Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

MADIK

Uma Abordagem Multiagente para o Exame Pericial de Sistemas Computacionais

Bruno Werneck Pinto Hoelz

Brasília
2009

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

MADIK

Uma Abordagem Multiagente para o Exame Pericial de Sistemas Computacionais

Bruno Werneck Pinto Hoelz

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof.ª Dr.ª Célia Ghedini Ralha

Brasília
2009

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenadora: Prof. Dr. Li Weigang

Banca examinadora composta por:

Prof.ª Dr.ª Célia Ghedini Ralha (Orientadora) — CIC/UnB
Prof. Dr. Wagner Meira Jr. — UFMG
Prof. Dr. Pedro de Azevedo Berger — CIC/UnB

CIP — Catalogação Internacional na Publicação

Hoelz, Bruno Werneck Pinto.

MADIK: Uma Abordagem Multiagente para o Exame Pericial de Sistemas Computacionais / Bruno Werneck Pinto Hoelz. Brasília : UnB, 2009.
137 p. : il. ; 29,5 cm.

Tese (Mestrado) — Universidade de Brasília, Brasília, 2009.

1. computação forense, 2. inteligência artificial, 3. sistemas multiagente

CDU 004.8

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

MADIK

Uma Abordagem Multiagente para o Exame Pericial de Sistemas Computacionais

Bruno Werneck Pinto Hoelz

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof.ª Dr.ª Célia Ghedini Ralha (Orientadora)
CIC/UnB

Prof. Dr. Wagner Meira Jr.
UFMG

Prof. Dr. Pedro de Azevedo Berger
CIC/UnB

Prof. Dr. Li Weigang
Coordenadora do Mestrado em Informática

Brasília, 5 de março de 2009

Dedicatória

Para minha querida avó Inah Werneck, que partiu antes de poder ver seu neto alcançar mais esta conquista.

Agradecimentos

À minha orientadora Profª Célia por toda a atenção, paciência e dedicação; por ter acreditado na proposta deste trabalho e por todo o incentivo e auxílio na superação dos inúmeros desafios encontrados no caminho.

Aos bolsistas de iniciação científica Rajiv e Hugo, que auxiliaram na implementação do protótipo e ao graduando Bruno Costa, pela contribuição no estudo de técnicas de tolerância a falhas na plataforma JADE, relacionado a este trabalho.

Finalmente, um imenso agradecimento aos familiares e amigos que compreenderam e suportaram tantas ausências decorrentes de todas as obrigações e compromissos encontrados ao longo desta empreitada.

Resumo

O exame pericial de sistemas computacionais e evidências digitais é uma tarefa complexa que exige habilidades altamente especializadas para identificar, coletar e analisar vestígios importantes para a investigação de um crime. Muitos são os desafios para a realização de um exame satisfatório, uma vez que os recursos humanos e materiais, bem como o tempo disponível, são na maioria das vezes muito limitados. Com isso, nem sempre é possível realizar um exame tão abrangente e completo quanto o desejado. Este trabalho propõe o desenvolvimento e a utilização de um Sistema Multiagente na realização de exames periciais de sistemas computacionais. O sistema proposto, denominado *Multi-Agent Digital Investigation toolKit* (MADIK), apresenta uma arquitetura baseada na estrutura hierárquica das organizações, dividida em quatro níveis: estratégico, tático, operacional e especializado. Os agentes localizados em cada um desses níveis trabalham de forma distribuída e cooperativa para examinar as evidências digitais relacionadas a uma investigação. Suas descobertas e recomendações são registradas em uma estrutura compartilhada de *blackboard*, permitindo ao especialista humano avaliar e revisar os resultados obtidos. A estrutura do *blackboard* permite também que os agentes realizem a correlação de suas descobertas. Este trabalho apresenta a proposta do Sistema Multiagente (SMA), a implementação de um protótipo e os resultados obtidos ao longo de vários experimentos. Os resultados obtidos foram muito positivos, com bons níveis de abrangência da análise do conteúdo das evidências e de correlação de vestígios, e com a redução do tempo total necessário para realização dos exames periciais quando comparados a um examinador humano utilizando as ferramentas forenses atuais em um cenário de recursos materiais e tempo limitados.

Palavras-chave: computação forense, inteligência artificial, sistemas multiagente

Abstract

The forensic examination of computational systems is a complex task that demands highly specialized skills to identify, collect, and analyze evidence of interest to the investigation of a crime. Many are the challenges on the way to a satisfactory examination, since the resources, both human and material, as well as the time available, are often limited. So it's not always possible to perform an examination as complete as desired. This work proposes the use of a Multi-Agent System in the performing of forensic examination of computational systems. The proposed Multi-Agent System (MAS), named *Multi-Agent Digital Investigation toolKit* (MADIK) presents an architecture based on the hierarchical structure of organizations, divided into four layers: strategic, tactical, operational and specialized. The agents in each one of these layers work in a distributed and cooperative manner to examine the digital evidence related to the investigation. Their findings and recommendations are placed in a *blackboard* shared structure, that allows the human specialist to evaluate and review the obtained results. The blackboard structure also allows other agents to correlate their findings with others. This work presents the proposed system, a working prototype and the results obtained after a series of experiments. The obtained results where very positive, with good coverage levels in the digital evidence content analysis and correlation of findings, as well as reduction in the time required to perform the forensic examination when compared to those obtained by a human examiner using the currently available forensic tools in a scenario of limited time and material resources.

Keywords: computer forensics, artificial intelligence, multiagent systems

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AID *Agent IDentifier*. 54

AMS *Agent Management System*. 54–57, 90

CBR *Case-Based Reasoning*, tradução para o inglês de raciocínio baseado em casos. 81

CT *Container Table*. 57

DF *Directory Facilitator*. 54–57, 90

EXIF *Exchangeable Image File Format*. 32, 124

FBI *Federal Bureau of Investigation*. 5, 6

FC fontes de conhecimento. 49–51

FIPA *Foundation for Intelligent Physical Agents*. 39, 46, 49, 54–58, 81, 87

FIPA-ACL *FIPA Agent Communication Language*. 49, 55

FTK *Forensic ToolKit*. 96, 101, 109, 110, 115

GADT *Global Agent Descriptor Table*. 57, 58

HTN *Hierarchical Task Network* ou rede de tarefas hierárquicas. 63, 64

IA Inteligência Artificial. 28, 37, 43, 49, 117

IAD Inteligência Artificial Distribuída. 1, 37, 51

JADE *Java Agent DEvelopment Framework*. 39, 46, 56–58, 86, 87, 89, 90, 96, 98

JVM *Java Virtual Machine*. 56, 73, 86

KIF *Knowledge Interchange Format*. 49

KQML *Knowledge Query and Manipulation Language*. 49

LADT *Local Agent Descriptor Table*. 58

MADIK *Multi-Agent Digital Investigation toolKit*. vi, vii, 4, 69, 87, 96, 98, 111, 112, 114–117

MAS *Multi-Agent System*, tradução para o inglês de Sistema Multiagente. vii, 52

MaSE *Multi-agent Systems Engineering*. 46, 52

MD5 *Message-Digest Algorithm 5*. 31, 93

MTS *Message Transport System*. 55

OWL *Web Ontology Language*. 53

PASSI *Process for Agent Societies Specification and Implementation*. 46–48

PDDL *Planning Domain Definition Language*. 61, 62

PF Polícia Federal. 5, 6, 92

RBC Raciocínio Baseado em Casos. 4, 65–68, 76, 91, 111, 114, 116, 117

SL *FIPA Semantic Language*. 49

SMA Sistema Multiagente. vi, 1, 3, 4, 23, 31, 36–43, 45, 46, 49, 51–56, 58, 64, 68–70, 75, 79–81, 86, 87, 89, 92, 95, 96, 98, 101, 103, 106, 109, 111, 114, 115, 117

SQL *Structured Query Language*. 92

UML *Unified Modelling Language*. 46, 47, 56

W3C *World Wide Web Consortium*. 53

Sumário

Capítulo 1

Introdução

O exame pericial de sistemas computacionais consiste na preservação, análise e apresentação dos vestígios relacionados à prática de algum crime ou a incidentes de segurança envolvendo computadores. É uma atividade complexa, que demanda grandes quantidades de recursos computacionais e humanos, além do tempo necessário para sua realização. Com relação a esses recursos, muitas vezes são escassos ou insatisfatórios, o que reflete diretamente no resultado obtido, que por vezes pode ser incapaz de identificar importantes vestígios para a elucidação do ocorrido.

Embora o exame pericial de sistemas computacionais possa ser realizado em diversos contextos e situações, neste trabalho ele será tratado exclusivamente do ponto de vista policial, na investigação de crimes de qualquer natureza que envolvam computadores.

Neste trabalho é apresentado um Sistema Multiagente (SMA), uma abordagem de Inteligência Artificial Distribuída (IAD), para aplicação na realização de exames periciais em sistemas computacionais e evidências digitais. No SMA proposto, agentes inteligentes auxiliam, de maneira autônoma e distribuída, os especialistas humanos na realização do seu trabalho, com o objetivo de aumentar a agilidade e a eficácia desses exames, com ganhos também no melhor aproveitamento dos recursos humanos e computacionais.

A Seção 1.1 apresenta o problema enfrentado atualmente na realização de exames periciais em sistemas computacionais, que serve como motivação para este trabalho. Os principais objetivos são descritos na Seção 1.2, enquanto a Seção 1.3 descreve a metodologia empregada neste trabalho.

1.1 Motivação

A presença crescente de computadores e mídias de armazenamento computacional na vida cotidiana refletiu-se também nas cenas de crime, onde comumente são encontrados computadores que apresentam relação com o fato sob investigação. Esses vestígios devem ser examinados para encontrar evidências importantes para a elucidação do ocorrido. Tal situação não ocorre somente no caso de crimes realizados em redes de computadores como a Internet, mas também em outros crimes que hoje são realizados com o auxílio de recursos computacionais.

Somados a essa demanda crescente, observa-se também o aumento constante da capacidade das mídias de armazenamento e consequentemente do volume de dados a ser examinado. Nos últimos anos tal tendência vem se acentuando, conforme pode ser visto

na Figura 1.1, que apresenta o tamanho médio dos discos rígidos examinados pela Polícia Federal desde 2003[1]. Pode-se notar que o tamanho médio de um disco rígido em 2003 não passava de 20 GB e cinco anos depois já ultrapassava a marca de 80 GB. Ressalta-se que há uma certa defasagem entre a época de realização do exame e da utilização do disco rígido pelo suspeito. Ou seja, os discos rígidos examinados hoje são aqueles comercializados em maior quantidade um ou dois anos atrás.

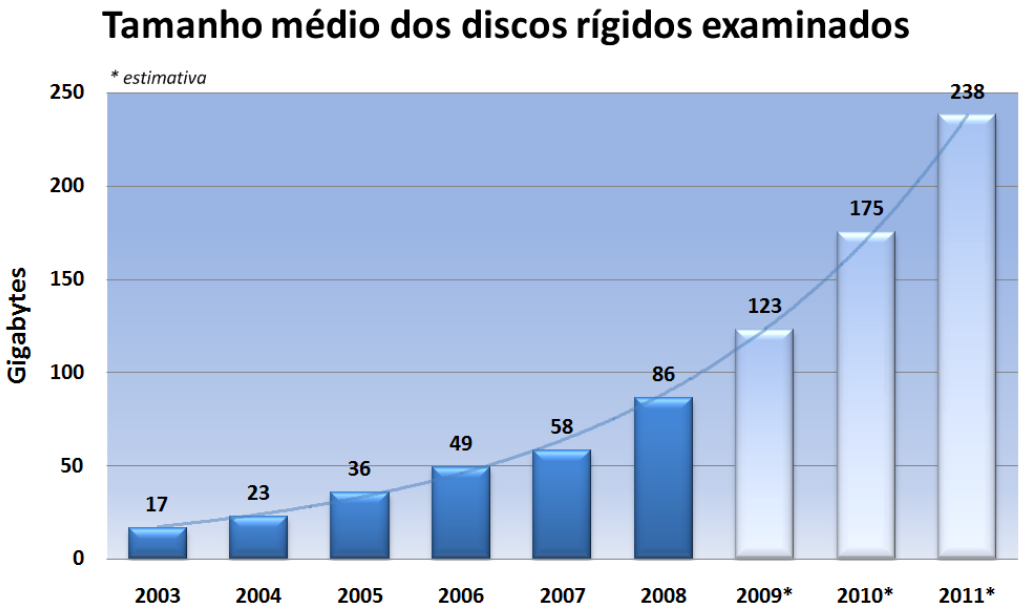

Figura 1.1: Tamanho médio dos discos rígidos examinados pela Polícia Federal

Nesse cenário de demanda crescente, as ferramentas atualmente disponíveis não têm fornecido novos recursos que auxiliem os especialistas em sua tarefa com formas mais eficientes de examinar e correlacionar um grande volume de dados. Com isso, um grande esforço ainda é despendido em exames isolados de vestígios, em que cada especialista examina um vestígio ou conjunto reduzido de vestígios de um caso em sua estação pericial, sem realizar qualquer correlação com outros vestígios do mesmo caso que estão sendo examinados por seus colegas. O uso de recursos computacionais pelas ferramentas atuais também está aquém do desejado, uma vez que as ferramentas disponíveis não fazem uso da distribuição do processamento e análise desses vestígios para aproveitar a capacidade ociosa dos recursos computacionais disponíveis nos laboratórios.

