pode ser útil para determinados tipos de *reasoning*;

iii) Quando regras de interpretação de contexto são representadas de forma explícita, técnicas de meta-*reasoning* podem ser desenvolvidas para detectar e resolver inconsistências de interpretações de contexto.

O *reasoning* de contexto do A-CoBrA utiliza o reasoner baseado em regras do JENA para realizar inferências. Na Figura 4.7, é possível visualizar o design das classes do mecanismo de *reasoning*, onde o comportamento que recebe as informações de contexto faz uso de uma classe fachada para realizar chamadas para um tratador. Este mecanismo de tratamento pode ser especializado para cada atividade ou pode utilizar um *reasoning* padrão. Portanto, o tratamento e raciocínio específico para cada atividade do processo é realizado por uma implementação de uma classe que implementa a interface *ActivityContextReasoningHandler*. Por conseguinte, as classes *ActivityConcreteHandler1* e *ActivityConcreteHandler2* devem ser substituídas por classes de instância da arquitetura.

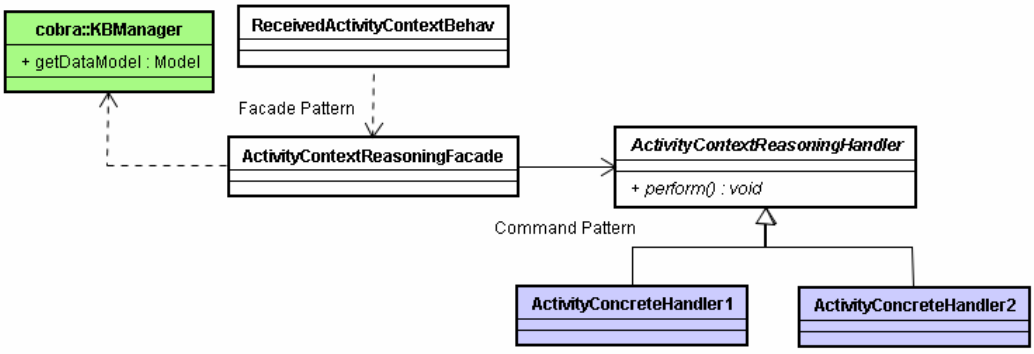

Figura 4.7 – *Handler* de *Reasoning* de Contexto de Atividades.

Na figura 4.8, é possível visualizar como é executado o comportamento de recebimento de conhecimento contextual até a chamada para a fachada do componente de *reasoning* de contexto de atividades. A utilização de uma fachada visa o reuso. Ao desacoplar o mecanismo de *reasoning*, ele pode ser reutilizado em outras soluções que não utilizem uma abordagem orientada a agentes ou o framework JADE.

O componente de *reasoning* é responsável por utilizar e garantir a consistência do modelo compartilhado de contexto de atividades. Para tanto, utiliza-se de mecanismos que garantem o acesso seguro para alterações no modelo (Figura 4.9), como incluir novas informações sobre contextos de atividades. Para a realização de inferências, é utilizado um dos tratadores específicos ou o padrão. O desenvolvimento de novos tratadores de *reasoning* deve ser realizado de acordo com as necessidades de *reasoning* para cada atividade do processo que está sendo apoiado na organização.

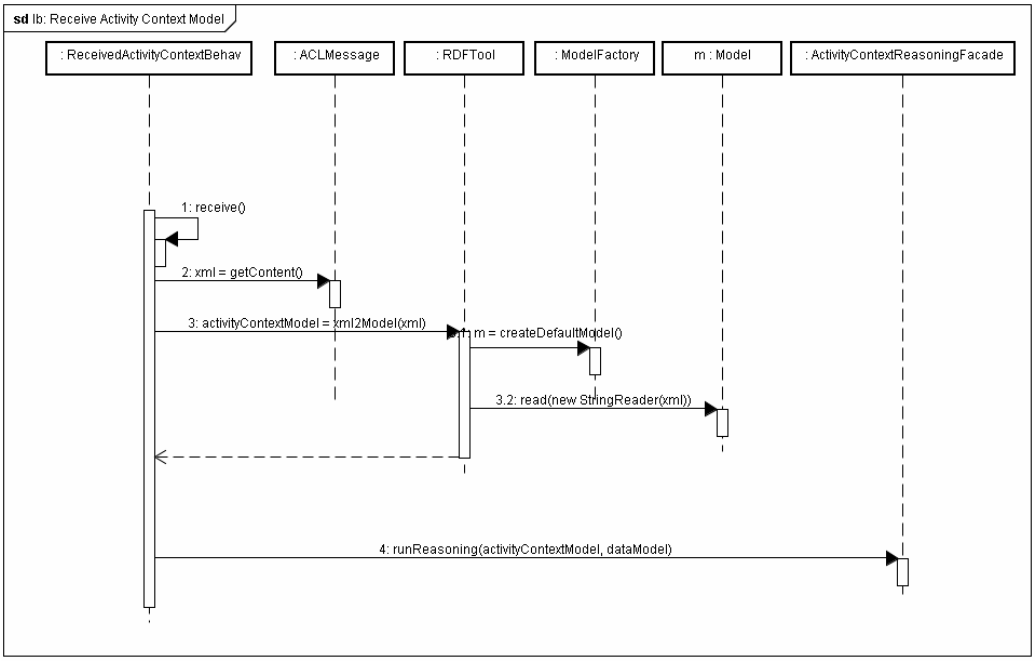

Figura 4.8 – Comportamento de recebimento de contextos de atividades até a chamada para o *reasoning* de contexto de atividades.

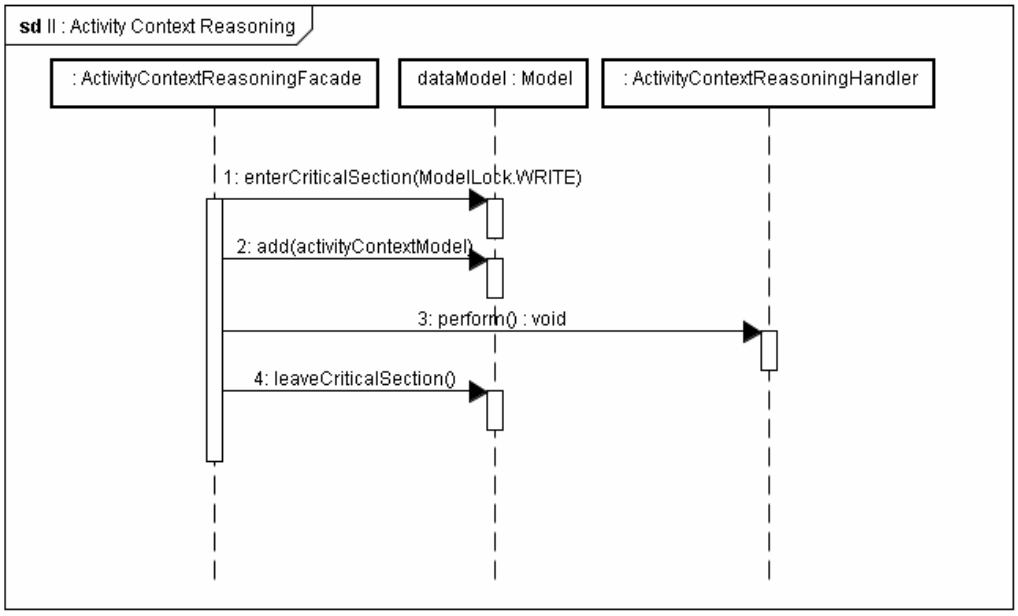

Figura 4.9 – *Handler* de *Reasoning* de Contexto de Atividades.

4.2.5.1 Regras de Reasoning de Contexto

Nesta seção, são apresentados exemplos de regras de *reasoning* de contexto para a ontologia de contexto. Na regra apresentada na Tabela 4.2, é realizada uma validação se um indivíduo tem competência para desempenhar determinado papel, enquanto que na Tabela 4.3 é apresentada um regra que valida se um indivíduo tem o perfil requerido. Essas regras podem ser utilizadas simplesmente para validação de conhecimento contextual, bem como podem servir para sugerir informações relevantes para um indivíduo que precisa assumir determinado papel e não tem perfil ou competências necessárias.

As regras foram especificadas utilizando a sintaxe disponibilizada pelo framework JENA para implementação de regras. Em adição, também foi utilizada a linguagem SWRL (2009), que atualmente é um padrão recomendado pela W3C para implementação de regras.

Definição da Regra I: um indivíduo só pode desempenhar um papel para o qual possui competência

Tabela 4.2 – Regra para validar se um indivíduo tem competência para desempenhar determinado papel

JENA RULE	**SWRL**
[definicaoDesempenhoDePapelBaseadoCo: (?i rdf:type acobra:Individuo) (?i acobra:temCompetencia ?c) (?c rdf:type acobra:Competencia) (?p acobra:requer ?c) (?p rdf:type acobra:Papel) → (?i acobra:podeDesempenhar ?p)]	acobra:Individuo(?i) ^ acobra:temCompetencia(?i, ?c) ^ acobra:Competencia(?c) ^ acobra:requer(?p,?c) ^ acobra:Papel(?p) → acobra:podeDesempenhar(?i,?p)

.