Não há nas ferramentas atuais recursos adequados para a fácil personalização dos exames a serem realizados em um vestígio, nem para a reutilização do conhecimento obtido em exames anteriores. Assim, algumas ferramentas realizam procedimentos desnecessários em alguns casos, consumindo um tempo valioso que não contribuirá para a conclusão célere do exame. Outras não permitem a personalização dos procedimentos levando-se em consideração a natureza do caso em questão. Assim, todo vestígio, seja ele de um crime de fraude financeira ou de um caso de invasão de servidor *web* é tratado da mesma forma, executando-se o mesmo procedimento, com as mesmas configurações, mesmo que

[1]Dados do Serviço de Perícias em Informática do Instituto Nacional de Criminalística.

parte do procedimento não seja necessário ou nem mesmo adequado ao caso. Com relação aos recursos humanos, nem sempre os especialistas compartilham do mesmo nível de conhecimento ou experiência em um determinado caso. A reutilização de conhecimento para orientar examinadores inexperientes em determinados casos, portanto, é um recurso extremamente necessário.

Diante dessas dificuldades e limitações, os especialistas cada vez mais realizam exames cujos resultados são pouco correlacionados e o conhecimento adquirido naquele caso não é disseminado. Devido às limitações na automação inteligente dos exames, examinadores com grande especialidade em áreas de maior complexidade gastam um valioso tempo realizando exames rotineiros, ao invés de utilizar seu conhecimento de mais alto nível. A limitação na capacidade dessas ferramentas de realizar o processamento distribuído e a correlação de um grande volume de dados, afeta diretamente a qualidade dos exames, o que pode se refletir na perda de importantes evidências para a solução de crimes e outros incidentes. Em alguns casos, essa perda pode resultar em impunidade ou injustiça, o que sempre representa um grande dano para a sociedade.

Logo, há uma necessidade premente de novas soluções para o exame pericial de sistemas computacionais, que possam torná-lo mais ágil e eficaz, impedindo que sua qualidade seja prejudicada pelo aumento da demanda ou que os resultados das investigações sofram com a lentidão na sua realização. Essas soluções devem levar em consideração a melhor utilização dos recursos humanos e computacionais e a obtenção e disseminação do conhecimento obtido em casos passados.

1.2 Objetivos

O objetivo maior deste trabalho é projetar um SMA que permita ganhos significativos de desempenho e eficácia nos exames periciais, com melhor aproveitamento de recursos computacionais e humanos e que forneça mecanismos de reutilização de conhecimento. Para alcançar esse objetivo principal, foram definidos três objetivos específicos:

1. fornecer uma maneira rápida de apresentar ao examinador arquivos contidos em uma evidência que tenham maior probabilidade de conter informações valiosas para a investigação. Com isso, procura-se reduzir o tempo total de realização dos exames;
2. permitir a análise e correlação de um grande número de arquivos provenientes de diferentes evidências de um mesmo caso, além da capacidade humana e das ferramentas disponíveis atualmente;
3. fornecer meios de obtenção e reutilização de conhecimento de casos anteriores, o que permitirá a melhoria dos exames e o aprendizado dos especialistas.

1.3 Metodologia

Para a realização deste trabalho, foram desenvolvidas diversas atividades. A primeira delas foi a revisão da literatura de Informática Forense e de SMA, com ênfase em questões de organização e planejamento.

Com base na revisão da literatura de Informática Forense, foram identificados os principais desafios e estudos correlatos que apresentavam soluções interessantes. Com a revisão

da literatura de SMA, definiu-se a metodologia para concepção do sistema, sua organização e formas de comunicação e planejamento. Adicionalmente, observou-se a necessidade de uma pesquisa mais apurada sobre o uso de ontologias e mecanismos de planejamento.

Após a revisão da literatura, foi definida a proposta de arquitetura organizada hierarquicamente, descrita em detalhe no Capítulo 4. Foi implementado o protótipo para a realização de testes de comunicação e distribuição utilizando os níveis operacional e especializado da hierarquia. Os experimentos e resultados foram apresentados em 2007 na *II International Conference on Forensic Computer Science* (Hoelz, 2007).

Dando continuidade ao trabalho, novos agentes especializados foram concebidos e o nível tático da arquitetura foi acrescentado. A primeira implementação do sistema, chamada *Multi-Agent Digital Investigation toolKit* (MADIK), foi apresentada em 2008 na *16th International Conference on Cooperative Information Systems* (Hoelz et al., 2008b), no México. Em seguida, os resultados do primeiro estudo de caso foram apresentados na *2008 IEEE/WIC/ACM International Conference on Intelligent Agent Technology* (Hoelz et al., 2008a), na Austrália. Os testes realizados com maior grau de distribuição e um maior número de agentes indicaram a necessidade de um novo modelo de planejamento das atividades e coordenação entre os agentes.

Assim, estudou-se em seguida o problema do planejamento automatizado em SMA. Foi concebido um processo de planejamento e alocação de recursos considerando a organização hierárquica da proposta e as especificidades de cada nível da hierarquia. Os mecanismos de interação também foram melhor definidos, assim como foi concebida uma ontologia mínima para ser utilizada na coordenação dos agentes.

Um segundo estudo de caso foi realizado e seus resultados descritos no artigo *Artificial Intelligence Applied to Computer Forensics*, a ser apresentado no *24th Annual ACM Symposium on Applied Computing* (Hoelz et al., 2009), nos EUA. Desses resultados observou-se a necessidade de um mecanismo adequado para a reutilização de conhecimento. Com isso, sugeriu-se que mecanismos relacionados ao uso de Raciocínio Baseado em Casos (RBC) fossem implementados em trabalhos futuros.

Os demais capítulos deste trabalho estão organizadas da seguinte forma: os Capítulos 2 e 3 apresentam a fundamentação teórica relacionada a este trabalho; o Capítulo 4 contém a proposta detalhada; no Capítulo 5 são apresentados os diversos experimentos realizados e a análise dos resultados obtidos; por fim, as conclusões e sugestões de trabalhos futuros são apresentadas no Capítulo 6.

Capítulo 2

Informática Forense

Este capítulo apresenta os principais conceitos, definições e pesquisas em Informática Forense. A Seção 2.1 apresenta uma visão geral da área. A Seção 2.2 discute alguns dos maiores desafios atuais enfrentados durante a realização de exames periciais de sistemas computacionais. Por fim, a Seção 2.3 apresenta pesquisas diversas em Informática Forense que compartilham alguns dos objetivos deste trabalho.

2.1 Visão Geral

Esta seção apresenta uma visão geral da Informática Forense. A Seção 2.1.1 apresenta um breve histórico do surgimento dessa área. Em seguida, na Seção 2.1.2, são apresentados conceitos e definições. A Seção 2.1.3 descreve os principais procedimentos e trabalhos relacionados ao estabelecimento de *frameworks* para a execução de exames periciais.

2.1.1 Histórico

Segundo Huebner et al. (2003), o primeiro caso criminal relacionado ao uso de computadores conhecido foi registrado nos EUA em 1966 e resultou em uma pena de cinco anos de prisão[1]. Nas décadas de 1970 e 1980, os computadores tornaram-se mais comuns e baratos, permitindo a utilização pessoal e comercial em maior escala. Com isso, a polícia identificou o surgimento de uma nova classe de crimes: os crimes relacionados a computadores. Desde 1984, os laboratórios do *Federal Bureau of Investigation* (FBI) começaram a desenvolver programas para examinar evidências em computadores. As primeiras ferramentas especializadas foram desenvolvidas em meados da década de 1980 e o primeiro treinamento de peritos em Informática surgiu em 1989 na *University of North Texas*.

Na década de 1990, a maioria das forças policiais dos países mais avançados tecnologicamente já tinham contato com crimes relacionados a computadores. No Brasil, a primeira unidade especializada surgiu em 1996, na Polícia Federal (PF). A PF contava então com apenas dez peritos com formação na área de computação, que atendiam todo o país. O primeiro evento internacional, o *International Law Enforcement Conference on Computer Evidence*, foi promovido pelo FBI em 1993 no EUA.

[1] Tratava-se de um caso de furto de programa de computador.

De acordo com Huebner et al. (2003), esse primeiro período na história da perícia em Informática caracterizou-se pelo tratamento de dispositivos com capacidades de armazenamento relativamente pequenas e poucas quantidades de informação. Isso permitia que cópias completas dos discos rígidos originais fossem feitas para outro disco, sendo o exame realizado sobre a cópia, preservando assim a evidência original.

Com o surgimento da Internet comercial no Brasil no início de 1995[2], surge uma nova demanda para as forças policiais brasileiras, relacionada à prática de crimes na Internet, hoje conhecidos comumente como crimes cibernéticos. Com o crescimento explosivo da Internet, também cresceram as ocorrências de incidentes relacionados como a invasão de servidores e a prática de *defacement* de páginas *web*. A caracterização desses incidentes como crimes ainda era difícil, não havendo entendimento jurídico consolidado sobre como processar esses indivíduos por condutas que não se assemelhavam àquelas elencadas até então no Código Penal brasileiro.

Além do crescimento da Internet, é importante lembrar também da evolução da telefonia móvel e dos meios de armazenamento de dados. Ambos tornaram-se extremamente acessíveis e hoje fazem parte da vida cotidiana. Da mesma forma, pode-se afirmar que fazem parte do cotidiano de criminosos, que muitas vezes fazem uso dessas tecnologias, mesmo que os crimes que cometam não tenham relação direta com a Informática. Hoje, discos rígidos com capacidade de dois terabytes já podem ser comprados. Esse grande volume de dados e diversidade de mídias é um dos grandes desafios enfrentados pela Informática Forense, que precisa buscar novas soluções para cenários onde não é mais possível realizar uma cópia integral de todos os dados originais, como em ambientes de rede complexos.

Para atender tal demanda, a Polícia Federal conta hoje, pouco mais de uma década após a contratação dos primeiros peritos, com mais de 160 desses especialistas, todos com formação de nível superior na área de computação e localizados em todos as unidades da Federação. Durante esse tempo outras forças policiais brasileiras criaram suas unidades de perícia em Informática e algumas já possuem núcleos especializados em investigações de crimes cibernéticos. Em 2004, foi realizado o primeiro evento de grande porte no Brasil na área de perícias em Informática. A *International Conference on Cyber Crimes Investigation* (ICCyber) foi promovida pela PF e contou com a participação de 20 países.

Segundo dados do Relatório Estatístico das Atividades do Sistema Nacional de Criminalística de 2008 da Polícia Federal, em 2000, a produção de laudos de Informática correspondia à apenas 1% do total de laudos emitidos. Em 2008, o número saltou para 19,23%, com 9060 laudos produzidos em todo o país. Além disso, os exames de Informática representam a maioria das solicitações de exames pendentes, com 24,53% do total. Ao final de 2008, os peritos em Informática correspondiam a 17,47% dos peritos em atividade, sendo a segunda maior área de perícia em número de especialistas na PF, com 163 peritos.

Portanto, tal como se observa o aumento do uso da Internet e de outras tecnologias na vida cotidiana, pode-se notar também uma tendência que o acompanha: o aumento dos crimes relacionados a computadores e a exigência de maior ação policial e consequente demanda por mais exames periciais.

Uma pesquisa anual do FBI nos EUA mostrou que entre 1999 e 2006, de 30% a 45% dos pesquisados não comunicou incidentes de invasão em seus computadores, sendo o medo

[2]Considerando a data de criação do Comitê Gestor de Internet em maio de 1995.

da publicidade negativa o principal motivo para tal (Huebner et al., 2003). Pesquisa semelhante em 2006 na Austrália mostrou um percentual de 69% que não comunicou ataques sofridos de fontes externas. Isso demonstra que a demanda real ainda é muito maior do que a vivenciada pelos órgãos policiais.

No Brasil, cabe ainda considerar a carência de uma legislação específica para punir diversas condutas no meio cibernético, que em muitos países já são considerados crimes, como a disseminação de programas maliciosos como vírus de computador. Diversos projetos de lei foram propostos ao longo dos anos, mas até a conclusão deste trabalho nenhum havia sido aprovado em caráter conclusivo. Uma discussão mais ampla sobre questões de legislação foge ao escopo deste trabalho, mas é importante lembrar que com a existência de leis específicas, amplia-se a possibilidade de ação policial e novamente a demanda por exames periciais adequados.

Durante todos esses anos, as práticas e ferramentas periciais foram discutidas e aperfeiçoadas. Em agosto de 2001, foi realizado na cidade de Utica, no estado de New York, nos EUA, a primeira edição do *Digital Forensic Research Workshop*, que segundo Palmer (2001), reuniu mais de 50 pesquisadores de universidades, especialistas em exames periciais e análise de computadores. O objetivo do *workshop* era reunir acadêmicos e profissionais interessados em melhor definir esse campo, assim como identificar as dificuldades e os desafios que encontrariam pela frente. Mas o maior dos seus objetivos era estabelecer uma comunidade de pesquisa, que aplicasse métodos científicos na busca de soluções, cujos resultados beneficiariam todos os envolvidos com a Informática Forense.

A Informática Forense ainda é uma área de pesquisa jovem, que carece de definições e padrões claros, tem necessidades bastante específicas e que constantemente encontra novos desafios. As seções seguintes deste capítulo apresentam as definições e conceitos mais utilizados atualmente, bem como os principais desafios encontrados ao longo dos anos e alguns dos trabalhos desenvolvidos sobre os pontos de vista policial, militar, acadêmico e comercial.

2.1.2 Definição

Antes de apresentar uma definição de Informática Forense é necessário esclarecer alguns pontos com relação à tradução do termo em inglês correspondente e de outras denominações utilizadas.

O termo em inglês associado à perícia em computadores é *Computer Forensics*. Em inglês, o termo *forensics* é definido como:

> a aplicação de conhecimento científico em problemas legais e especialmente a análise científica de evidências físicas como as encontradas em uma cena de crime[3].