Definição da Regra II: um indivíduo só pode desempenhar um papel para o qual possui perfil

Tabela 4.3 – Regra para validar se um indivíduo tem perfil para desempenhar determinado papel

JENA RULE	**SWRL**
[definicaoDesempenhoDePapelBaseadoEm: (?i rdf:type acobra:Individuo) (?i acobra:temPerfil ?p) (?p rdf:type acobra:Perfil) (?pa acobra:temPerfil ?p) (?pa rdf:type acobra:Papel) → (?i acobra:podeDesempenhar ?pa)]	acobra:Individuo(?i) ^ acobra:temPerfil(?i, ?p) ^ acobra:Perfil(?p) ^ acobra:temPerfil(?pa, ?p) ^ acobra:Papel(?pa) → acobra:podeDesempenhar(?i, ?pa)

4.3 Instanciando a Arquitetura

Esta seção descreve como a arquitetura A-CoBrA pode ser instanciada considerando o uso da arquitetura especificamente para o contexto de atividades de processo. A-CoBrA também pode ser instanciada para outros tipos de contexto devido ao seu projeto ter sido desenvolvido considerando este requisito funcional. Para tanto, bastaria especializar o *Broker* de Contexto para o contexto desejado, criando em

conjunto seus tratadores de *reasoning*. Porém, como o objetivo deste trabalho é o contexto de atividades de processo, esta seção descreve em particular como instanciar a A-CoBrA para este fim. A instanciação deve ser feita no componente de *reasoning*. Para instanciar, bastaria adicionar os tratadores específicos ao *reasoning* dependente das atividades de processo, como por exemplo, tratador para uma atividade de planejamento de projeto (Figura 4.10). Tais tratadores devem saber inferir sobre as regras e repositório de contexto definidos na arquitetura.

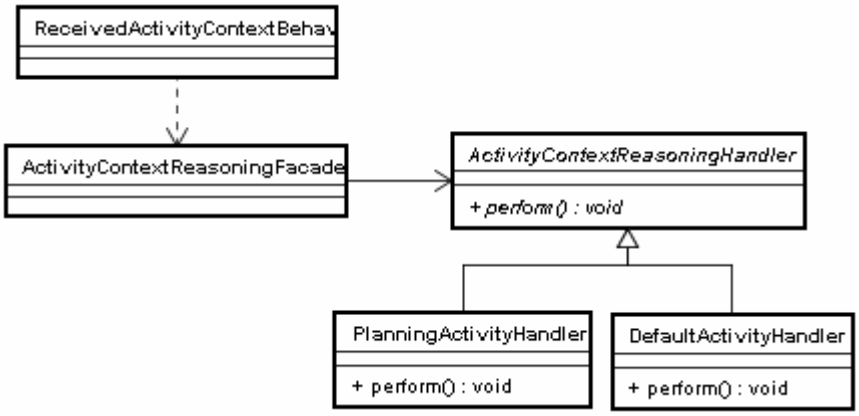

Figura 4.10 – Exemplo de instanciação do componente de *reasoning* de contexto de atividades.

Além disso, também é preciso que seja configurado o mapeamento entre as atividades de um processo e os **tratadores/manipuladores (MP)** desenvolvidos para realizar o *reasoning* específico para cada atividade. Essa configuração é flexível e pode ser feita nas configurações do *Broker* de Contexto de Atividades, definindo assim sobre que atividades ele é capaz de realizar o *reasoning*. Desta forma, os agentes sensores e sensíveis ao contexto de atividades podem realizar solicitações ao *Broker* de acordo com o contexto percebido em cada atividade.

A Figura 4.11 ilustra o diagrama de colaboração entre os agentes sensíveis ao contexto (os agentes sensores), o *Broker* de **Contexto de Atividades (CA)** e seus correspondentes manipuladores de *reasoning* que devem ser criados na instanciação da arquitetura. Na inicialização do sistema, o *Broker* de Contexto de Atividades deve se

registrar no Serviço de Registro de Manipuladores/Tratadores de *Reasoning* de Contexto. Este serviço segue o padrão FIPA de registro e busca de serviços entre agentes. Os agentes sensíveis ao contexto devem então buscar pelo *Broker* de Contexto de Atividades que oferece o serviço buscado no Serviço de Registro. Uma vez com o endereço do *Broker*, os agentes sensíveis ao contexto podem manter o contexto na memória organizacional.

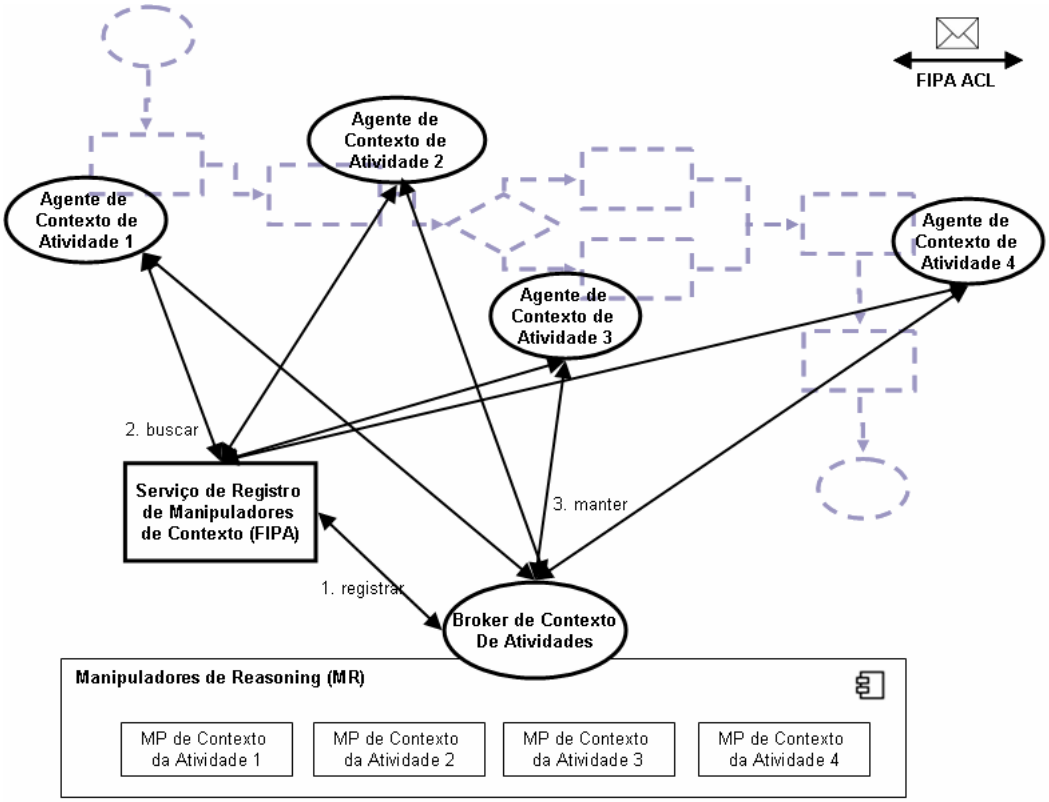

Figura 4.11 – Diagrama de colaboração como exemplo de instanciação.

Dependendo da complexidade do processo, pode ser necessário criar uma federação de *Brokers* de **Contexto de Atividades (CA)** (Figura 4.12). Para os agentes sensíveis ao contexto isto ficará transparente, uma vez que eles só precisam buscar pelos serviços de **manipulação de *reasoning* (MR)** de contexto de atividade, não importando quem exatamente oferece. Neste cenário, todos os *Brokers* da federação deverão se registrar no Serviço de Registro para que possam ser encontrados.

A utilização de uma federação de *Brokers* também é necessária nos casos em que não se quer depender de um único *Broker* central que pode falhar e comprometer todo o sistema. A federação de *Brokers* também pode tornar o sistema escalável e robusto.

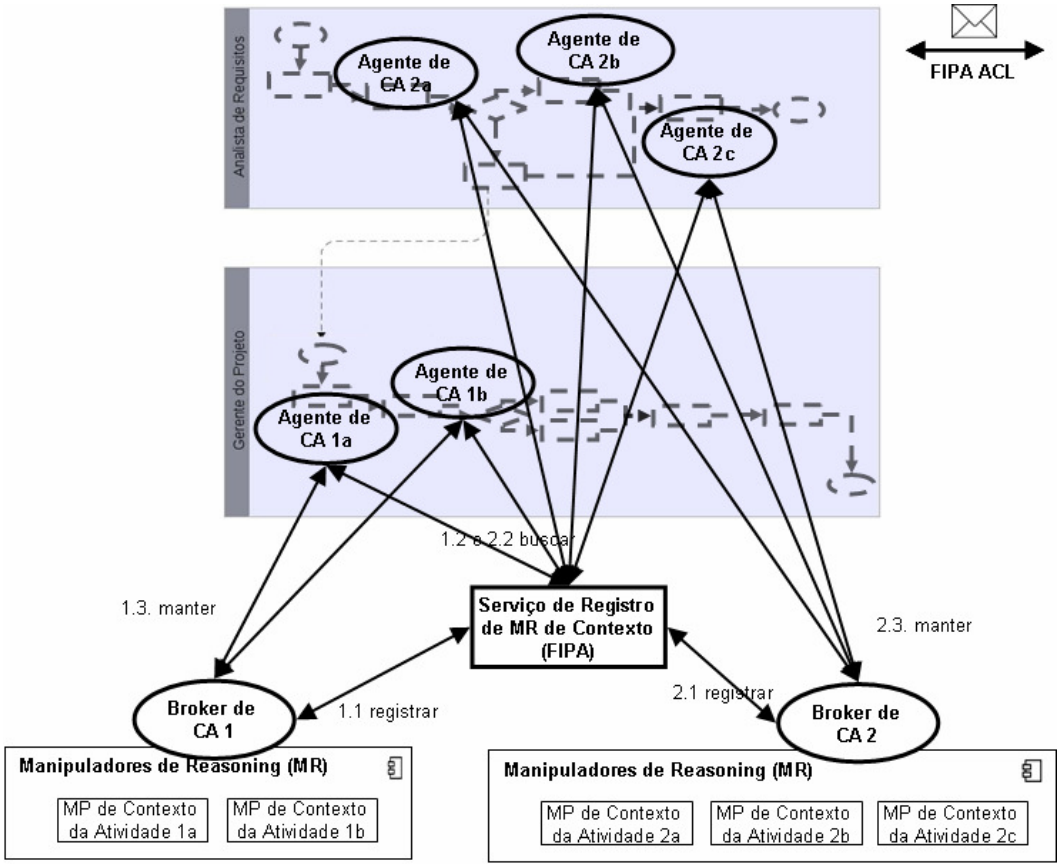

Figura 4.12 – Diagrama de colaboração como exemplo de instanciação com federação de *Brokers*.

Na Figura 4.12 é possível visualizar um exemplo de *Brokers* federados que registram os seus mecanismos de *reasoning* no Serviço de Registro de Contexto. Os agentes sensíveis ao contexto de atividades de cada processo buscam no Serviço de Registro e recebem a referência para o *Broker* de forma transparente, diminuindo assim o acoplamento entre os agentes.

4.4 Considerações Finais do Capítulo

Neste capítulo foi apresentada a arquitetura A-CoBrA. Esta arquitetura é uma refatoração e extensão da arquitetura CoBrA (CHEN, 2004) para que seja sensível ao contexto de atividades de processos de trabalho organizacionais.

A arquitetura foi contextualizada através de um problema fictício sobre um processo de aprovação de hipoteca simplificado e seus exemplos foram correlacionados com a arquitetura genérica proposta por SCHMOHL e BAUMGARTEN (2008), que também serviu de base para a sua descrição.

O fluxo de interações da arquitetura foi apresentado, segundo o modelo de gestão de conhecimento baseado em contexto proposto por NUNES (2007). Desta forma, foi possível apresentar como a arquitetura apóia o modelo por meio da utilização de agentes sensores, atuadores e principalmente pelo seu componente central, o *Broker* de Contexto de Atividades e seus mecanismos de *reasoning*.

De forma a abstrair detalhes de implementação e demonstrar os relacionamentos entre os componentes da arquitetura também foi apresentado um modelo conceitual que esclarece todas as suas dependências, relacionamentos e especializações. Posteriormente, foram discutidos os aspectos de projeto, implementação e instanciação, incluindo seus mecanismos de *reasoning*.

5 Avaliação da Arquitetura A-CoBrA: Uma Implementação do *Broker* de Contexto de Atividades para um Processo de Planejamento de Projetos

Este capítulo apresenta a implementação utilizada para avaliação da arquitetura A-CoBrA proposta. A implementação tem como objetivo principal avaliar o *Broker* de Contexto de Atividades. Para isso, foi escolhido um processo de planejamento de projetos. Desta forma, foi necessário: (i) definir o método de avaliação; (ii) descrever o cenário e modelar o processo; (iii) instanciar a arquitetura A-CoBrA; (iv) definir instâncias para a ontologia; (v) definir regras para o domínio; (vi) avaliar e analisar os resultados.

5.1 Metodologia de Avaliação

Segundo BERNDTSSON *et al*. (2008), muitas pesquisas em Ciência da Computação e Sistemas de Informação consistem no desenvolvimento de novas soluções, como por exemplo, uma nova arquitetura que resolve algum problema de um modo novo, e que tem alguma vantagem sobre soluções existentes. Em um trabalho deste tipo, quase sempre é necessário utilizar o método de pesquisa de implementação, a fim de verificar as vantagens e desvantagens da solução proposta. Portanto, o objetivo da implementação é demonstrar que a solução tem certas propriedades ou sobre certas condições se comporta de um modo específico. Também é importante verificar a validade e confiabilidade da solução implementada. Em outras palavras, é preciso avaliar que a implementação de fato é uma implementação da solução proposta

(validade) e que a implementação funciona bem o bastante para ser útil na pesquisa (confiabilidade).

Para MATTOSO (2004), a caracterização da contribuição determina a escolha de técnica ou método de avaliação a ser utilizado. No caso de arquiteturas, são indicados o desenvolvimento de protótipo, caso de uso e/ou modelo de custo, a depender do que se pretende verificar.

Como este trabalho propõe uma arquitetura como solução para um problema em aberto levantado, foi definida como forma de avaliação a implementação de um protótipo, a fim de verificar a viabilidade e vantagens oferecidas pela arquitetura. No entanto, no âmbito de um projeto de dissertação de mestrado, não seria possível implementar e avaliar todos os itens da arquitetura. Portanto, o escopo foi estabelecido de forma a avaliar a implementação do *Broker* de Contexto de Atividades e o seu mecanismo de *Reasoning*, os quais são considerados os componentes centrais da arquitetura.

De acordo com os objetivos descritos no capítulo 1, durante a implementação e avaliação realizadas, o trabalho foi observado sob os seguintes aspectos específicos:

- A arquitetura proposta pode ser estendida a partir da implementação existente da arquitetura CoBrA?
- A arquitetura proposta para a gestão de contexto de atividades oferece pontos de extensão e reutilização?
- A arquitetura oferece suporte à reutilização e inferência de contextos de atividades?

Para avaliar a instanciação e implementação da arquitetura A-CoBrA foi necessário:

- Descrever o cenário, modelar o processo de negócio e identificar atividades de gestão de contexto, bem como as informações de contexto que poderiam ser relevantes nesse cenário;
- Instanciar a arquitetura A-CoBrA para o processo de negócio modelado;
- Estender e definir instâncias para a ontologia de contexto, para o domínio do processo de negócio;
- Definir consultas e regras de inferência para o domínio do processo de negócio.

5.2 Descrição do Cenário e Processo de Negócio

Nesta seção, é descrito o cenário utilizado para a avaliação da arquitetura proposta. No caso, um sub-processo do domínio de gerenciamento de projetos de software, que foi adaptado dos passos indicados para planejamento de um projeto de software do processo OPENUP (2009). O processo é composto por atividades necessárias para o planejamento de um projeto de software, partindo do pressuposto que essa atividade é realizada por um Gerente de Projetos. Essa abordagem foi utilizada com o objetivo de flexibilizar e facilitar a ilustração do exemplo e avaliação, porém sem deixar de demonstrar a avaliação das propriedades fundamentais da arquitetura, como a inferência e recuperação de contextos de atividades.

Na Figura 5.1, é possível visualizar o diagrama em notação BPMN (2009) do processo, incluindo as atividades de captura de contexto que foram consideradas seguindo o proposto por NUNES (2007) e apresentados no Capítulo 2.

Quando um cliente realiza uma requisição para desenvolvimento de um software, um Analista de Requisitos deve receber essa requisição e realizar uma atividade de definição de requisitos (Definir Requisitos). Esta atividade deve gerar os seguintes

artefatos: Glossário (contém as definições de termos relevantes no domínio do software), Documento de Visão (contém um resumo dos principais requisitos) e Casos de Uso (contém o detalhamento de todos os requisitos, os fluxos principais e alternativos).

A partir dos requisitos definidos, o Gerente de Projetos deve definir a equipe (Definir Equipe) que irá trabalhar no projeto de acordo com as habilidades e competências necessárias para o seu desenvolvimento. Em seguida, a próxima atividade é a de estimar o tamanho do projeto (Estimar Tamanho do Projeto) utilizando uma lista de itens de trabalho que deve conter todas as possíveis atividades, em um nível macro, previstas para o projeto. Neste ponto, o sistema sensível ao contexto deve ser capaz de perceber, capturar e inferir informações de contexto sobre a atividade de estimar o tamanho do projeto. Para executar esta atividade, um Gerente de Projetos utiliza o seu conhecimento tácito sobre o domínio do software e sobre a complexidade das tarefas envolvidas para estimar, de forma grosseira, o tamanho do projeto. Neste caso, ele utiliza o conhecimento sobre experiências de planejamento de projetos anteriores para estimar o tamanho do projeto atual. Contudo, caso este gerente não possua experiência no domínio ou com esse tipo de projeto, como ele poderia realizar esse planejamento? O sistema sensível ao contexto de atividades deve inferir, a partir de contextos de planejamentos de projetos anteriores, informações úteis para o planejamento do projeto atual, como por exemplo:

- **Tempo necessário para absorver o conhecimento sobre o domínio do software;**
- **Principais riscos associados a projetos de software dessa natureza;**
- **Exemplos de planejamentos realizados anteriormente para softwares do mesmo domínio ou com tecnologias semelhantes.**