Uma tradução muito utilizado comercialmente no Brasil é *Forense Computacional*, embora seja uma substantivação forçada do adjetivo forense que não explicita a mesma semântica da definição do termo *forensics*. O termo mais próximo de *forensics* com base na definição apresentada seria de fato *perícia* e, por conseguinte, *Computer Forensics* poderia ser traduzido adequadamente como *perícia em computadores*.

[3]Definição do dicionário Merriam-Webster disponível em `http://www.merriam-webster.com`

O profissional responsável pela realização de exames periciais é chamado perito, também chamado neste trabalho de especialista, que é a tradução do termo *expert* utilizado em inglês, ou examinador.

Neste trabalho, utilizar-se-á perícia em Informática, um termo um pouco mais amplo que perícia em computadores, como análogo do termo *Computer Forensics* em inglês e *Informática Forense* como a denominação da área de pesquisa relacionada, cujo termo mais utilizado em inglês é *Digital Forensic Science*. Outros termos eventualmente utilizados em inglês são *Digital Investigation*, *Forensic Computer Science*, *Forensic Discovery* e *Eletronic Discovery*. Esses termos apresentam algumas diferenças sutis, relacionadas aos diferentes pontos de vista policial, militar, comercial ou acadêmico. A Figura 2.1 apresenta as áreas de aplicação das pesquisas na área em Informática Forense.

Figura 2.1: Áreas de interesse e pesquisa na Informática Forense, adaptada de Palmer (2001)

A definição utilizada neste trabalho para a Informática Forense é emprestada da definição de *Digital Forensic Science* discutida e apresentada em Palmer (2001). Assim, a Informática Forense é definida como:

> o uso de métodos cientificamente estabelecidos e comprovados para a preservação, coleta, validação, identificação, análise, interpretação, documentação e apresentação da evidência derivada de fontes digitais para o propósito de facilitar ou promover a reconstrução de eventos que causem a perturbação de operações planejadas.

Segundo Palmer (2001), a maioria das pesquisas na área de Informática Forense visa o uso policial. Desse ponto de vista, a "perturbação de operações planejadas" representa uma potencial conduta criminosa. Assim sendo, a definição acima não enfatiza o caráter legal que diferencia a Informática Forense. O *Australian Institute of Criminology* apresenta uma definição mais adequada a esse ponto de vista, no qual a perícia em Informática é definida como:

> o processo de identificar, preservar, analisar e apresentar evidências digitais de uma maneira legalmente aceitável.

Tabela 2.1: Objetivos da Informática Forense em áreas diversas, adaptada de Palmer (2001)

Área	**Objetivo primário**	**Objetivo secundário**	**Quando atua**
Policial	Persecução penal	–	Depois do fato
Militar	Continuidade	Persecução penal	Tempo real
Comercial	Disponibilidade	Persecução penal	Tempo real

Para melhor entender os crimes envolvendo computadores, Huebner et al. (2003) sugere uma classificação em três tipos:

1. Crimes centrados no computador: atividade criminosa cujos alvos são sistemas computacionais, redes, meios de armazenamento digital e outros dispositivos. Podem ser vistos como uma nova classe de crime. Exemplo: invadir um sítio na Internet e modificar seu conteúdo.

2. Crimes auxiliados por computador: uso de computadores para a prática de crimes que não necessariamente necessitam de um computador. Podem ser vistos como uma nova maneira de cometer crimes "antigos". Exemplo: posse e disseminação de material pornográfico contendo crianças e adolescentes.

3. Crimes por computador incidentais: neles o uso do computador é incidental, não tendo relação direta com a atividade criminosa. Exemplo: registro de tráfico de drogas encontrados em um planilha eletrônica.

Ainda complementando a definição de Informática Forense, é preciso esclarecer a definição de *evidência digital*, que segundo Huebner et al. (2003) é:

> qualquer informação de valor probatório que é armazenada ou transmitida de forma digital.

Observe-se que tal definição não se limita apenas a computadores, mas áudio e vídeo em formatos digitais e dispositivos eletrônicos que armazenam ou transmitem dados como, por exemplo, aparelhos de fax, GPS e telefones celulares.

Analogamente, o termo *network forensics* ou perícias em rede tem sido cada vez mais utilizado. Em Palmer (2001), é apresentada a seguinte definição de *network forensics*:

> uso de técnicas cientificamente comprovadas para coletar, unir, identificar, examinar, correlacionar, analisar e documentar evidências digitais de múltiplas fontes digitais processando e transmitindo ativamente, com o propósito de descobrir fatos relacionados ao intento planejado ou sucesso apurado de atividades não autorizadas destinadas a perturbar, corromper ou comprometer componentes de sistema, bem como prover informação para auxiliar na resposta ou recuperação após atividades.

A definição das perícias em rede apresenta grande semelhança com as áreas de segurança de redes e resposta a incidentes, cujo objetivo principal é a proteção e a manutenção da disponibilidade de seus sistemas e redes. A Tabela 2.1, adaptada de Palmer (2001) apresenta os principais objetivos de cada área com relação à pesquisa e aplicação da perícia em Informática.

Tabela 2.2: Diferenças entre a segurança de computadores e a perícia em Informática, adaptada de Ruibin and Gaertner (2005)

Segurança	**Perícia**
Busca proteger o sistema de ataques	Não protege o sistema de ataques
Ação em tempo real ou logo após um incidente	Após os incidentes (*post mortem*)
Ambientes restritos para apresentação dos acontecimentos	A evidência é quase sempre apresentada para pessoal não técnico
Pode ser contornada por indivíduos confiáveis	A integridade da evidência é o mais importante

Em um cenário que não envolva o desejo de processar os criminosos, mas de proteger algum patrimônio ou informação, a ação invasora pode ser interrompida enquanto ocorre e, consequentemente, correções no sistema ou rede que implicam em perda de evidências do ocorrido podem ser feitas. Ou seja, ações que podem ser boas práticas em respostas a falhas de segurança podem ser devastadoras do ponto de vista pericial. Essa situação é encontrada pelos profissionais de segurança de redes e detecção de intrusão, cujos procedimentos nem sempre são consoantes com os procedimentos periciais, já que a persecução penal, nesse caso, é uma preocupação secundária. Portanto, apesar das semelhanças entre a perícia em Informática e a segurança de computadores, tanto em termos de conhecimento quanto de ferramentas, existem algumas diferenças significativas, como as apresentadas por Ruibin and Gaertner (2005), aqui adaptadas e exibidas na Tabela 2.2.

Em Ruibin and Gaertner (2005), umas das características atribuídas à perícia é que "podem ser conduzidos por especialistas, embora esse não costume ser o caso". Tal característica não se aplica à organização policial nem ao ordenamento jurídico brasileiro, que exigem que o perito criminal seja especialista em sua área de competência.

Embora em Palmer (2001) sugira-se que algo que diferencia as forças policiais das demais áreas é a impossibilidade de agir até que haja razão suficiente para se acreditar que um crime tenha ocorrido, na realidade atual, cada vez mais as forças policiais passam a agir com o intuito de prevenir a ocorrência de crimes também no meio cibernético, utilizando quando possível dados coletados em tempo real para fins investigativos. O uso de técnicas cada vez mais sofisticadas de proteção e ocultamento de dados como criptografia integral de disco[4] e esteganografia, também reforçam a necessidade de uma ação mais proativa para identificar e preservar vestígios.

Na perícia em Informática, há três tipos bem definidos de exames periciais:

1. exames em mídias de armazenamento como discos rígidos, mídias óticas como CD e DVD, disquetes, cartões de memória e *pen drive*, dentre outras mídias;

2. exames em programas e sistemas como exames em sistemas de bancos de dados e exames de *softwares* maliciosos como cavalos-de-troia, vírus e *keyloggers*;

[4]Do inglês *full-disk encryption*.

3. exames em redes de computadores como, por exemplo, exames de fluxos de dados capturados, mensagens de correio eletrônico, conteúdo de sítios na Internet e *logs* de dispositivos de rede como roteadores, *firewalls* e sistemas de detecção de intrusão.

Na Seção 2.1.3 são apresentados os procedimentos e técnicas utilizados nesses exames.

2.1.3 Procedimentos

O objetivo dos exames periciais sempre foi fornecer informações precisas obtidas por meio de métodos comprovados e bem compreendidos. A aplicação desses métodos em juízo só é possível após rigorosos testes científicos. Palmer (2001) cita o conhecido caso do uso de exames de DNA como meio de prova, que atualmente é bem aceito, mas só foi utilizado pela primeira vez dois anos após a sugestão de que o DNA poderia ser utilizado para identificar uma pessoa.

Segundo a legislação brasileira, o juiz não está adstrito ao laudo pericial, podendo formar a sua convicção com outros elementos ou fatos provados nos autos. Portanto, existe uma importância na confiabilidade dos métodos e na sua aceitação científica, de forma que o juiz, enquanto responsável pela decisão nos processos judiciais possa se apoiar na prova apresentada.

Esta seção apresenta diversos estudos para o estabelecimento de *frameworks* para a realização de exames periciais em Informática. As características essenciais a cada um desses procedimentos também são discutidas. Por fim, alguns exames específicos são descritos, bem como alguns trabalhos relacionados.

Segundo Palmer (2001), por definição, a perícia em Informática tem uma natureza investigativa e seus praticantes devem seguir um processo investigativo na realização de seu trabalho. Ao investigar crimes relacionados a computadores, deve ficar claro que os princípios básicos aplicados a cenas de crime "comuns" também se aplicam. De acordo com Huebner et al. (2003), a primeira coisa de que um investigador deve estar ciente é o *Princípio da Troca de Locard*, segundo o qual

> qualquer pessoa ou coisa entrando em uma cena de crime leva algo da cena consigo ou deixa algo de si para trás quando sai da cena,

ou como apresentado por Reith et al. (2002), toda atividade em um computador provavelmente produzirá uma modificação no sistema em que foi realizado como, por exemplo, modificações no sistema de arquivos ou, no mínimo, modificações na sua memória principal.

Nesse sentido, um princípio básico é o da preservação dos vestígios originais. Sempre que possível, procura-se trabalhar sobre uma cópia integral e exata dos dados originais. Um alto nível de integridade dos vestígios é necessário em todos os exames periciais de Informática, uma vez que materiais digitais são mais facilmente adulterados e forjados do que materiais físicos.

Com o desenrolar dos exames e a descoberta de evidências no material examinado, é importante manter a rastreabilidade dessas descobertas e de suas correlações. Da mesma forma, os dados são transformados e interpretados por ferramentas diversas. É desejável que todos os procedimentos realizados sejam claros e totalmente compreendidos pelos especialistas, embora nem sempre as ferramentas utilizadas permitam uma análise e avaliação mais profunda do seu funcionamento.

Huebner et al. (2003) apresentam uma série de passos investigativos convencionais que o especialista deve também aplicar em crimes por computador:

- assegurar e isolar o local do crime;
- registrar a cena do crime;
- conduzir uma busca sistemática por evidências;
- coletar e embalar evidências;
- manter a cadeia de custódia.

A cadeia de custódia é o registro de todas as pessoas que manusearam ou locais que mantiveram a custódia de uma evidência durante toda a sua existência, desde a coleta na cena do crime até o seu uso final no processo judicial. Turner (2005) discute extensivamente a importância de comprovar a procedência das evidências obtidas em meios digitais. A incapacidade de demonstrar a continuidade dessa cadeia de custódia em um processo tem um sério impacto na aceitação da evidência.

Com relação às evidências digitais, também é de suma importância estabelecer claramente a ligação entre os domínios físico e lógico. Turner (2005) cita um exemplo ilustrativo em que uma faca é encontrada com algumas impressões digitais em uma cena de crime. A faca caracterizaria o aspecto físico da investigação, enquanto o aspecto lógico é o exame e a interpretação que serão realizados sobre a impressão digital. Ou seja, o exame e a interpretação dos dados extraídos de um disco rígido, por exemplo, deve estar claramente associados ao dispositivo físico que representa o vestígio coletado originalmente na cena do crime.

Para auxiliar o processo de análise das evidências, Ruibin and Gaertner (2005) sugerem a inclusão do conceito de "relevância para o caso", definido como:

> a propriedade de qualquer pedaço de informação, que é utilizada para medir sua capacidade de responder às perguntas investigativas (quem, o que, onde, quando, porquê e como) em uma investigação criminal.

A Figura 2.2 ilustra os diferentes níveis de relevância sugeridos por Ruibin and Gaertner (2005).

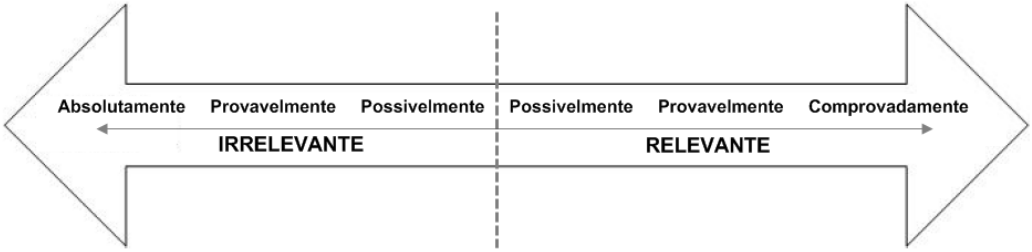

Figura 2.2: Níveis de relevância da informação para o caso, adaptada de Ruibin and Gaertner (2005)

Reith et al. (2002) apresentam uma ideia geral do processo investigativo. Após uma análise preliminar do caso, os investigadores formulam hipóteses com base nas circunstâncias do caso e coletam evidências com base nessas hipóteses. Em seguida, correlacionam

as evidências com as hipóteses. Se necessário, ajustam as hipóteses iniciais, repetindo o processo até que a interpretação do caso esteja altamente consistente. A Figura 2.3 ilustra esse processo.