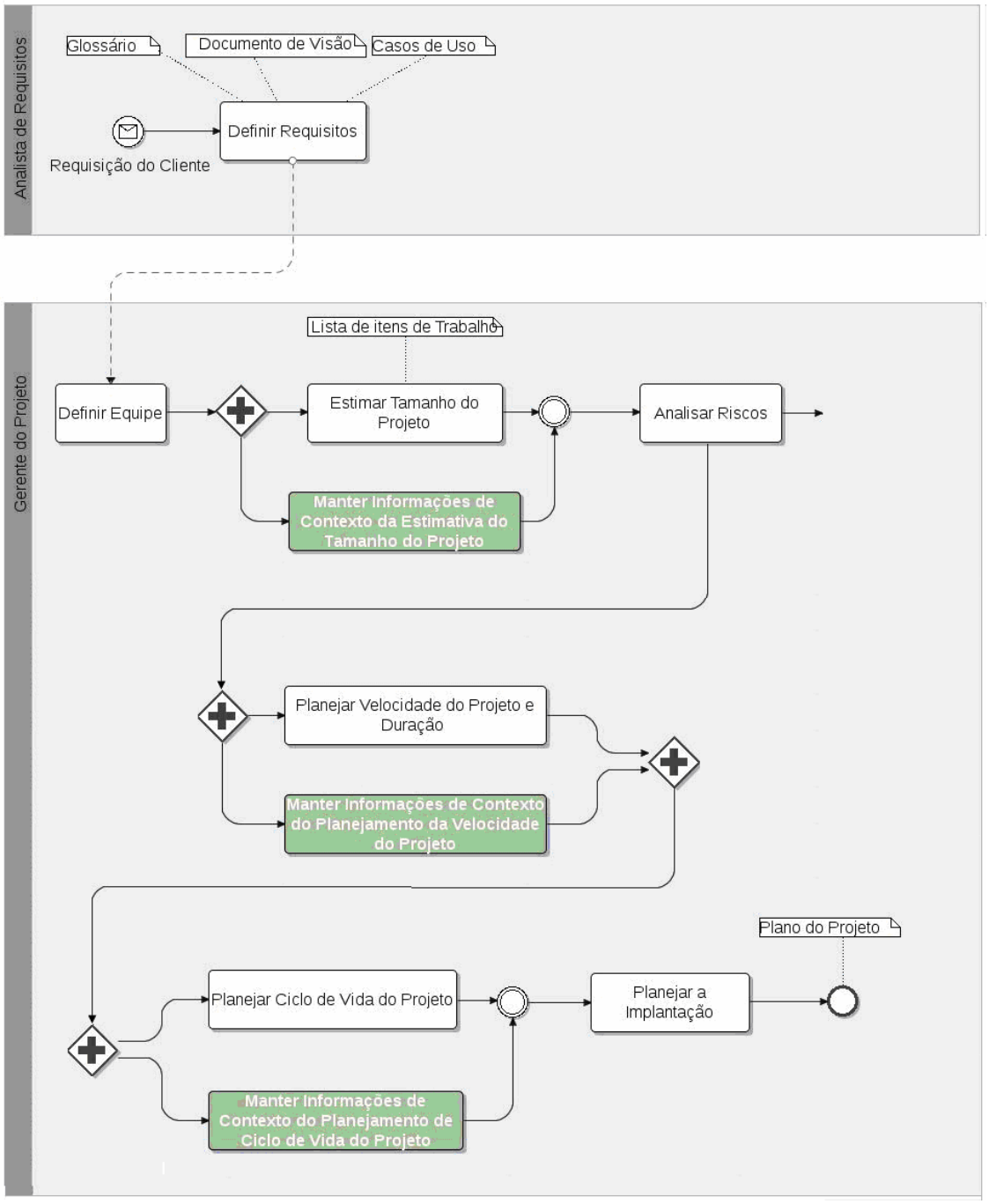

Figura 5.1 – Processo de Planejamento de Projeto com Captura de Contexto.

Portanto, no processo de planejamento de projetos ilustrado na Figura 5.1, é possível visualizar a atividade de gestão das informações de contextos da estimativa do tamanho do projeto (Manter Informações de Contexto da Estimativa do Tamanho do

Projeto). Esta atividade foi inserida no processo e ocorre em paralelo à atividade Estimar Tamanho do Projeto, por meio dos componentes instanciados para o problema, a partir da arquitetura baseada em contexto de atividades. Nesta atividade em especial, as informações de contexto para a estimativa do tamanho do projeto podem ser discussões entre Gerentes de Projeto e os clientes (*stakeholders*), para determinar o que é possível realizar de acordo com as restrições e limitações do projeto.

Posteriormente, é realizada a atividade de Análise de Riscos, onde o Gerente de Projetos avalia os riscos inerentes ao projeto por meio de uma análise de risco qualitativa, a fim de determinar a gravidade de cada risco. Neste processo, novamente um Gerente de Projetos experiente pode se utilizar do seu conhecimento tácito para determinar e analisar os riscos, de acordo com as informações levantadas até esta etapa do projeto. No entanto, para um Gerente de Projetos, deve ser relevante conhecer informações de contexto sobre análises de riscos realizadas para outros projetos de software do mesmo domínio e complexidade. Neste caso, as informações de contexto poderiam ser as razões pelas quais riscos de projetos similares foram aceitos ou não, dado que não é possível planejar ações para todos os riscos de um projeto.

Na atividade de planejamento da velocidade do projeto e duração (Planejar Velocidade do Projeto e Duração) são definidos os tamanhos das interações dentro do projeto e o número de itens em cada iteração. Este número depende, fundamentalmente, da velocidade da equipe e da estimativa de tempo para cada item de trabalho. Portanto, a velocidade do projeto é mensurada pelo número de tarefas que são realizadas a cada iteração. Sendo que cada tarefa tem uma estimativa de tamanho que tem relação com a complexidade e esforço estimado para realizar determinada tarefa.

Normalmente, na atividade supracitada, as interações não seriam detalhadas a ponto de especificar os responsáveis por cada atividade. No entanto, nesse processo

adaptado, para fins de simplificação, assumimos que as interações devem ser detalhadas no plano do projeto e, portanto, para cada item de trabalho deve ser especificado o seu responsável. Nesse caso, como informações de contexto, poderíamos ter:

- **Raciocínio utilizado pelo gerente de projeto para decidir sobre qual é o melhor recurso para executar determinada tarefa.**

Já a atividade de planejamento do ciclo de vida do projeto (Planejar Ciclo de Vida do Projeto) tem como objetivo organizar as interações em um conjunto de fases, onde cada fase deve terminar com um marcador. Este marcador é importante para que os clientes possam controlar os recursos financeiros, escopo, risco e outros aspectos relevantes do projeto. Para esta atividade, a atividade paralela de gestão de contexto, poderia gerir as seguintes informações:

- **Discussões sobre ciclos de vida de projetos anteriores;**
- **Histórico de marcadores significativos para os clientes em outros projetos.**

Por fim, a atividade de planejamento da implantação é executada para planejar a estratégia para implantar o software. O plano do projeto deve capturar os objetivos da implantação e as datas em que as versões devem ser liberadas. Para esta atividade, a gestão de contexto deve se preocupar com:

- **Raciocínio sobre as datas de planejamentos de implantações anteriores.**

O produto final do processo é o plano do projeto que é um artefato passível de persistência na memória organizacional com a sua devida associação com as informações de contexto que possibilitam a gestão completa do conhecimento contextual e seus artefatos gerados.

5.3 Instanciação da Arquitetura

Esta seção descreve como a arquitetura A-CoBrA proposta foi instanciada considerando o uso para o processo de Planejamento de Projetos. Para tanto, especializou-se o *Broker* de Contexto de Atividades, criando em conjunto seus tratadores de *reasoning* para cada uma das atividades do processo de Planejamento de Projetos.

A instanciação adicionou tratadores específicos ao *reasoning* de cada atividade, como por exemplo, tratador para a atividade de planejamento da velocidade e tamanho do projeto (Figura 5.2). Tais tratadores sabem inferir sobre o repositório de contexto, utilizando consultas e regras, de acordo com as informações de contexto de cada atividade do processo.

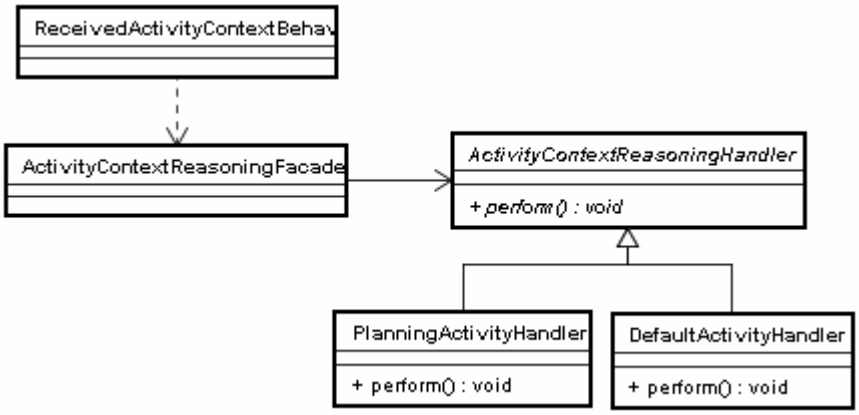

Figura 5.2 – Exemplo de instanciação do componente de *reasoning* de contexto de atividades para a atividade de planejamento da velocidade e tamanho do projeto.

Nas configurações do *Broker* de Contexto de Atividades foi feito o mapeamento entre as atividades do processo e os tratadores específicos para cada atividade, definindo assim qual tratador é responsável por realizar o *reasoning* de cada atividade.

5.4 Instanciação e Extensão da Ontologia de Contexto para o Domínio do Processo de Negócio

Esta seção descreve as etapas que foram realizadas para instanciação e extensão da ontologia de contexto para o domínio de planejamento de projetos de software.

5.4.1 Descrição de um Possível Cenário para Instâncias da Ontologia

O cliente José solicita que seja desenvolvido o sistema THOR. Esta requisição chega ao Analista de Requisitos Pedro, responsável por definir os requisitos e gerar o Glossário, Documento de Visão e Casos de Uso. Após a definição dos requisitos, o Gerente de Projetos João precisa planejar o projeto de desenvolvimento do sistema THOR. A primeira etapa de planejamento é a definição da equipe. Para este projeto, João define que irá contar com três desenvolvedores: Marcio, Fabio e Paulo. Para estimar o tamanho do projeto, João elabora a lista de itens de trabalho (atividades) do projeto THOR. Alguns itens de trabalho são: Definir Arquitetura do Sistema, Desenvolver Funcionalidade X, Desenvolver Funcionalidade Y, Testar Funcionalidade X e Testar Funcionalidade Y. A seguir, João pensa nos riscos associados ao projeto, como o conhecimento dos desenvolvedores sobre a área de negócio do sistema THOR e a integração com os sistemas legados, o que depende de outras equipes.