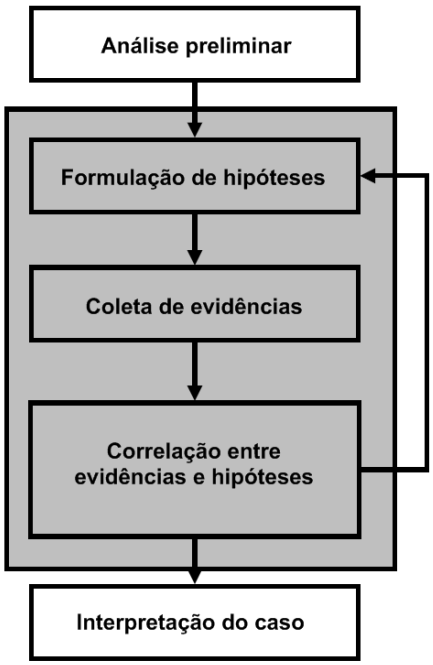

Figura 2.3: Processo investigativo, adaptada de Reith et al. (2002)

Na realização do processo investigativo citado anteriormente, quatro elementos chave são a (1) identificação, (2) preservação, (3) análise e (4) apresentação das evidências. Diversas propostas apresentando um número variável de passos para os exames periciais podem ser encontradas na literatura, embora todas as propostas estudadas apresentem de alguma forma esses quatro elementos. Dentre algumas, podem ser citados os trabalhos de Ciardhuáin (2004), Reith et al. (2002), Beebe and Clark (2005) e Ieong (2006).

Beebe and Clark (2005) apresentam uma lista não exaustiva de princípios que devem ser observados durante todo o processo, independentemente de quantos e quais passos são realizados. Princípios como preservação da evidência, documentação dos procedimentos, autoridade apropriada do examinador, respeito aos níveis de sigilo da informação, prioridade investigativa, fluxo de informação e controle, gerência do caso e realimentação para melhoria do processo são alguns exemplos. A Figura 2.4 ilustra essa ideia.

Segundo Beebe and Clark (2005), uma investigação digital, seja pericial ou não, requer rigor científico, o que pode ser facilitado pelo uso de processos padronizados, que ainda assim podem ser processos complexos. Nos trabalhos de Beebe and Clark (2005), Reith et al. (2002) e Ieong (2006) são avaliados vários dos modelos propostos ao longo dos anos em termos de suas fases e nível de especificidade. Alguns modelos apresentam apenas quatro fases enquanto outros sugerem até 13 fases distintas.

Ieong (2006) afirma que para romper a barreira técnica existente entre especialistas, investigadores e operadores do Direito como advogados, procuradores e juízes, deve ser utilizada uma proposta independente do caráter técnico e que incorpore questões legais. Com base nesse princípio, Ieong (2006) define oito papéis com responsabilidade distintas

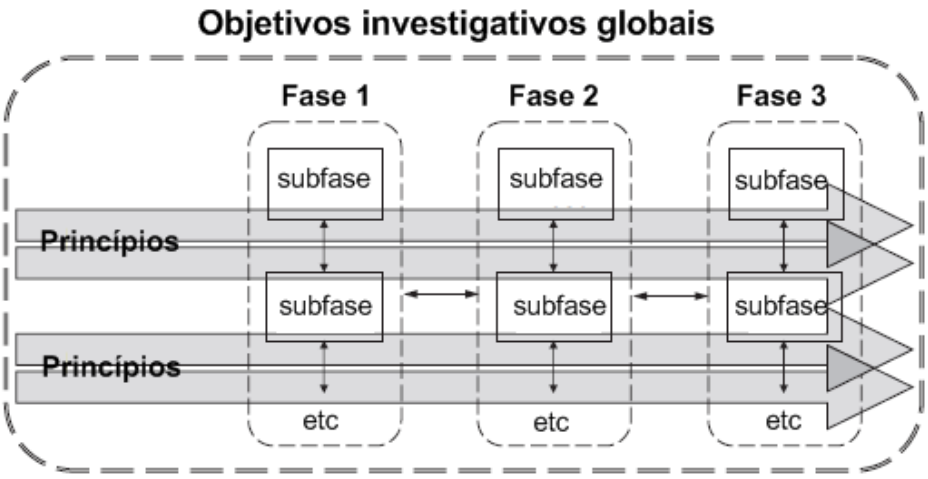

Figura 2.4: Aplicação de princípios ao longo do processo, adaptada de Beebe and Clark (2005)

em uma investigação digital. Mais de um papel pode ser realizado pela mesma pessoa se necessário. Os papéis sugeridos são:

- líder do caso: planeja todo o processo investigativo e determina se o caso deve continuar ou não;
- dono do sistema ou negócio: vítima do crime investigado (também pode ser considerado um suspeito);
- conselheiro legal: conselheiro disponível para orientar o líder do caso em questões legais;
- arquiteto de sistema/segurança: é entrevistado para que o líder do caso possa conhecer e avaliar o sistema ou ambiente a ser examinado;
- especialista forense: contratado para planejar os exames periciais necessários;
- investigador forense: realiza o trabalho de coleta e preservação das evidências digitais;
- analista forense: realiza a análise em si dos dados obtidos, considerando a hipótese estabelecida pelo especialista e
- representante legal: recebe os resultados do líder do caso e conduz o caso na esfera judicial.

A visão dessa proposta apresenta um caráter fortemente direcionado à perícia em ambientes privados e na ocorrência de crimes cujos alvos são sistemas e redes de computador. Embora ainda haja um intuito forense, os papéis sugeridos não refletem a realidade das investigações criminais realizadas pelos órgãos policiais. Essencialmente, do ponto de vista criminal teríamos um número reduzido de papéis (sem considerar o ordenamento jurídico de um país específico):

- investigador líder: determina a linha de investigação e os trabalhos necessários para sua realização;
- especialista forense: realiza todo o trabalho pericial, desde a coleta até os exames, podendo ser convocado a expor suas descobertas em juízo;
- analista de informações: analisa os dados resultantes da perícia segundo a linha de investigação determinada pelo investigador líder.
- promotor da ação penal: recebe o resultado da investigação e conduz o caso na esfera judicial penal.

Em resumo dos vários modelos estudados, Beebe and Clark (2005) sugerem uma divisão em seis fases, nas quais um número arbitrário de subfases pode ser definido, considerando a necessidade de sua execução conforme a natureza do caso. As seis fases são:

1. preparação (pré-incidente);
2. resposta ao incidente;
3. coleta de dados;
4. análise;
5. apresentação das descobertas;
6. encerramento do incidente.

A Figura 2.5 apresenta a relação das seis fases com a possibilidade de iteração entre elas ou de todo o processo.

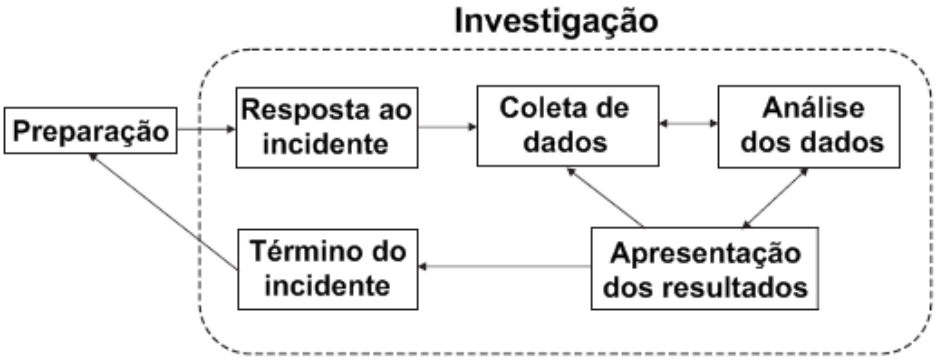

Figura 2.5: Fases do trabalho pericial, adaptada de Beebe and Clark (2005)

Este trabalho utilizará como referência essa divisão em seis fases proposta por Beebe and Clark (2005) com adaptações para o ponto de vista do exame pericial no âmbito policial, utilizando para tanto a seguinte nomenclatura:

1. planejamento;
2. atuação no local;
3. coleta de vestígios;

4. exame pericial;
5. laudo pericial;
6. revisão.

A seguir é apresentada uma descrição de cada uma das seis fases de alto nível.

Planejamento

Na fase de planejamento, avaliam-se os riscos de destruição e ocultação de evidências que podem ser causadas pelos investigados ou responsáveis pelo sistemas de interesse da investigação. Procura-se também levantar informações sobre as tecnologias, aplicativos, sistemas e servidores utilizados, de forma que os equipamentos e procedimentos forenses mais adequados sejam utilizados.

Outras atividades citadas nessa fase por Beebe and Clark (2005) dizem respeito ao conceito de *forensic readiness*, que representa um estado de prontidão da perícia para atender os casos. Para atingir esse estado de prontidão, atividades como o treinamento de pessoal e o desenvolvimento de procedimentos de preservação e manipulação de evidências devem ser realizadas e mantidas.

Atuação no local

Em um local de crime ou durante uma ação de busca e apreensão, o especialista avalia as condições encontradas no local. No caso de grandes ambientes de rede, o especialista estuda a topologia e configuração da rede e os sistemas em operação. Se necessário, entrevista os responsáveis técnicos pelo local, que poderão fornecer esclarecimentos. Então, busca minimizar o risco de perda de vestígios. Um exemplo é a existência de acesso remoto que permita a modificação de dados por pessoas que não estejam presentes no local.

Cabe lembrar que toda ação desenvolvida nessa fase deve ter seu devido suporte legal, com os mandados judiciais ou outros dispositivos legais adequados.

Coleta de vestígios

O propósito desta fase é a coleta em si dos vestígios, que pode incluir:

- dados obtidos de dispositivos de rede como sistemas de detecção de intrusão, roteadores e *firewalls*;
- dados obtidos de *hosts* individuais ou servidores como dados da memória volátil e relatórios de sistemas;
- mídias de armazenamento como discos rígidos, fitas de *backup*, disquetes, *pen drives*, CD e DVD;
- dispositivos contendo cartões de memória ou armazenamento interno como câmeras e filmadoras digitais, impressoras e MP3 *players*.

Deve-se lembrar que a maioria absoluta dos dispositivos de armazenamento de dados são de propósito geral. Assim, mesmo que um cartão de memória encontre-se em uma câmera fotográfica, não há limitação quanto ao armazenamento de qualquer tipo de arquivos nesse cartão.

Nesta fase inicia-se a cadeia de custódia. O material arrecadado no local deve ser bem identificado, embalado adequadamente e lacrado. A partir daí, toda mudança na custódia do material deve ser registrada.

No contexto policial, nos casos de crimes por computador incidentais, o material provavelmente não será coletado por um especialista forense, sendo coletado pela própria equipe de investigação. O especialista recebe então o material em laboratório para a realização dos exames.

Portanto, muitas vezes a participação do especialista já se inicia na fase do exame pericial, o que muitas vezes significa que o perito precisará examinar um grande volume de material de pouca relevância, pois a equipe de investigação não possui o mesmo nível de conhecimento do especialista para identificar e avaliar o potencial investigativo do material encontrado no local da ação policial.

Exame pericial

A fase do exame pericial é a mais complexa e demorada no processo de uma investigação digital. O objetivo é confirmar ou refutar as hipóteses da investigação com base nas evidências encontradas no material analisado, determinando a materialidade, autoria e a dinâmica dos fatos. Nessa fase buscam-se respostas para as perguntas básicas de uma investigação: o que (o ocorrido), o porquê (a motivação), como (os procedimentos), quem (as pessoas), onde (o local) e quando (o tempo).

Beebe and Clark (2005) subdividem esta fase em três subfases, apresentadas na Figura 2.6. Essas subfases são iterativas e dividem-se em levantamento, extração e exame dos dados.

Durante os exames periciais vários procedimentos podem ser realizados, conforme a natureza do caso em questão. Dentre esses procedimentos, podem ser citados alguns como:

- identificação de partições e sistemas de arquivos;
- análise de assinaturas de tipos de arquivos e uso de *hashes* criptográficos para identificar arquivos conhecidos;
- análise da cronologia do sistema de arquivos e de outros registros temporais;
- exame de chaves de registro e configurações do sistema operacional;
- identificação de fluxos de dados ocultos e esteganografia;
- identificação de programas de limpeza profunda de dados (*wipe*);
- recuperação de arquivos apagados e fragmentos de dados;
- pesquisas por arquivos contendo palavras-chave;
- extração de metadados em documentos e imagens
- análise do histórico e *cache* do navegador de Internet;
- quebra de senhas.

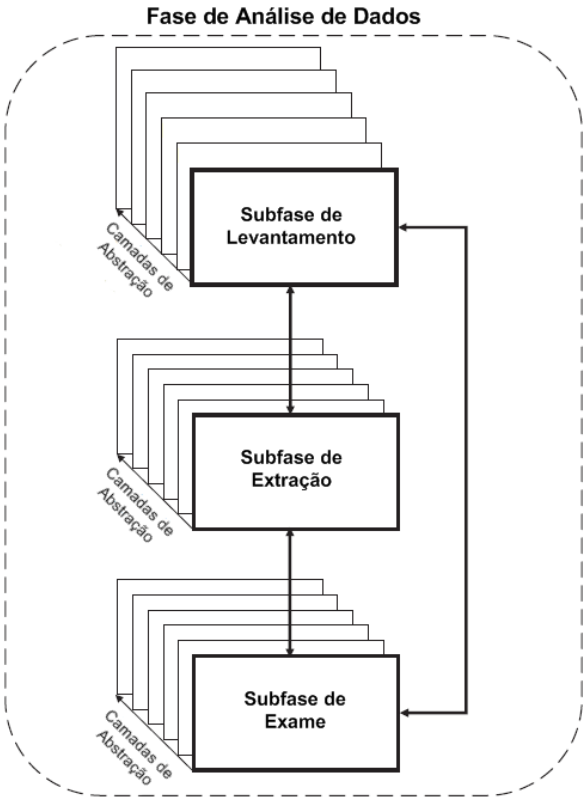

Figura 2.6: Subfases do exame, adaptada de Beebe and Clark (2005)

Alguns desses procedimentos, como a identificação de partições e sistemas de arquivos, fazem parte da subfase do levantamento de dados. A recuperação de arquivos, por exemplo, faz parte da subfase de extração de dados, enquanto a análise da cronologia do sistema de arquivos é um exemplo da subfase de exame dos dados.