No planejamento da velocidade e duração do projeto, João define o tempo para cada atividade e determina qual desenvolvedor deve executá-la, de acordo com o conhecimento que João possui do grupo. Por exemplo, João sabe que Márcio é o mais experiente em integração de sistemas, portanto, designa Márcio para o desenvolvimento da Funcionalidade X, que possui integração com diversos sistemas legados. Essa informação vem do conhecimento tácito de João sobre a equipe e foi adquirida ao gerenciar outros projetos em que Márcio esteve presente.

Para o planejamento do ciclo de vida do projeto, João organiza as atividades em grupos e define fases para o projeto. Para cada uma das fases definidas, João determina marcadores e entregas a serem realizadas. Ao final, João planeja a implantação do sistema definindo as datas para liberação das versões do projeto no ambiente de produção.

Como visto anteriormente, mas agora de forma resumida, na atividade de Planejamento da Velocidade e Duração do projeto, poderíamos ter as seguintes informações de contexto:

- Raciocínio utilizado pelo gerente de projeto para decidir sobre qual é o melhor recurso para executar determinada atividade;
- Principais riscos associados a projetos de sistemas dessa natureza;
- Tempo necessário para uma pessoa absorver o conhecimento sobre o domínio do sistema;
- Tempo necessário para uma pessoa absorver o conhecimento sobre as tecnologias e frameworks necessários para o desenvolvimento do sistema;
- Velocidade de desenvolvimento de um recurso executando determinado papel;
- Velocidade da cada desenvolvedor em atividades anteriores;
- Tempo recomendado para uma atividade de acordo com a velocidade do desenvolvedor e complexidade da atividade;
- Exemplos de planejamentos realizados anteriormente para sistemas do mesmo domínio e/ou tecnologias semelhantes.

5.4.2 Descrição da Ontologia de Contexto Estendida para o Domínio

Para que a arquitetura proposta suporte o cenário descrito, é necessário estender a ontologia de contexto para o domínio do processo de negócio modelado. Nesta

ontologia foram adicionadas as classes de domínio de Risco, Projeto, Tecnologia e Domínio. As demais classes da ontologia de contexto foram aproveitadas tanto para representar informações de contexto, quanto para representar informações de domínio de Planejamento de Projetos de Software, que também inclui conceitos como Atividade, Artefato, Individuo, Papel e outros. A seguir, é apresentada a estrutura de classes e propriedades da ontologia que são relevantes para o cenário descrito.

- **Artefato:** nome, tipo, descrição;
- **Atividade:** nome, descrição;
- **Ator:** nome;
- **Competência:** conhecimento tecnológico, liderança, conhecimento do negócio;
- **Complexidade**:
    - Complexidade Baixa;
    - Complexidade Média;
    - Complexidade Alta;
- **Conhecimento:**
    - Conhecimento Básico;
    - Conhecimento Intermediário;
    - Conhecimento Avançado;
- **Contexto:**
- **Discussão**: assunto;
- **Domínio:** nome, complexidade, tempo de aprendizado;
- **Indivíduo:** formação, local de trabalho, horário de trabalho, interesses, experiências;
- **Interação:** comportamento, sentimento;
- **Mensagem**: conteúdo, tipo da mensagem;

- **Papel:** nome, função;
- **Perfil:** característica;
- **Procedimento**: nome, descrição;
- **Processo:** nome; meta;
- **Projeto:** nome, descrição, complexidade;
- **Raciocínio:** descrição;
- **Riscos:** descrição;
    - Risco Alto;
    - Risco Médio;
    - Risco Baixo.
- **Tecnologia**: nome, descrição, complexidade, tempo de aprendizado;
- **Tempo:** prazo, data e hora de início, data e hora de fim;

A ontologia foi estendida utilizando o software PROTÉGÉ OWL (2009) e o modelo foi gerado utilizando o plugin ONTOVIZ (2009). No Apêndice A é possível ver os relacionamentos entre as classes e as suas dependências no modelo.

5.4.3 Instâncias para as Classes e Atributos da Ontologia no Cenário Descrito

A partir do cenário descrito e da ontologia estendida é possível definir um conjunto de instâncias para as classes da ontologia, de modo que seja possível avaliar a construção do protótipo de arquitetura. A seguir, são apresentados exemplos de instâncias, geradas arbitrariamente, de propriedades para algumas das classes da ontologia estendida.

- **Processo:** Planejamento do Sistema THOR; Gerar o plano do projeto;
- **Projeto:** Projeto_THOR; Sistema Especialista; Alta;
- **Domínio:** Saúde; Alta; 6 meses;

- **Tecnologias**: JAVA; linguagem de programação; média; 6 meses;
- **Atividade:** Planejar Velocidade do Projeto e Duração; Esta atividade estima os tempos e define quem deve executar cada atividade do projeto;
- **Procedimento**: Como estimar o tempo para cada atividade; Observe a complexidade, tamanho e velocidade do desenvolvedor;
- **Raciocínio:** Márcio foi escolhido para a atividade de integração por sua senioridade e experiência com esse tipo de atividade de alto risco;
- **Riscos:** Integração com sistemas legados; alto;
- **Tempo:** 8 horas para realizar o planejamento; 09/09/2009 - 09:00; 09/09/2009 - 12:00;
- **Artefato:** Plano do projeto THOR; Plano de projeto; Planejamento do projeto THOR;
- **Papel:** Gerente do Projeto; Analista de Requisitos;
- **Ator:** João;
- **Indivíduo:** Ciência da Computação; Escritório da empresa; 08:00 às 18:00; Metodologias ágeis; CRM, SAP, Supply Chain;
- **Competência:** Avançado, Intermediário e Básico;
- **Interação:** João tem ótimo relacionamento com o cliente ao negociar prazos; As suas interações são sempre positivas;
- **Discussão**: Prazo para implantação da primeira funcionalidade do projeto;
- **Mensagem**: Alerta sobre atrasos na entrega do plano do projeto; alerta;

5.4.4 Consulta para Recuperar Instâncias

Na Tabela 5.1, é apresentado um exemplo de consulta que permite a recuperação de instâncias sobre a ontologia estendida. Nesta consulta, escrita na linguagem RDQL (2004) e suportada pelo CoBrA, são utilizados os conceitos de Atividade, Projeto,

Raciocínio e Risco para recuperar conhecimento de domínio e contexto relevante sobre uma atividade de Planejamento de Velocidade do Projeto e Duração. Na avaliação realizada, os resultados dessa consulta são devolvidos como resposta pelo *Broker* de Contexto de Atividades para o agente que enviou informações sobre a atividade atual.

Tabela 5.1 – Consulta RDQL para recuperar informações de domínio relevantes para a atividade de Planejar Velocidade do Projeto e Duração.

```
SELECT ?nome_atividade, ?nome_projeto, ?descricao_raciocinio, ?risco_descricao
WHERE
(acobra:Planejar_Velocidade_do_Projeto_e_Duracao, acobra:Nome, ?nome_atividade)
(acobra:Planejar_Velocidade_do_Projeto_e_Duracao, acobra:produz, ?raciocinio)
(acobra:Planejar_Velocidade_do_Projeto_e_Duracao, acobra:faz_parte, ?projeto)
(?projeto, rdf:type, acobra:Projeto)
(?projeto, acobra:Nome, ?nome_projeto)
(?projeto, acobra:possui_Risco, ?risco)
(?risco, acobra:Descrição, ?risco_descricao)
(?raciocinio, rdf:type, acobra:Raciocinio)
(?raciocinio, acobra:Descrição, ?descricao_raciocinio)
USING acobra FOR <http://localhost/ontologies/acobra.owl#>
```

5.5 Definição de Regra de Inferência de Domínio

Nesta seção, é apresentada uma das regras de inferência de domínio que foram implementadas para o cenário descrito anteriormente. Na regra apresentada na Tabela 5.2, é realizada uma análise de risco simplificada para um projeto de software, onde se levam em conta somente a complexidade do projeto e o conhecimento do grupo ou equipe que irá trabalhar no projeto. A regra foi implementada utilizando a sintaxe disponibilizada pelo framework JENA (2009) para implementação de regras. No entanto, para fins da validação, a linguagem SWRL (2009) foi utilizada devido ao suporte oferecido na ferramenta Protégé para implementação e validação nessa linguagem, que atualmente é um padrão recomendado pela W3C.

Descrição da regra: Se um projeto tem complexidade alta e todos individuos do grupo que fazem parte do projeto tem conhecimento básico então o risco do projeto é alto.

Tabela 5.2 – Regra para análise de risco sobre a ontologia de contexto estendida para o domínio de gerenciamento de projetos de software.

JENA RULE	**SWRL**
[calculoRiscoAlto: (?p rdf:type acobra:Projeto) (?p acobra:temComplexidade ?c) (?c rdf:type acobra:Complexidade_Alta) (?i rdf:type acobra:Individuo) (?i acobra:constitui ?g) (?g rdf:type acobra:Grupo) (?p acobra:temGrupo ?g) (?i :temConhecimento ?k) (?k rdf:type acobra:Conhecimento_Basico) (?p :possui_Risco ?r) → (?r type acobra:Risco_Alto)]	acobra:Projeto(?p) ^ acobra:temComplexidade(?p , ?c) ^ acobra:Complexidade_Alta(?c) ^ acobra:Individuo(?i) ^ acobra:constitui(?i, ?g) ^ acobra:Grupo(?g) ^ acobra:temGrupo(?p, ?g) ^ acobra:temConhecimento(?i, ?k) ^ acobra:Conhecimento_Basico(?k) ^ acobra:possui_Risco(?p, ?r) → acobra:Risco_Alto(?r)

5.6 Resultados

A avaliação começou com a refatoração da implementação da arquitetura CoBrA para suportar a extensão proposta, incluindo alteração de privacidade de métodos e evolução de funcionalidades de configuração para que se tornassem extensíveis. Em seguida, foram implementadas todas as classes e arquivos de configuração da arquitetura A-CoBrA proposta.