Beebe and Clark (2005) apresentam um levantamento, exibido na Figura 2.7, de tarefas de análise de dados associados a vários objetivos. Por exemplo, para redução do volume de dados a serem examinados, são sugeridas a análise da assinatura de tipo dos arquivos, a análise de *hashes* e a ordenação cronológica das atividades no sistema de arquivos. Objetivos mais complexos como detectar um modificação não autorizada no sistema requerem a realização de mais tarefas como análise de chaves de registro do sistema, recuperação e análise de arquivos apagados e análise de *logs* de eventos, dentre outras.

Laudo pericial

Os resultados do exame pericial são apresentados na forma de um laudo pericial. Apesar da linguagem científica utilizada no laudo, a apresentação das conclusões deve ser de clara compreensão do público não especializado. Isso porque ele será utilizado no processo judicial que envolverá pessoal não especializado como procuradores, advogados e juízes. Segundo determinação do juiz, pode ser necessária a sustentação oral das conclusões em juízo.

Matriz de tarefas - fase de análise de dados

Tarefas de análise de dados	Objetivos da análise								
	Redução dos dados	Avaliação do nível de conhecimento	Recuperação de arquivos apagdos	Detecção de dados ocultos	Cronologia das atividades	Recuperação de e-mails	Histórico de uso da Internet	Recuperação de dados de chats	Detecção de modificação não autorizada no sistema
Análise de assinatura	✓	✓		✓					
Análise de hashes	✓	✓		✓					✓
Cronologia dos arquivos	✓				✓				✓
Análise de chaves de registro					✓				✓
Identificação de fluxos de dados		✓		✓					
Detecção de esteganografia		✓		✓					
Identificação de utilitários de wipe		✓							
R&A de arquivos apagados			✓					✓	✓
R&A do histórico de arq. apagados			✓		✓				✓
Recuperação de partições		✓	✓						
Pesquisa por palavras-chave			✓	✓		✓		✓	
Recuperação de arq. por assinatura			✓	✓		✓			
Análise de metadados				✓	✓				
Histórico de navegação na Internet					✓		✓		
R&A de Metadados de cookies					✓		✓		
R&A dos "favoritos" da Internet							✓		
R&A do cache da Internet							✓	✓	
Análise de dados proprietários						✓			
R&A de cliente de e-mail						✓			
R&A de webmail						✓			
R&A de arquivos temporários						✓		✓	
R&A de logs de eventos		✓			✓				✓
Análise lógica de arquivos						✓	✓	✓	✓

R&A = recuperação e análise

Figura 2.7: Matriz de tarefas e objetivos da análise de dados, adaptada de Beebe and Clark (2005)

O nível de detalhamento do laudo deve estabelecer a confiabilidade sobre a procedência das evidências (Turner, 2005). Dessa forma, o laudo deve apresentar os procedimentos aplicados nos exames com clareza e concisão, facilitando assim a sua compreensão. A descrição dos procedimentos também deve permitir que o exame possa ser reproduzido, para verificar suas conclusões.

Revisão

A fase de revisão não significa apenas o término das atividades relacionadas à perícia. Segundo Beebe and Clark (2005), é importante tentar preservar o conhecimento adquirido para melhorar as ações futuras.

Bruschi and Monga (2004) enfatizam a importância dos modelos de investigação digital permitirem revisões independentes do processo. Esse é o primeiro passo para uma revisão crítica do processo e a identificação de possíveis lições para o futuro. Em Palmer (2001), sugere-se o estabelecimento de um repositório de conhecimento pericial. Bruschi and Monga (2004) apresentam o que parece ser o primeiro trabalho nesse sentido.

Além da manutenção do conhecimento adquirido, algumas atividades devem ser realizadas como:

- definir novas ações investigativas ou persecutórias com base nas descobertas apresentadas;

- encaminhar as evidências para o destino adequado segundo o processo judicial como, por exemplo, envio à Justiça, restituição ao dono, destruição, perdimento e reutilização.

Todas as seis fases apresentadas anteriormente ainda poderiam se beneficiar de ferramentas mais especializadas e de novas pesquisas. Em especial, tendo em vista os desafios citados anteriormente nesse capítulo, um esforço maior deve ser direcionado para o desenvolvimento de tecnologias colaborativas, que permitam a divisão da complexidade e do volume de dados na fase do exame pericial.

2.2 Desafios

Muitos dos desafios enfrentados hoje na Informática Forense são produto dos grandes avanços tecnológicos observados nos últimos 15 anos. Esta seção apresenta alguns do desafios mais discutidos nessa década e que são os principais alvos de pesquisas e trabalhos como este.

Nas discussões do *First Digital Forensic Research Workshop*, ocorrido em 2001 e apresentado em Palmer (2001), alguns dos desafios de alta prioridade citados então eram a confiabilidade da evidência digital e perícias em ambientes de rede. Essas questões ainda estão presentes atualmente e em escala cada vez maior. As evidências digitais tornaram-se cada vez mais comuns e as redes de computadores maiores e mais presentes. Comentou-se então que a ubiquidade dos sistemas de informática e equipamentos eletrônicos cada vez mais indicava que um dia todos os crimes teriam uma "ciberdimensão".

No *5th Annual Digital Forensic Research Workshop*, ocorrido em 2005 e editado por Reust (2006), as discussões foram divididas em quatro tópicos centrais: ocultamento de evidências (e suas técnicas de análise), escalabilidade e automação para enfrentar a sobrecarga de evidências digitais, ferramentas forenses e aspectos legais. Este último é um desafio permanente que foge ao escopo deste trabalho. Cabe a cada força policial avaliar a adequabilidade de alguns dos procedimentos sugeridos que podem ou não ser aplicáveis à legislação ou ao processo legal empregado no seu país.

Alguns casos envolvendo evidências digitais estão se tornando muito grandes em termos de capacidade dos discos rígidos, extensão dos *logs*, quantidade de computadores e mídias de armazenamento de dados. Sommer (2004) afirma que a Justiça não tem compreendido as implicações desses casos grandes em termos de gerenciamento dos recursos e procedimentos necessários antes e durante um julgamento. Ainda segundo Sommer (2004), alguns métodos simplesmente não podem ser aumentados de escala. Em grandes investigações, somente o registro da cadeia de custódia das evidências gera um grande esforço administrativo, além das inúmeras provas derivadas dos discos rígidos como arquivos recuperados, arquivos de *log*, planilhas, resultados de pesquisas por palavras-chave. Provas essas que serão então examinadas exaustivamente pela defesa dos acusados e por outros especialistas e poderão gerar solicitações de novos exames ou o fornecimento do conteúdo original completo, incluindo o que não foi utilizado pela acusação, para uso pela defesa.

Em Reust (2006), é citado um caso envolvendo 450 terabytes de dados armazenados em fitas magnéticas, dos quais apenas 6% apresentava dados relevantes e um cujo custo de análise foi de 250.000 dólares americanos. Por isso, um dos desejos enfatizadas pelas

forças policiais é conseguir obter os dados úteis de grandes conjuntos de dados em tempo hábil com o menor custo possível. Muitas vezes a equipe policial pode se deparar com o dilema de coleta ou não evidências digitais uma vez que elas podem atrasar a conclusão dos trabalhos. O uso de sistemas distribuídos para processar os casos mais rapidamente e o uso de colaboração poderiam ajudar a solucionar esse problema, como sugerido em Roussev and Richard III (2004). Sugere-se o estabelecimento de uma abordagem "piramidal", em que o processamento básico é feito pela polícia local e a análise mais avançada pelos laboratórios periciais. O Dr. Eugene Spafford afirma em Palmer (2001) que precisa-se de ferramentas que direcionem a atenção para informações realmente úteis e rapidamente indiquem o material de interesse para a investigação.

Portanto, o grande desafio na prática tem sido a sobrecarga de evidências digitais. Com mídias de armazenamento digital de capacidades cada vez maiores e presentes em dispositivos tão diversos, aumentou o volume de dados contido nas evidências digitais envolvidas em investigações criminais. Assim surgiu também a necessidade de identificar rapidamente as fontes de dados de interesse, correlacionar as várias evidências e utilizar esses dados como fontes de inteligência nas investigações. Segundo Case et al. (2008), os peritos tornaram-se vítimas do próprio sucesso. Os meios de armazenamento digital como discos rígidos e *pen drives* são uma fonte tão valiosa de informação que passaram a ser rotineiramente arrecadados em investigações. Multiplicando o aumento da coleta de evidências digitais e o aumento da capacidade média das mídias de armazenamento, obtém-se o grande crescimento no volume de trabalho pericial sobre as evidências digitais.

A Figura 2.8 ilustra um caso real com aproximadamente 2,5 milhões de itens extraídos de dez discos rígidos[5] em um caso de fraude em licitações. Nesse caso, a ferramenta utilizada precisou de cinco dias e oito horas ininterruptas para processar as evidências. Antes disso, o perito não pode realizar qualquer análise utilizando a ferramenta.

Para combater esse crescimento, uma das necessidades levantadas foi a de conseguir maior velocidade e inteligência no processamento das evidências digitais. Segundo Reust (2006), em um levantamento realizado entre especialistas da área de perícia em computadores, 25% dos participantes apontaram o volume de dados como a principal barreira para processar os vestígios em tempo hábil. Os participantes apontaram também a velocidade de processamento (14%) e a falta de automação (12%) como as principais limitações na capacidade de análise. Roussev and Richard III (2004) afirmam que as soluções atuais não são capazes de atender satisfatoriamente as necessidades dos examinadores, nem de superar as dificuldades futuras. Assim, há a necessidade de se buscar novas soluções.

Uma ideia citada em Reust (2006) para lidar com a sobrecarga de dados é o uso de IA em domínios em que essa possa ser eficiente. O objetivo não seria um investigador forense artificial[6], mas apenas objetivos restritos como processamento de linguagem natural para criar sumários automatizados, agrupar imagens ou fazer uma busca de tópicos em vez de uma busca por palavras-chave. Segundo Ruibin and Gaertner (2005), a perícia em Informática terá um papel cada vez mais importante na investigação criminal. Porém, ao examinar o progresso atual do uso da inteligência computacional na perícia, ela parece estar muito atrás de outros campos de pesquisa e implementação. Este trabalho busca ampliar esses horizontes de aplicação de IA na Informática Forense.

[5] A ferramenta contabiliza as partições dos discos rígidos como evidências distintas, totalizando então 22 evidências.

[6] Traduzido de *AI Digital Forensic Investigator*.

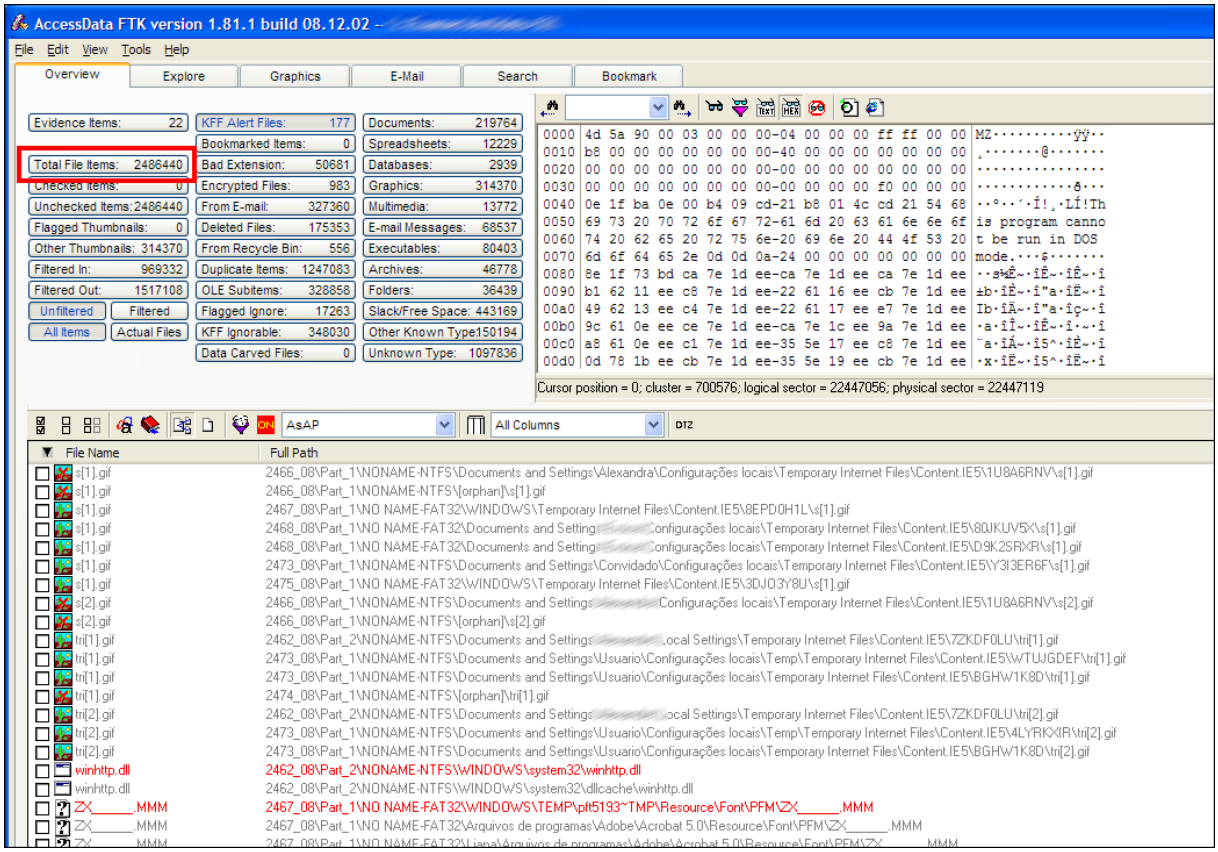

Figura 2.8: Exemplo de caso com grande volume de conteúdo

Case et al. (2008) apontam alguns problemas decorrentes das limitações das ferramentas atuais. Um deles é a priorização imprópria das evidências, pois ainda não é possível priorizar uma evidência facilmente baseando-se na informação que ela contém. Outro é perda de oportunidades de correlacionar dados entre evidências, uma vez que cada disco rígido costuma ser examinado independentemente, sem uma chance de "ligar os pontos" em grandes casos envolvendo vários discos rígidos. Há ainda, segundo Case et al. (2008), uma ênfase imprópria na recuperação de documentos, em detrimento da recuperação de informações mais profundas.