O ambiente demo da arquitetura CoBrA possui um agente cliente denominado *ScriptPlay*, que pode ser utilizado para testes do *Broker* de Contexto. Esse cliente foi alterado para que pudesse se comunicar com o *Broker* de Contexto de Atividades, estendido a partir da arquitetura CoBrA.

A avaliação do *Broker* de Contexto de Atividades foi realizada com envio de mensagens XML que fazem referências para instâncias da ontologia de contexto estendida. A transmissão dessas mensagens simula o envio de contexto de alto nível, já em representação ontológica OWL.

A instanciação e extensão da ontologia de domínio para um cenário de um planejamento de projeto de software foi o primeiro passo para definição das instâncias

da ontologia. A ferramenta Protégé possui a funcionalidade de salvar as instâncias e esquema da ontologia em formato OWL. A partir daí, o esquema e instâncias foram separados para que fosse possível simular o envio das instâncias para o *Broker* de Contexto de Atividades utilizando o agente ScriptPlay.

A ontologia estendida também serviu de base para gerar as classes JAVA, que são utilizadas para interagir com a representação da ontologia instanciada em uma classe de modelo do framework JENA. Para tanto, foi utilizada uma ferramenta chamada *schemagen* que torna possível a criação dessas classes de forma automatizada.

No *Broker* de Contexto de Atividades, foram implementados os comportamentos necessários para recebimento das mensagens, como por exemplo o comportamento *ReceivedActivityContextBehav*, responsável por receber as mensagens de contexto sobre atividades. O framework JENA foi utilizado para instanciação de modelos a partir das mensagens e para navegação e consultas sobre as instancias representadas pelos modelos. Foram realizadas consultas em RDQL para identificação da atividade sobre a qual as instâncias de contexto se referiam e para recuperar informações relevantes de contexto sobre a atividade atual.

O Protégé também foi utilizado para validação das regras de contexto e de domínio em linguagem SWRL e as regras foram transformadas manualmente para a linguagem de regras suportada pelo framework JENA. Após a transformação, as regras foram inseridas nos arquivos de regras da arquitetura A-CoBrA e foram implementados os seus mecanismos de utilização e manipulação, utilizando a máquina de inferência de propósito geral do framework JENA.

Os resultados alcançados com a implementação foram satisfatórios, dado que, de acordo com a metodologia de avaliação, a implementação demonstrou a viabilidade da extensão da arquitetura CoBrA para a arquitetura A-CoBrA proposta. Em adição, a

infra-estrutura herdada e estendida da arquitetura CoBrA facilitou a avaliação de consultas e regras de inferência de contexto e de domínio. Os padrões de fachada e os *handlers* específicos por atividade, na prática, demonstraram flexibilidade e estensibilidade para a adição de variados *handlers*, um para cada atividade específica, quando necessário.

5.7 Considerações Finais do Capítulo

Este capítulo descreveu como foi feita a avaliação da arquitetura proposta e serve como um guia para outras implementações e instanciações para os outros domínios ou processos de negócio. Nele foram definidas a metodologia de avaliação, a descrição do cenário e processo de negócio, a instanciação da arquitetura e extensão da ontologia para o domínio de planejamento de projetos de software e a definição de regras de inferência específicas ao domínio. Também foram apresentados os resultados alcançados a partir da implementação da arquitetura utilizando o framework JENA e da implementação de consultas e regras de inferência de contexto e de domínio.

De um modo geral, a implementação atendeu ao propósito esperado de prototipar e avaliar as características esperadas pela arquitetura, como estensibilidade e sensibilidade ao contexto de atividades. No próximo capítulo serão apresentadas as conclusões finais do trabalho e um resumo de todas as contribuições.

6 Conclusões

A criação de mecanismos que facilitem a reutilização do conhecimento e a colaboração entre as pessoas durante a execução de suas atividades pode ser importante para uma organização, ao passo que estimula a troca de informações, idéias e experiências sobre atividades, que estão inseridas e são realizadas continuamente dentro de um processo de trabalho.

Em (NUNES, 2007) foi proposto um modelo conceitual de Gestão de Conhecimento que é integrado ao ambiente de trabalho, de forma a tratar as situações e especificidades ocorridas durante a execução de atividades, em um processo de trabalho. Dado o alto nível de abstração do modelo, surgiu a necessidade de se investigar como esse modelo poderia ser implementado por uma arquitetura, que pudesse oferecer suporte computacional ao reuso de conhecimento relacionado com as atividades. Assim, o problema tratado nesta dissertação foi criar uma arquitetura baseada em contexto de atividades para suportar a gestão de conhecimento em processos de trabalho.

O processamento, raciocínio e compartilhamento de conhecimento contextual são características importantes para a arquitetura, visto que deve ser possível recuperar o conhecimento contextual de atividades intrínsecas a um processo de trabalho.

Como resultados parciais, foram alcançados:

- Análise das arquiteturas para sistemas sensíveis ao contexto propostas na literatura até o momento, sem distinção por formas de representação ou

definição de contexto, buscando arquiteturas que de fato aproximaram-se do nível de implementação, quer seja com a implementação de middlewares, frameworks ou arquiteturas com componentes físicos;

- A identificação de um padrão existente entre as arquiteturas e a sua utilização para a definição de uma arquitetura reutilizável e instanciável para a gestão de contexto e conhecimento em processos de trabalho;
- Instanciação e avaliação da arquitetura proposta. Para isto foi necessário:
    - Modelar o processo do domínio de planejamento de projetos de software escolhido;
    - Estender e Instanciar a ontologia definida por NUNES (2007) para o processo modelado;
    - Construir regras de inferência sobre o conhecimento contextual e sobre o conhecimento do domínio;
    - Avaliar a arquitetura com a simulação e execução no domínio escolhido.

A seguir são apresentadas as contribuições do trabalho e os trabalhos futuros.

6.1 Contribuições

As principais contribuições deste trabalho são listadas a seguir:

- Definição de uma arquitetura baseada em contexto de atividades para implementar um cenário de suporte a GC tal como proposto por NUNES (2007) de uma forma estruturada, extensível e escalável.
- A evolução, através da inclusão de novas classes de contexto, na representação ontológica de contexto proposta por NUNES (2007)

- A definição e avaliação da arquitetura baseada em contexto de atividades para gestão de conhecimento em processos de trabalho

6.2 Trabalhos futuros

Durante o decorrer da pesquisa para este trabalho de dissertação foram identificados alguns possíveis trabalhos futuros, que são indicados a seguir:

- Pesquisa e evolução da arquitetura em relação às questões de armazenamento na memória organizacional;
- Pesquisas sobre mineração de informações na memória organizacional utilizando a representação ontológica de contexto;
- Pesquisas sobre representação dos cenários e processos utilizando grafos contextuais (BRÉZILLON e POMEROL, 2001);
- Pesquisas sobre estratégias de *meta-reasoning* de contexto de acordo com os tipos diferentes de informações de contexto de atividade, ou seja, incorporação de *meta-reasoning* de contexto de atividade como *frozen spots* (partes fixas) da arquitetura;
- Pesquisas sobre utilização de estratégias de tolerância à falhas no elemento central da arquitetura, o *Broker* de Contexto de Atividades. Apesar de os manipuladores de raciocínio de contexto de atividade serem distribuídos por atividade, o acesso a eles é feito de forma centralizada. Foi visto que a arquitetura poderia ser instanciada com a criação de uma federação de *Brokers*, entretanto, mecanismos de tolerância a falhas tal como monitoramento e replicação ainda precisariam ser desenvolvidos.
- Pesquisas avaliativas, com estudos de casos em organizações, o que inclui:

- Pesquisas sobre agentes sensores utilizando os padrões de captura definidos por NUNES (2007);
- Pesquisas sobre agentes atuadores e as formas de apresentação e visualização de contexto.

Considerando este último item sobre estudos de casos em organizações, uma possível direção para deixar a proposta independente de domínio, seria integrar os agentes sensores e atuadores em ambientes de automação de processos. Em particular para o problema usado como avaliação da arquitetura, um projeto de desenvolvimento de software, existem ambientes para automação de processos de software presentes na literatura. Eles são genericamente denominados *Process-Centered Software Engineering Environments* – PSEEs (LIMA REIS, 2003) e possibilitam a coordenação das atividades de equipes de desenvolvimento de software, acompanhamento dos prazos e utilização de recursos, além de facilitar a reutilização de boas práticas gerenciais a partir de diferentes projetos já concluídos. Uma vez que o ambiente é executável, os agentes poderiam ser integrados ao ambiente sem muito esforço de implementação. Porém pesquisas devem ser feitas para identificar a melhor forma de integração.

Por fim, *Business Process Management Systems* (BPMS) foram introduzidos com o objetivo de prover controle para definir e coordenar a execução de processos de negócio. BPMS apóiam a automatização de processos de negócio. O ciclo de vida de automatização inicia com a definição do processo. Em seguida, a definição é registrada em um BPMS. Para executar um processo, o sistema cria uma instância de processo e então coordena, monitora e registra sua execução. O registro pode ser analisado, podendo gerar uma definição aprimorada do processo (VAN DER AALST *et al*., 2003). Da mesma forma como pesquisas de integração podem ser feitas em um PSEE, pesquisas podem ser feitas para integração em um BPMS.