Uma tendência observada nos últimos anos é a expansão da perícia além do simples exame dos discos rígidos. A análise de memória volátil e de sistemas em operação[7] tem recebido bastante atenção em termos de pesquisa e de ferramentas específicas. É interessante notar que a perícia em sistemas em operação "desrespeita" um dos princípios básicos da perícia em mídias de armazenamento, que é a preservação total dos dados originais. Isso porque qualquer atividade em um sistema em operação causa mudanças nos dados armazenados na memória. Como tal situação é inevitável, Huebner et al. (2003) argumentam que evidências coletadas dessa forma tem que ser aceitáveis em juízo. O monitoramento e análise de dados de sistemas em operação e em redes de computadores será essencial para o trabalho policial, tendo em vista o aumento no volume de evidências digitais apresentadas pelos casos.

[7] Chamada de *live analysis*

As perícias em rede também apresentam uma complexidade que supera a já complexa análise de computadores isolados e apresenta novas dificuldades, como a escassez de ferramentas que permitam a coleta distribuída de evidências e a correlação dos volumes de dados em conjuntos compreensíveis para permitir sua análise. Segundo Huebner et al. (2003), a mineração de dados será uma área chave de pesquisa.

Além das dificuldades discutidas acima, que afetam de maneira geral o trabalho pericial em Informática, Huebner et al. (2003) apresenta ainda diversos desafios técnicos como:

- sistemas de arquivos que permitem ocultar dados do usuário comum, sendo visíveis apenas se utilizadas ferramentas especiais;
- propriedades e mecanismos de sistemas operacionais e aplicativos sem documentação ou utilizados para ocultar dados;
- armazenamento de dados *online*, que permite o armazenamento de dados em serviços da Internet, cujo acesso pode ser difícil ou pode encontrar-se fora da jurisdição legal daquela polícia e pode até requerer ações demoradas de cooperação internacional;
- uso extensivo de criptografia forte, que sugerem a necessidade de um trabalho maior de investigação para evitar que as evidências digitais do suspeito estejam protegidas dessa forma;
- dispositivos móveis de alta capacidade e dimensões muito reduzidas, que podem ser facilmente destruídos ou ocultados, ou que podem ser utilizados para evitar que dados importantes fiquem armazenados nos discos rígidos dos computadores utilizados;
- serviços *online* diversos como *webmail*, redes sociais ou programas de mensagens instantâneas cujas vestígios encontram-se nas mãos dos provedores dos serviços, o que dificulta sua coleta.

Deficiências no treinamento e certificação dos especialistas também são problemas comumente citados. A grande variedade de tecnologias e situações que podem ser encontradas nos casos de crimes por computador, dificulta a preparação dos especialistas. Assim, com o tempo a experiência dos especialistas torna-se bastante distinta e o nível de preparação irregular. Em Reust (2006), afirma-se que o conhecimento detalhado de sistemas operacionais, sistemas de arquivos e aplicativos está restrito a alguns especialistas. Reith et al. (2002) afirma ainda que é improvável que um especialista isolado tenha todas as habilidades necessárias para interpretar as evidências de um caso não trivial. Essas preocupações sugerem a pesquisa e o desenvolvimento de ferramentas que auxiliem na reutilização de conhecimento de casos anteriores em suporte a novos casos, o que constitui um dos objetivos deste trabalho.

Por fim, há ainda diversas discussões com relação à padronização de formatos de dados utilizados pelas ferramentas, definições de termos e procedimentos, interoperabilidade entre ferramentas forenses e nível de confiabilidade das ferramentas utilizadas.

Portanto, embora tenha-se observado nesta década um avanço substancial nas técnicas e ferramentas utilizadas na perícia de sistemas computacionais, muitos dos desafios identificados há muitos anos permanecem em aberto. A complexidade e a dimensão dos

novos desafios e problemas enfrentados continua aumentando e as ferramentas disponíveis atualmente ainda não apresentaram soluções satisfatórias para nenhum dos desafios citados anteriormente.

2.3 Trabalhos Correlatos

Nas seções seguintes são apresentados trabalhos cujos objetivos estão relacionados aos objetivos deste trabalho. Durante esta pesquisa não foram encontrados estudos específicos sobre a aplicação de um Sistema Multiagente (SMA) para a perícia em Informática. Dessa forma, os trabalhos citados a seguir apresentam alguma relação com partes diversas da proposta apresentada no Capítulo 4.

Automatização e distribuição de processos

O problema de aumentar a eficiência e eficácia do exame pericial de computadores em face da demanda crescente tem sido abordado de duas perspectivas distintas. Uma delas, apresentada em Kenneally and Brown (2005), busca a redução do volume de material enviado para exame através do desenvolvimento de técnicas e ferramentas para realização de uma pré-análise ou triagem automatizada. A outra, apresentada por Roussev and Richard III (2004), busca o aumento da automatização dos processos e do melhor aproveitamento dos recursos computacionais disponíveis por meio da distribuição do processamento das evidências.

Com relação à primeira perspectiva, Kenneally and Brown (2005) levantam um conjunto de dificuldades encontradas na coleta de evidências em sistemas computacionais causado pelo grande volume de dados, pela complexidade dos ambientes computacionais e por restrições legais. Argumentam ainda que uma mudança nas técnicas atuais não torna os resultados menos confiáveis do ponto de vista legal e que continuar insistindo nas técnicas atuais para casos envolvendo grandes volumes de dados em ambientes de rede não é mais possível. Na proposta apresentada pelos autores, deve ser realizada uma filtragem e redução do conteúdo a ser copiado e encaminhado para exame. Os autores argumentam que tal procedimento já é realizado nos casos em que há limitações legais ou técnicas para a cópia completa dos dados e apresentam diversos casos em que as decisões judiciais apoiaram procedimentos como os contidos nessa proposta. Não há, no entanto, referência a qualquer ferramenta que seja capaz de permitir a aplicação da proposta de maneira simples e confiável.

Para a segunda perspectiva, Roussev and Richard III (2004) argumentam que alguns dos exames realizados em laboratório tornam-se impraticáveis com o aumento do volume de dados. Um exemplo é a pesquisa de palavras-chave: o processo de indexação, que consiste na criação de um índice das palavras encontradas e suas ocorrências na mídia examinada, consome muito tempo e espaço em disco. A pesquisa sem índice, que varre toda a extensão do disco, também é muito lenta em discos rígidos muito grandes. Segundo os autores, algumas das ferramentas utilizadas atualmente já atingiram seu limite com o volume de dados encontrado atualmente. Para superar essa limitação, os autores propõe o uso de processamento distribuído, que até o momento somente foi visto na prática para a quebra de senhas. Os autores apresentam um protótipo para pesquisa de palavras-chave, utilizando uma arquitetura mestre-escravo com bons resultados.

Reutilização de conhecimento, correlação e utilização de IA

Os especialista mais experientes possuem maior conhecimento sobre as fontes de evidências mais interessantes para cada caso. Para os especialistas iniciantes, ter acesso a esse conhecimento é muito importante para o seu aprendizado e para a obtenção de melhores resultados. Isso também fomenta a capacitação continua e o nivelamento do conhecimento entre os especialistas.

Em Bruschi and Monga (2004) é proposto um modelo que permita organizar o conhecimento forense de uma maneira reutilizável. Assim, experiências passadas podem ser utilizadas para treinar pessoal novo, fomentar o compartilhamento de conhecimento e expor as informações coletadas para controle de qualidade independente. Seu objetivo, portanto, é fornecer uma abordagem sistemática e uma ferramenta para produzir conhecimento forense reutilizável para ser utilizado como suporte em novas investigações organizar as experiências passadas para fomentar o compartilhamento de conhecimento entre especialistas forenses registrar a informação coletada de forma a facilitar o controle de qualidade.

A abordagem sugerida por Bruschi and Monga (2004), baseia-se na utilização de *frameworks* de hipóteses, que seria uma abordagem com alto grau de reuso em casos similares e mesmo em casos diferentes que compartilhem partes similares. Para isso, uma hipótese inicial H é decomposta em sub-hipóteses cuja verificação é mais simples, conforme a Equação 2.1. Essa decomposição não é única e depende do raciocínio do especialista. Por isso ela deve ser registrada para uso futuro e revisão.

$$H \rightarrow H_1, H_2, H_3, \ldots, H_n \tag{2.1}$$

Como exemplo, considere-se a seguinte hipótese (Bruschi and Monga, 2004):

> H: A conta de e-mail `user@domain`, registrada pela usuária Alice foi utilizada para enviar uma mensagem nociva M para o usuário Bob. Alice foi a autora e remetente de M.

Um possível decomposição dessa hipótese poderia ser feita da seguinte forma:

> H_1: Alice enviou a mensagem M do computador C.
> H_2: o programa de transferência de e-mails `sendmail` estava instalado em C e configurado para utilizar `user@domain` no campo *remetente* do cabeçalho.
> H_3: quando M foi enviada (no tempo T), C estava em uso.
> H_4: quando M foi enviada (no tempo T), C estava conectado à Internet
> ...
> $H \rightarrow H_1, H_2, H_3, H_4$

Essa decomposição poderia ser feita repetidamente até que a nova decomposição não apresentasse nenhuma simplificação. O resultado final é uma cadeia de raciocínio, que é uma grafo de hipóteses acíclico.

Da mesma forma, a síntese é a recomposição das soluções parciais do problema decomposto, conforme apresenta a Equação 2.2:

$$H \mapsto E_1, E_2, E_3, \ldots, E_n \tag{2.2}$$

Um exemplo de síntese para a hipótese H_3, *quando M foi enviada (no tempo T), C estava em uso*, é a seguinte (Bruschi and Monga, 2004):

> E_1: verificar se arquivos foram modificados, criados, acessados ou apagados em T.
> E_2: verificar se há arquivos que contém informação sobre atividade do usuário (ex.: navegação na Internet) em T.
> E_3: verificar se há arquivos contendo informações sobre atividade do sistema (ex.: registros de eventos do sistema operacional) em T.
> ...
> $H_3 \mapsto E_1, E_2, E_3$

Uma falha nas evidências associadas à hipótese H_3 enfraquece a hipótese e consequentemente enfraquece a hipótese inicial H. A confirmação das evidências, por outro lado, reforça as hipóteses.

Bruschi and Monga (2004) sugere que essas decomposições e sínteses sejam utilizadas para revisar e avaliar o trabalho do especialista. O armazenamento e reuso dessa informação também pode servir de guia para examinadores menos experientes.

Segundo Case et al. (2008), a ampla maioria das ferramentas forense atuais tem foco em extrair informações diretas de artefatos individuais como nome, tamanho, localização, palavras-chave, dentre outras e apresentá-las ao especialista. O que leva a uma abordagem entediante de "navegar e procurar". A Figura 2.9, apresentada por Ruibin and Gaertner (2005), ilustra essa situação. Com o crescimento rápido da complexidade dos sistemas, torna-se cada vez mais difícil para o especialista realizar um exame completo, oportuno e confiável. Segundo estatísticas do FBI, o tamanho médio dos casos triplicou em três anos, passando de 80 GB em 2003 para 250 GB em 2006 (Case et al., 2008).

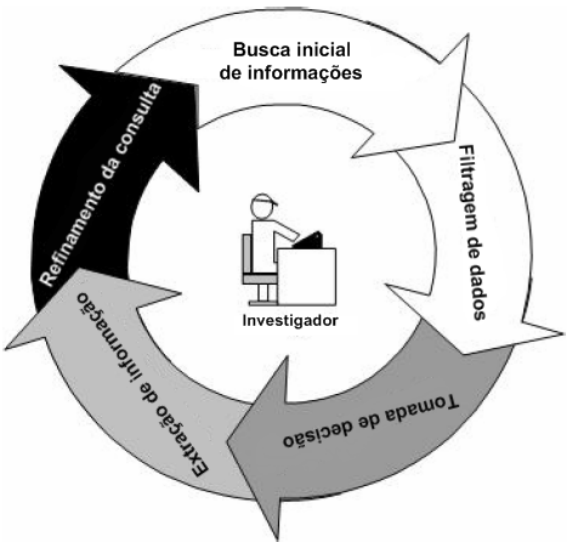

Figura 2.9: Ciclo de busca e extração de informações, adaptada de Ruibin and Gaertner (2005)

Essa tendência de crescimento tem tido grande impacto no trabalho pericial, pois apenas os procedimentos iniciais de cópia de dados, extração de evidências e pré-processamento

têm consumido um tempo significativo, aumentando o tempo de resposta para os solicitantes dos exames.