Referências

ABECKER, A., DECKER, S., “Organizational Memory: Knowledge Acquisition, Integration and Retrieval Issues”. In: *Proceedings of XPS-99. Springer LNAI 1570, p.113-124, 1999.*

ACKERMAN, M.S., “Definitional and contextual issues in organizational and group memories”. In: *Proceedings of Twenty-seventh IEEE Hawaii International Conference of System Sciences (HICSS 94), 191-200, 1994.*

ALAVI, M., LEIDNER, D., “Knowledge Management Systems: Emerging Views and Practices from the Field”. In: *Proceedings of 32nd Hawaii International Conference on System Sciences, Hawaii, 1999.*

ARAUJO, R. M., BRÉZILLON, P., “Reinforcing Shared Context to Improve Collaboration”, *Revue d´Intelligence Artificielle, França, v. 19, n. 3, p. 537-556, 2005.*

BALDAUF, M., DUSTDAR, S., ROSENBERG, F., “A Survey On Context-Aware Systems”, *International Journal of Ad Hoc and Ubiquitous Computing, 2(4), 63-277, Inderscience Publishers, 2007.*

BARDRAM, J., "The Java Context Awareness Framework (JCAF) – A Service Infrastructure and Programming Framework for Context-Aware Applications". In: *Hans Gellersen, Roy Want, and Albrecht Schmidt, editors, Proceedings of the 3rd International Conference on Pervasive Computing (Pervasive 2005), volume 3468 of Lecture Notes in Computer Science, pages 98–115, Munich, Germany, May 2005. Springer Verlag.*

BARTLETT, C., "McKinsey & Company: Managing Knowledge and Learning", *Harvard Business School, Case 9-396-357, 1996.*

BAUER, J., "Identification and Modeling of Contexts for Different Information Scenarios in Air Traffic", Diplomarbeit, 2003.

BERNDTSSON, M., HANSSON, J., LUNDELL, B. *et al*., *Thesis Projects.* 2 ed. London, Springer-Verlag, 2008.

BERZTISS, A., SYSLAB, "Knowledge and workflow systems". In: *Proceedings of the 11th International Workshop on Database and Expert Systems Applications, pp. 1102–1106, 2000.*

BIEGEL, G., CAHILL, V., "A Framework for Developing Mobile, Context Aware Applications". In: *2nd IEEE Conference on Pervasive Computing and Communications. Orlando, FL, March 14-17, 2004.*

BOUZY, B., AND CAZENAVE, T., "Using the Object Oriented Paradigm to Model Context in Computer Go". In: *Proceedings of Context'97, Riode Janeiro, Brazil, 1997.*

BPMN, 2009, "Business Process Modeling Notation (BPMN)", versão1.2, disponível em: http://www.omg.org/spec/BPMN/1.2/, acessado em agosto de 2009.

BRÉZILLON, P., POMEROL, J. C., "Modeling and using context for system development: Lessons from experiences**",** *Journal of Decision Systems, Vol. 10, pp. 265—288, 2001.*

BRÉZILLON, P., "Context in problem solving: A survey", *The Knowledge Engineering Review, vol. 14, No1, pp.1-34, 1999.*

BRÉZILLON, P., *Making context explicit in communicating objects.* Communicating Objects, C. Kintzig, G. Poulain, G. Privat, P-N. Favennec (Eds.), Hermes Science Editions, Lavoisier, 2002.

BROWN, J. S., DUGUID, P., "Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation", *Organization Science, 2, 1, pp. 40-57, 1991.*

CHEN, H., FININ, T., JOSHIRRY, A., "An Ontology for Context-Aware Pervasive Computing Enviroments". *Article, Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review, Vol. 18, No. 3, pp. 197-207, May 2004.*

CHEN, H. L., An Intelligent Broker Architecture for Pervasive Context-Aware Systems. Ph.D. Dissertation, Philosophy Department, University of Maryland, Maryland, USA, 2004.

CHOI, I., JUNG, J., SONG, M., "An integrated framework for process knowledge management", *International Journal of Innovation and Learning 1 (4), 399–408, 2004.*

CHRISTOPOULOU, E., GOUMOPOULOS, C., KAMEAS, A., "An ontology-based context management and reasoning process for ubicomp applications". In: *sOc-EUSAI '05: Proceedings of the 2005 joint conference on Smart objects and ambient intelligence, pages 265-270.*

CONKLIN, E. J., "Capturing Organizational Memory", In: *Proceedings of GroupWare '92, D. Coleman (Ed.), Morgan Kaufmann, San Mateo, CA, 133-137, 1992.*

DEY, A. K., SALBER, D., ABOWD, G.D., "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications", a*nchor article of a special issue on Context-Aware Computing, Human-Computer Interaction (HCI) Journal, Vol. 16(2-4), pp. 97-166, 2001.*

FOWLER, M., *Analysis patterns: reusable object models*. Menlo Park: Addison Wesley Longman, 1997.

FRIEDMANN-HILL, E.J., 1997, "Jess, The Expert System Shell for the Java Platform". Disponível em: http://herzberg.ca.sandia.gov/jess/docs/61/, acessado em agosto de 2009.

GELLERSEN, H.W., SCHMIDT, A., BEIGL, M., “Multi-sensor context-awareness in mobile devices and smart artifacts”. Mobile Networks and Applications, 7(5):341-351, 2002.

GRANT, R.M., “Prospering in Dynamically-Competitive Environments: Organizational Capability as Knowledge Integration”, *Organization Science, 7, 4, pp.375-387, July-August, 1996.*

GU, T., PUNG, H. K., ZHANG, D.Q., "A Service-Oriented Middleware for Building Context- Aware Services", *Elsevier Journal of Network and Computer Applications (JNCA), v. 28, n.1 (2005), pp. 1-18.*

GRUBER, T., 1993, “A translation approach to portable ontologies”, Knowledge *Acquisition*, 5(2):199-220.

GRUBER, T., 2008, “Ontology”. *Encyclopedia of Database Systems*, Liu. L., Özsu. M. (Eds.), Springer-Verlag.

JACOB, C., LINNER, D., RADUSCH, I *et al*., “Loosely coupled and context aware service provision incorporating the quality of rules”. In: *ICOMP 07: Proceedings of the 2007 International Conference on Internet Computing. CSREA Press, 2007.*

JADE, 2009, “Java Agent Development Framework”, disponível em http://sharon.cselt.it/projects/jade/, acessado em agosto de 2009.

JENA 2, 2009, “Jena 2”, disponível em http://jena.sourceforge.net, acessado em agosto de 2009.

KAGAL, L., FININ, T., JOSHI, A., “A policy language for a pervasive computing environment”. In: *IEEE 4th International Workshop on Policies for Distributed Systems and Networks, 2003.*

KLEMKE, R., Modelling Context in Information Brokering Processes. Ph.D. Thesis, RWTH, Aachen, Germany, 2002.

KOKINOV B., “Dynamics and automaticity of context: a cognitive modelling approach”. *Second International and Interdisciplinary Conference (CONTEXT'99), Trento, Italy, September 1999, in Lecture Notes in Artificial Intelligence 1688, Springer, Heidelberg, 1999.*

KORPIPÄÄ, P., MANTYJARVI, J., KELA, J., *et al*., “Managing Context Information in Mobile Devices”. *IEEE Pervasive Computing 02(3), 42-51, 2003.*

KUHN, O., ABECKER, A., "Corporate memories for knowledge management in industrial practice: prospects and challenges", *Journal of Universal Computer Science, Vol. 3 No.8, pp.929-54, 1997.*

LAI, J., FAN, Y., “Workflow and knowledge management: approaching an integration”. In: *Proceedings of the 1st International Conference on Engineering and Deployment of Cooperative Information Systems, Vol. 2480 of Lecture Note in Computer Science, Springer-Verlag, Berlin, pp.16–29, 2002.*

LIMA REIS, C.A., Uma Abordagem Flexível para Execução de Processos de Software Evolutivos. Tese de D.Sc., PPGC/UFRGS, Porto Alegre, RS, Brasil, 2003.

MAHÉ S., RIEU C., “A Pull Approach to Knowledge Management”, In: *Proceedings of the 2nd Int. Conf. On Practical Aspects of Knowledge Management (PAKM98), Basel, 1998.*

MALHORTA, Y., 1998, “Knowledge Management for the New World of Business”. Disponível em: http://www.brint.com/km/whatis.htm, acessado em: agosto de 2009.

MANTORO, T., JOHNSON, C. “Location history in a low-cost context awareness environment”. In: *ACSW Frontiers '03: Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003, pages 153-158, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc.*

MATTOSO, M., “Orientações para Orientandos – III Workshop de Teses e Dissertações em Banco de Dados”. Brasilia, DF, 2004.

MAURER, F., HOLZ, H., “Process oriented knowledge management for learning software organizations”. In: *Proceedings of the 12th Knowledge Acquisition Workshop (KAW’99), Banff, Canada, 1999.*

MCBURNEY, P., PARSONS, S., “Locutions for argumentation in agent interaction protocols”. In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, Volume 3, 2004.*

NONAKA, I., TAKEUSHI, H., “The Knowledge Creating Company”. *Harward Business Review, 1991.*

NONAKA, I., TOYAMA, R., HIRATA, T., *Managing Flow: A Process Theory of the Knowledge-Based Firm.* Palgrave Macmillan, 2008.

NUNES, V. T., SANTORO, F.M., BORGES, M., “Context in Knowledge Intensive Collaborative Work”. In: *10th International Conference on CSCW in Design, 2006, Nanjing. Proceedings on 10th International Conference on CSCW in Design, 2006. v. 2. p. 793-798.*

NUNES, V.T., Um Modelo de Suporte à Gestão de Conhecimento Baseado em Contexto. Dissertação de M.Sc., NCE-IM/UFRJ, Rio de Janeiro, RJ, Brasil, 2007a.

NUNES, V. T., SANTORO, F. M., BORGES, M., “Capturing Context about Group Design Process”. In: *The 11th International Conference on CSCW in Design, 2007, Melbourne. Proceedings of The 11th International Conference on CSCW in Design, 2007b. v. 1. p. 18-23.*

O'LEARY, D. E., "Enterprise knowledge management". *IEEE Computer Systems, vol.13, no.2, pp.54-61, 1998.*

ONTOVIZ, 2009, OntoViz, versão 1.0, disponível em: http://protegewiki.stanford.edu/index.php/OntoViz_1.0, acessado em agosto de 2009.

OPENUP, 2009, "OpenUp", versão:1.5.0.1, disponível em: http://epf.eclipse.org/wikis/openup/, acessado em agosto de 2009.

ÖTZTÜRK, P., AAMODT, A. "Towards a model of context for case-based diagnostic problem solving". In: *Context-97; Proceedings of the interdisciplinary conference on modeling and using context, pp. 198-208, Rio de Janeiro, RJ, Brazil, 1997.*

PARRINI, E., Gestão do Conhecimento no Suporte à Decisão em Ambiente OLAP. Dissertação de M.Sc., NCE-IM/UFRJ, Rio de Janeiro, RJ, Brasil, 2002.

PERRET, R., BORGES, M.R.S., SANTORO, F.M., "Applying group storytelling in knowledge management". In: *CRIWG 2004. San Carlos, Costa Rica, 2004.*

POOLE, D., "Compiling a default reasoning system into prolog". *New Generation Computing 9(1):3–38, 1991.*

PRIÉ Y., MILLE A., PINON J.M., "A Context-Based Audiovisual Representation Model for Audiovisual Information Systems". *Second International and Interdisciplinary Conference (CONTEXT'99), Trento, Italy, September 1999, in Lecture Notes in Artificial Intelligence 1688, Springer, Heidelberg, 1999.*

PROTÉGÉ OWL, 2009, "Protégé OWL Editor, versão 3.4, disponível em: http://protege.stanford.edu/overview/protege-owl.html, acessado em agosto de 2009.

RDF, 2004, "Resource Description Framework (RDF)", disponível em: http://www.w3.org/RDF/, acessado em agosto de 2009.

RDF VALIDATOR, 2009, "W3C RDF Validation Service", disponível em: http://www.w3.org/RDF/Validator/, acessado em agosto de 2009.

RDQL, 2004, "RDQL - A Query Language for RDF", disponível em: http://www.w3.org/Submission/RDQL/, acessado em agosto de 2009.

REYNOLDS, D., 2009, “Jena 2: Inference Support”. Disponível em: http://jena.sourceforge.net/inference/index.html, acessado em agosto de 2009.

ROCHA, R., ENDLER, M., “Evolutionary and eficient context management in heterogeneous environments”. In: *MPAC '05: Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing, pages 1-7, New York, NY, USA, 2005. ACM Press.*

RODRIGUEZ M.A., EGENHOFER M.J., “Putting similarity assessments into context: Matching functions with the user's intended operations”. *Second International and Interdisciplinary Conference (CONTEXT'99), Trento, Italy, September 1999, in Lecture Notes in Artificial Intelligence 1688, Springer, Heidelberg, 1999.*

ROMAN, M., HESS, C., CERQUEIRA, R. *et al.*, “Gaia: A middleware infrastructure to enable active spaces”. IEEE Pervasive Computing , pp. 74-83, 2002.

SCHMOHL, R. e BAUMGARTEN, U., “Context-aware Computing: a Survey Preparing a Generalized Approach” In: *Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol I, IMECS 2008, 19-21 March, 2008, Hong Kong, 2008.*

SEABORNE, A., “RDQL - A Query Language for RDF”. W3C Member Submission, Hewlett Packard, 2004. Disponível em: http://www.w3.org/Submission/RDQL/, acessado em agosto de 2009.

SENSIPER, S., “AMS Knowledge Centers,” *Harvard Business School Case 9-697-068, 1997.*

SINGH, A., CONWAY, M., "Survey of Context aware Frameworks – Analysis and Criticism", version: 1, Information Technology Services, University of North Carolina, Chapel Hill, North Carolina, USA, 2006.

STEELS, L., “Corporate knowledge management”. *Proceedings of ISMICK'93 Conference, p. 9-30, Compiégne, France, 1993.*

STRANG, T., LINNHOFF-POPIEN, L., “A Context Modeling Survey”. *First International Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham, 6th International Conference on Ubiquitous Computing. UK. pp. 33-40, 2004.*

SWRL, 2009, “A Semantic Web Rule Language Combining OWL and RuleML”, disponível em: http://www.w3.org/Submission/SWRL, acessado em agosto de 2009.

TERRA, J. C. C., GORDON, C., *Portais corporativos: a revolução na gestão de conhecimento*. São Paulo: Negócio, 2002.

UML, 2009, “Unified modeling language”, disponível em http://www.uml.org, acessado em agosto de 2009.

VAN DER AALST, W. M. P. et al, “Business process management: A survey”. *Proceedings of the 1st International Conference on Business Process Management, Berlin, Germany, pp. 1–12, 2003.*

VIEIRA, V., BRÉZILLON, P., SALGADO, A.C. *et al.*, "A Context-Oriented Model for Domain-Independent Context Management", *Revue d'Intelligence Artificiel, v. 22, n.5, 2008, pp. 609-627.*

VIEIRA, V., CEManTIKA: A Domain Independent Framework for Designing Context-Sensitive Systems. Tese de D.Sc., Departamento de Ciências da Computação, UFPE, Recife, Pernambuco, Brasil, 2008.

WANG, X., ZHANG, D.Q., GU, T. et al., “Ontology based context modeling and reasoning using OWL”. *Workshop on Context Modeling and Reasoning at IEEE International Conference on Pervasive Computing and Communication, Orlando, FL, 2004.*

WANT, R., HOPPER, A., FALCAO, V., *et al.*, "The Active Badge Location System", ACM Transcation on Information Systems (TOIS), 10 (1), pp. 91-102, 1992.

WOOLDRIDGE, M, JENNINGS, N.R. “Agent Theories, Architectures and Languages: a Survey” in Wooldridge and Jennings Eds., *Intelligent Agents*, Berlim: Springer-Verlag, 1-22, 1995

YAU, S.S., KARIM, F., WANG, Y., *et al*., “Reconfigurable context-sensitive middleware for pervasive computing”. *IEEE Pervasive Computing, 1(3):33-40, 2002.*

Apêndice A – Ontologia estendida para o domínio de planejamento de projetos de software

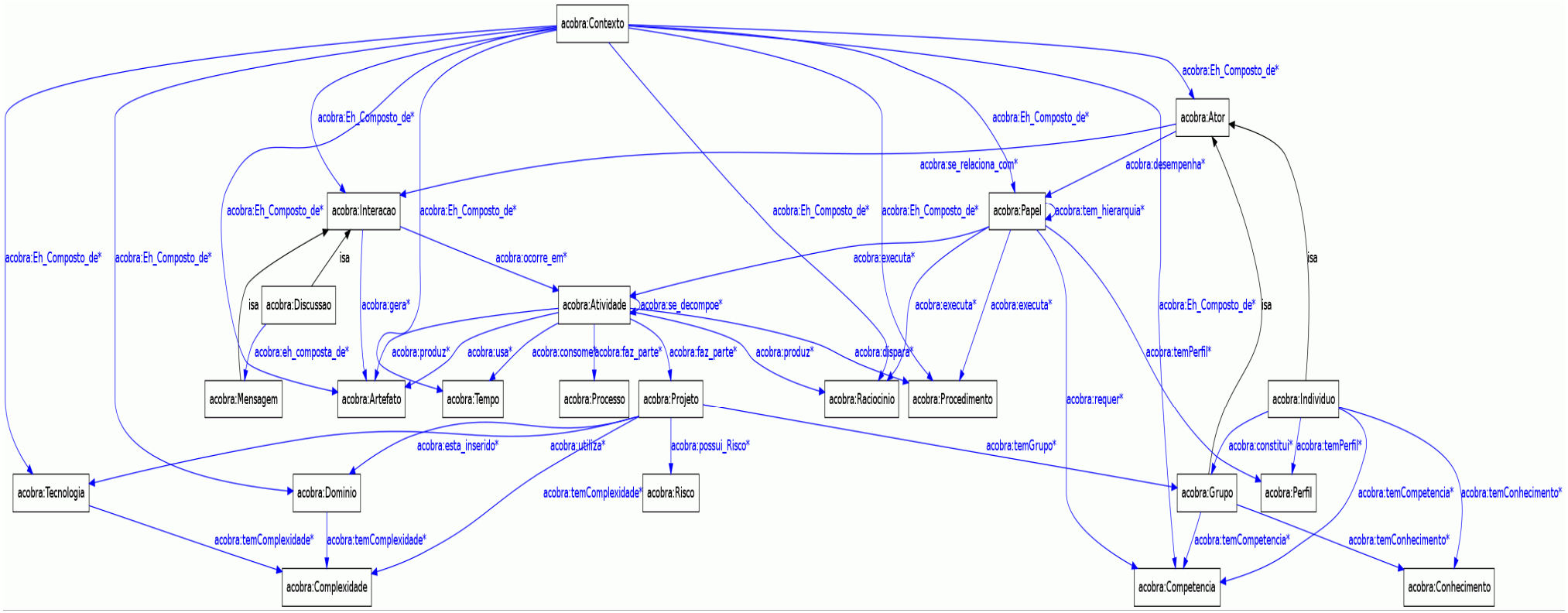

Figura A.1 – Ontologia estendida para o domínio de planejamento de projetos de software.