A ferramenta proposta por Case et al. (2008), chamada *Forensics Automated Correlation Engine* (FACE) permite organizar e correlacionar rapidamente as evidências de várias fontes, incluindo *dumps* de memória, tráfego de rede e imagens de sistemas de arquivos, arquivos de *log* e arquivos de configuração e contas de usuário. Essa ferramenta demonstra a possibilidade de análise integrada desses diversos tipos de evidências, permitindo que o especialista tenha uma visão mais coerente do caso.

Ruibin and Gaertner (2005) propõem o uso de um módulo de extração automático de evidências. Para isso um sistema especialista prepara um conjunto de palavras-chave com base no perfil do caso que serve de entrada para esse módulo. O módulo utiliza técnicas de extração e recuperação de informação para gerar novas palavras-chave e realizar novas pesquisas nos dados. Por fim, um módulo inteligente decide pela relevância do resultado para o caso. Se o dado for relevante, ele é acrescentado ao resultado final, ordenado por relevância. A Figura 2.10, apresentada por Ruibin and Gaertner (2005), ilustra a posição desse módulo no processo de exame pericial.

Com relação ao processo de coleta de evidências digitais, Turner (2006) afirma que ele tradicionalmente é pouco sofisticado. Em geral, copia-se a integridade das mídias de armazenamento de interesse. No entanto, é reconhecido que nem sempre é possível ou prático coletar todo e qualquer dado. No processo de coleta seletiva de dados, decide-se não coletar todas os dados possíveis. Uma situação de coleta seletiva pode ser imposta, por exemplo, por uma questão legal, que não permita a cópia integral de todos os dados.

Assim, Turner (2006) sugere a prática de coleta de dados seletiva e inteligente utilizando o conceito de *Digital Evidence Bags* (DEB), que representam contêineres universais de evidências digitais de quaisquer fontes. No DEB fica também registrada a procedência da evidência e a cadeia de custódia ao longo de todo o processo. A Figura 2.11 apresenta esse conceito.

A coleta seletiva pode ser realizada de maneira manual, semi-automatizada ou totalmente automatizada. Na coleta semi-automatizada, o especialista forense decide os tipos de arquivos ou categorias de informação que serão coletados. Na coleta automatizada, o especialista apenas indica a fonte e o destino dos dados e um sistema realiza a coleta, com base em parâmetros pré-configurados ou em circunstâncias particulares daquele caso ou investigação.

Uma abordagem inteligente de coleta, segundo Turner (2006), é o processo de capturar o conhecimento e a experiência dos especialistas do domínio em um sistema inteligente. Isso permite que um investigador que não tem proficiência técnica e que só tem conhecimento do tipo de investigação possa utilizar esse sistema de coleta automatizado.

Por exemplo, em investigações de fraude ou roubo de propriedade intelectual, mesmo sem conhecer os tipos de arquivos e localidades em que a informação se encontra, o investigador pode coletar os dados de interesse do caso. Ou seja, apenas a partir da seleção do tipo de investigação, o sistema de coleta tem a inteligência necessária para coletar tudo que seria normalmente relevante.

Segundo Turner (2006), sua proposta não é uma tentativa de utilizar métodos ou técnicas de Inteligência Artificial (IA) para decidir o que capturar. Ela poderia, no entanto, alertar o examinador para a presença de categorias de material fora da linha inicial da investigação. O autor levanta alguns riscos e dificuldades dessa abordagem:

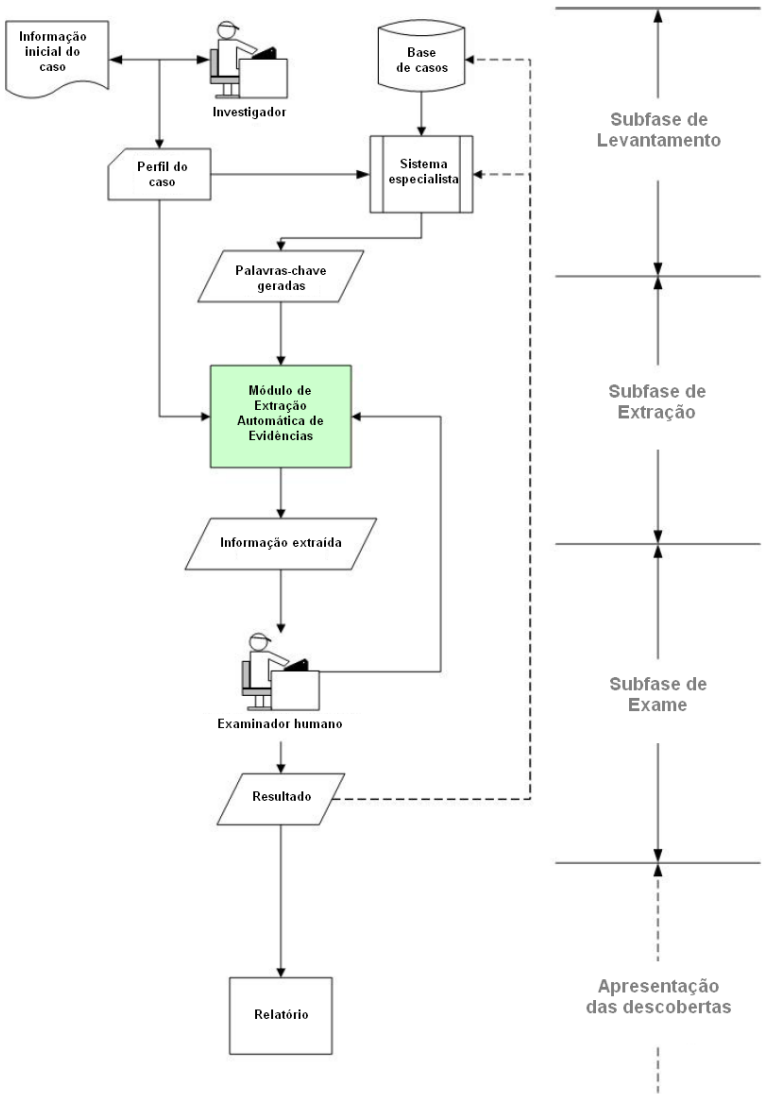

Figura 2.10: Módulo automatizado de extração de evidências, adaptada de Ruibin and Gaertner (2005)

- Como capturar e combinar o conhecimento dos especialistas do domínio pericial, familiarizados com as complexidades técnicas, com os especialistas do domínio jurídico?
- Como saber que tudo que é relevante para o caso foi coletado e que nada foi deixado para trás?

Esses pontos não são respondidos por Turner (2006), mas são pontos que devem ser considerados no futuro. A proposta deste trabalho, de certa forma, depara-se com esses desafios, ainda que não se concentre na fase de coleta de dados.

Em Alinka et al. (2006) é descrita uma abordagem baseada em XML, chamada *XML Information Retrieval Approach to digital Forensics* (XIRAF), cujo objetivo é gerenciar e pesquisar evidências digitais. Nessa abordagem cada ferramenta gera anotações em XML que podem se referir a uma região de bytes ou a um arquivo. XIRAF armazena essas

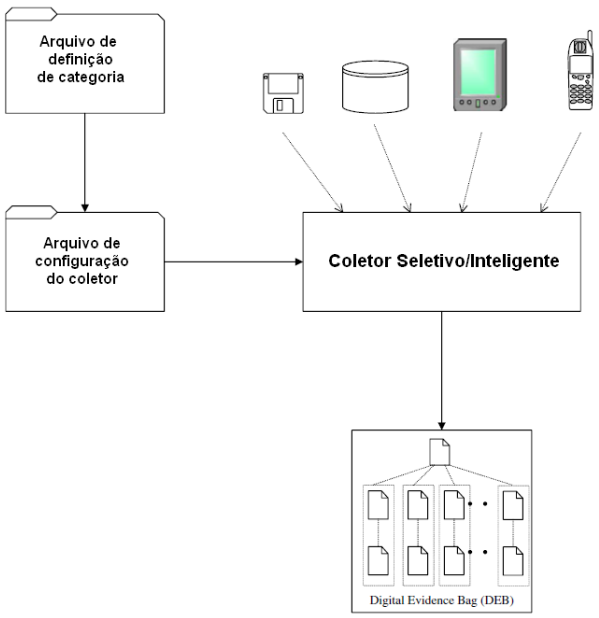

Figura 2.11: Coleta de dados seletiva, adaptada de Turner (2006)

anotações em um banco de dados XML que pode ser pesquisado utilizando a linguagem XQuery.

Um repositório de ferramentas mantém um conjunto de ferramentas de extração de características. Cada ferramenta é constituída de um programa que realiza a extração e de um *wrapper* que descreve como utilizar o programa e converter seus resultados para XML. Segundo Alinka et al. (2006), muitos dos programas forenses baseados em linha de comando podem ser aproveitados após a definição do *wrapper* para elas. O tempo necessário para criar o *wrapper* para um programa forense dependo do seu formato de saída. Se a ferramenta já produz uma saída em XML, a adaptação é rápida. Caso contrário, a saída do programa deve ser transformada. XIRAF roda as ferramentas em processos separados, evitando que se uma ferramenta apresente um erro todo o sistema pare. A Figura 2.12 ilustra esse processo.

A proposta deste trabalho é semelhante a do XIRAF no sentido de que também utiliza um repositório central com os resultados obtidos dos agentes inteligentes. Os programas forenses também podem ser aproveitados após serem encapsulados por um agente inteligente forense, que é capaz de examinar de maneira autônoma um conjunto de arquivos. A execução de cada um dos agentes é distribuída, isolada do restante da plataforma, de forma que uma falha isolada não compromete a execução do sistema como um todo.

Ontologias forenses

Dois trabalhos relacionados à concepção de ontologias relacionadas à perícia de Informática são apresentadas em Brinson et al. (2006) e Harrill and Mislan (2007). A definição de ontologia em si é apresentada na Seção 3.3.

Na primeira proposta, sugere-se a definição de uma ontologia de alto nível, que permita definir as áreas de especialização, certificação e educação no domínio da Informática Forense. Duas divisões principais podem ser identificadas na proposta: uma tecnológica,

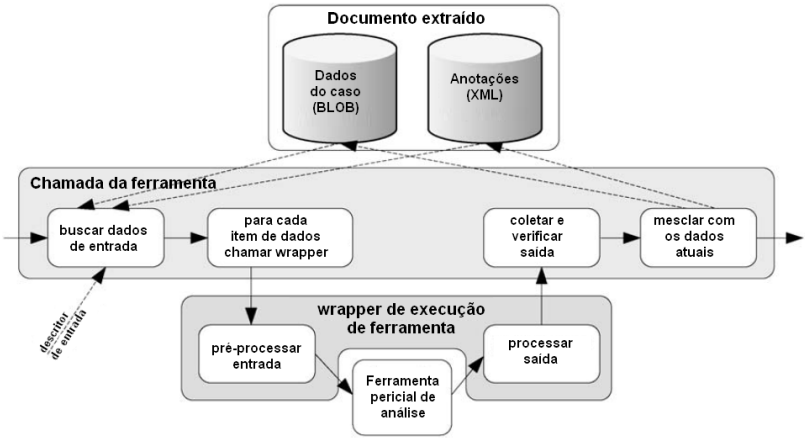

Figura 2.12: Reutilização de ferramentas via *wrappers*, adaptada de Alinka et al. (2006)

que define questões de *hardware* e software, tanto do ponto de vista do exame pericial quanto das ferramentas forenses, e outra profissional, que define as áreas de especialização policial, acadêmica, de pesquisa e privada. A Figura 2.13, adaptada de Brinson et al. (2006), apresenta uma parte dessa ontologia relacionada ao domínio penal.

Do ponto de vista da proposta deste trabalho, o altíssimo nível dos conceitos estabelecidos na ontologia proposta por Brinson et al. (2006) ainda é bastante distante dos conceitos utilizados pelos agentes inteligentes, que dizem respeito a exames periciais específicos e características mais detalhadas de sistemas operacionais, sistemas de arquivos e do formato e conteúdo dos arquivos em si.

A segunda proposta apresenta uma ontologia para a perícia em dispositivos de pequena escala como telefones celulares, PDAs, cartões de memória, entre outros. Embora apresenta definições interessantes com relação às mídias de armazenamento removíveis, não é um trabalho exaustivo, além de ter um domínio bem específico e restrito.

Além desses trabalhos, Bogen and Dampier (2005) propõem o uso de ontologias e modelagem de domínio como uma abordagem estruturada para analisar fatos de um caso, identificar os conceitos mais relevantes, determinar relações críticas entre conceitos e documentar essas informação. Essa proposta, no entanto, é destinada à fase de planejamento de grandes investigações.

Procedimentos periciais específicos

Esta seção apresenta alguns estudos correlatos relacionados à procedimentos periciais específicos. Embora o foco principal deste trabalho não seja o seu aperfeiçoamento, os trabalhos apresentadas nesta seção servem como motivadores dos agentes inteligentes concebidos no protótipo inicial do SMA proposto neste trabalho e também como indicadores de futuras melhorias nesses agentes.

Utilização de *hashes*

Por definição, uma função de *hash* criptográfico é uma função unidirecional que recebe um bloco de dados de tamanho arbitrário e produz um saída de tamanho fixo, tal que qualquer

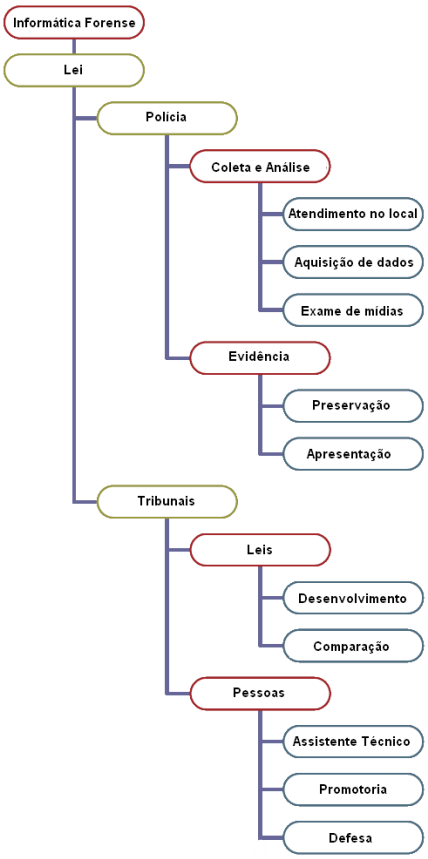

Figura 2.13: Parte de uma ontologia para o domínio da Informática Forense, adaptada de Brinson et al. (2006)

Texto	**Valor de *hash***
Eis o hash MD5 deste pequeno texto.	`ec7687899557ccbee783f727c341373d`
Eis o hash MD5 deste pequeno texto!	`30327c2890114ba25b35cc5c23d77894`

Tabela 2.3: Exemplo de cálculo de *hash*

mudança, acidental ou intencional nos dados, gerará um valor de *hash* completamente diferente.

A Tabela 2.3 apresenta um exemplo de cálculo de *hash* criptográfico utilizando o algoritmo *Message-Digest Algorithm 5* (MD5) de dois textos com apenas um sinal de pontuação diferente.

Qualquer alteração na frase gera um valor completamente diferente. Ou seja, é extremamente improvável que os dados originais sejam alterados e mantenham o mesmo valor de *hash* criptográfico original. Da mesma forma, é extremamente improvável que duas mensagens diferentes tenham valores de *hash* criptográfico igual, embora esse fenômeno, chamado de colisão, seja probabilisticamente possível.

Com isso, é possível utilizar o valor do *hash* criptográfico de um arquivo para localizar arquivos iguais a ele. Também é possível construir uma base de *hashes* que armazenam uma listagem de arquivos conhecidos. No exame pericial, essas duas técnicas são utilizadas para reduzir o volume de dados a ser examinado. Os arquivos duplicados são facilmente

identificados e as cópias não precisam ser examinadas novamente.

O *National Institute of Standards and Technology* (NIST), nos EUA, organiza e mantém a *National Software Reference Library* (NSRL), uma biblioteca de hashes de *softwares* diversos. No caso em que um computador tenha instalado o sistema operação Microsoft Windows 2000, uma redução de mais de 4.000 arquivos é obtida com o uso dessa biblioteca. Em um estudo apresentado em Mead (2006), em um exame de mais de 10,5 milhões de *hashes* de arquivos utilizando os algoritmos MD5 e SHA-1, não foi constatada nenhuma ocorrência de colisão.

A NSRL mantém uma base de *softwares* predominantemente em inglês, embora inclua *software* em outras línguas. Com isso, a redução no volume de arquivos conhecidos não é tão significativa quando utilizam-se essas bibliotecas de *hashes* para examinar sistemas com softwares predominantemente em português. Outro problema com essas bibliotecas é o seu custo de atualização e manutenção, uma vez que *softwares* novos ou novas versões são lançados diariamente. Além disso, quanto maiores as bibliotecas, mais demorado é o processo de identificação de arquivos conhecidos.

As ferramentas periciais mais utilizadas atualmente, no entanto, ainda não fornecem meios mais inteligentes de utilizar essas bibliotecas de *hashes*, tais como:

- as ferramentas atuais permitem a criação de bibliotecas personalizadas de *hash*, mas não auxiliam o usuário nessa tarefa;
- as ferramentas poderiam realizar uma identificação rápida dos *softwares* conhecidos comparação de alguns valores chave de *hash*, evitando comparações desnecessárias;
- identificar arquivos repetidos em várias casos e sugerir sua inclusão na biblioteca de *hashes*.

Os valores de *hashes* também podem ser utilizados para alertar o examinador quanto à presença de arquivos específicos de interesse. Em uma investigação de exploração sexual de crianças e adolescentes, uma base de *hashes* de imagens e vídeos de conteúdo pornográfico infantil conhecidos pode auxiliar na identificação rápida dessas evidências em meio de outras centenas de milhares de arquivos encontrados no material examinado.

Uma deficiência no uso de *hashes* para tais propósitos está no fato de que qualquer modificação, seja ela de apenas um bit, modifica completamente o valor do *hash*. Isso dificulta o processo de localizar arquivos que semanticamente podem ser considerados iguais tais como:

- fotos digitais da mesma cena com taxas de compressão ou metadados EXIF diferentes;
- arquivos MP3 da mesma música, porém com metadados ID3 distintos;
- mensagens de e-mail examinadas do ponto de vista do remetente e do destinatário;
- páginas web com elementos dinâmicos como anúncios publicitários;
- documentos ou planilhas em que apenas os metadados, como por exemplo o tempo de edição, foram alterados.

Algumas pesquisas realizadas para superar essas limitações são apresentadas por Kornblum (2006) e Roussev et al. (2007). Na primeira abordagem, chamada de *Context Triggered Piecewise Hash* (CTPH) uma assinatura de *hash* é obtida pela combinação de técnicas

tradicionais de *hash*, porém considerando o contexto dos dados de entrada. Isso permite que essa assinatura de *hash* resultante identifique também versões modificadas de arquivos conhecidos. A segunda, chamada de *Multi-Resolution Similarity Hash* (MRS *hash*), tem o objetivo de calcular uma medida de similaridade de objetos mais primitivos da mídia de armazenamento. Isso ocorre ainda enquanto os dados são copiados do material original e permite a pesquisa de similaridade em vários níveis de granularidade sem a necessidade de indexação ou pré-processamento.

Pesquisa por palavras-chave

A pesquisa por palavras-chave é um dos métodos mais utilizados para identificar rapidamente arquivos de interesse. As ferramentas atuais são capazes de procurar ocorrências das palavras desejadas utilizando operadores simples para combiná-las e em várias codificações diferentes como ASCII ou Unicode. Algumas fornecem possibilidades adicionais como busca de sinônimos, derivações e variações de alguns caracteres da palavra original, embora muitos recursos não estejam disponíveis para a língua portuguesa. Também é possível realizar pesquisas utilizando expressões regulares.

Dois métodos principais são utilizados para realizar as buscas. Na busca exaustiva, uma pesquisa é feita em cada um dos setores ou arquivos da evidência examinada, do início ao fim dos dados, para cada pesquisa desejada. Na busca indexada, um índice é criado com todas as palavras encontradas na evidência e a posição das ocorrências. A pesquisa então é feita de forma instantânea por meio da consulta do índice.

No entanto, as ferramentas atuais apresentam limitações, sendo as principais:

- não realizam a busca com base no contexto, apenas com base na palavra-chave em si;
- a criação de um índice para a pesquisa indexada é muito demorada e não pode ser feita de forma distribuída;
- a pesquisa sem auxílio de um índice é lenta em dispositivos de grande capacidade.

Em Beebe and Clark (2007) é apresentada uma proposta de agrupamento temático dos resultados das pesquisas de palavras-chave utilizando um algoritmo baseado em Mapas de Kohonnen. Com a busca contextual não são exibidos somente os resultados que atendam a 100% dos critérios especificados, o que gera um grande número de ocorrências irrelevantes. Os mecanismos de busca na Internet já utilizam algoritmos de classificação dos resultados, apresentando os mais relevantes primeiro. Nas ferramentas forenses atuais não há essa possibilidade, pois não há qualquer forma de agrupamento ou ordenação dos resultados segundo sua relevância.

Com relação a propostas de melhoria no desempenho das pesquisas por palavras-chave, dois estudos interessantes são apresentados por Lee et al. (2008) e Roussev and Richard III (2004). Lee et al. (2008) apresentam uma abordagem baseada em *hardware*, que geralmente estão limitadas a aplicações em duplicação de discos rígidos e quebra de senhas. Segundo Lee et al. (2008), as ferramentas atuais de busca fazem pesquisa com velocidades médias de 20 MB/s, o que exigiria 14 horas para realizar uma busca em um disco rígido de um terabyte de capacidade de armazenamento. A utilização de uma placa Tarari[8] permite o desenvolvimento de programas que utilizem um agente *Tarari*

[8]Uma placa específica para processamento de conteúdo.

RegEx que fornece identificação e caracterização de conteúdo. Roussev and Richard III (2004) apresentam um protótipo de ferramenta utilizando uma arquitetura mestre-escravo para conduzir de forma distribuída as pesquisas por palavras-chave, obtendo reduções significativas no tempo necessário para realizá-las.

Análise temporal

A análise temporal consiste no exame de quaisquer dados temporais encontrados e sua correlação para a identificação de eventos significativos. Os dados temporais mais comuns são aqueles relacionados à criação, acesso e modificação de arquivos no sistema de arquivos. Esses podem servir para identificar hábitos de utilização do computador ou eventos como instalação de *softwares* ou atualizações do sistema operacional.

Outros dados temporais incluem as data e hora de envio e recebimento de mensagens de correio eletrônico, de geração de fotos digitais, de registros de *logs*, do registro do sistema (como a data de instalação e do último *boot*), de navegação em páginas da Internet, dentre vários outros.

Hosmer (1998) lembra que muitas vezes as evidências digitais serão confrontadas com evidências físicas, álibis e testemunhas. Portanto, essas evidências temporais podem ajudar a estabelecer a localização de indivíduos, a ocorrência de comunicação entre suspeitos, dentre outras indícios. Alguns complicadores nesse caso são eventos no computador que podem ser acionados remotamente ou que podem ser agendados, ou computadores móveis cuja interpretação incorreta de uma evidência temporal (como um erro de fuso horário, por exemplo) permitem que o suspeito alegue estar em outro lugar no momento do crime.

Um estudo apresentado em Buchholz and Tjaden (2007) discute como os *hosts* conectados à Internet gerenciam seus relógios. Um dos grandes desafios na utilização de vestígios temporais está em mostrar a acurácia dos dados coletados. Em Boyd and Forster (2004) é apresentado um estudo de caso em que um especialista forense foi acusado incorretamente de adulterar uma evidência devido à interpretação correta do assistente técnico da defesa. Quando existe a necessidade de correlação entre horários diferentes diversos fatores devem ser levados em consideração como relógios adiantados ou atrasados, fuso horário em cada localidade e existência de horários especiais (como horário de verão). Essas são apenas algumas das armadilhas que devem ser evitadas quando se lida com evidências temporais.

Stevens (2004) apresenta um modelo para unificar os diferentes intervalos de tempo encontradas em evidências temporais provenientes de fontes de tempo diferentes. Nesse caso, a própria imprecisão dos relógios existentes em cada local deve ser levada em consideração. Assim cada valor obtido deve receber, se necessário, uma correção, de forma que ao final do levantamento das evidências, todos os dados temporais estejam perfeitamente ajustados. A Figura 2.14 ilustra um exemplo de linha de tempo com dois eventos.

Dois outros trabalhos apresentam ferramentas para a reconstrução de eventos a partir de dados temporais presentes nas evidências examinadas. Khan et al. (2007) apresenta uma abordagem baseada em redes neurais artificiais utilizando as atividades do sistema de arquivos. Buchholz and Falk (2005) apresenta uma ferramenta gráfica que além de utilizar as atividades do sistema de arquivos, também examina *logs* de sistema e de *firewalls*, para construir uma linha do tempo com eventos de várias evidências. Os eventos relacionados entre si também podem ser agrupados em hierarquias.

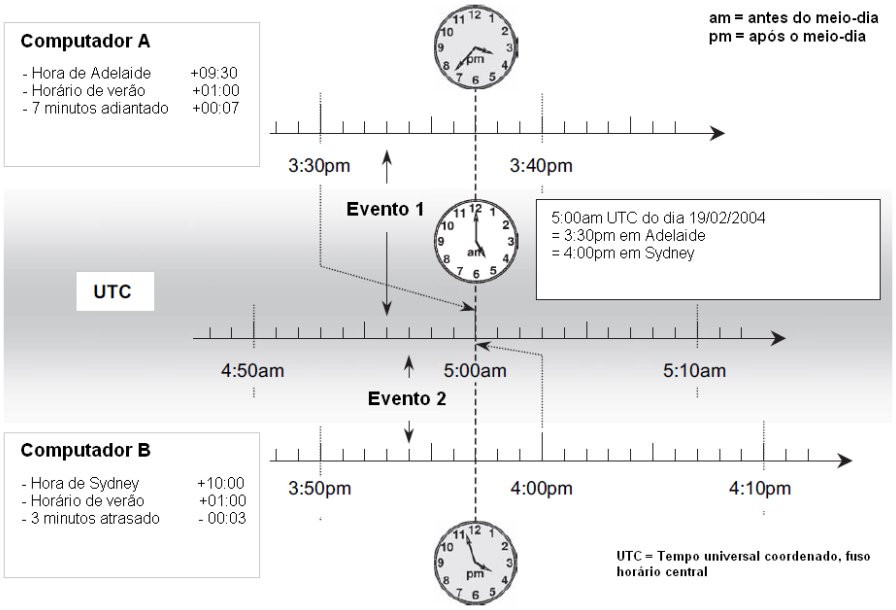

Figura 2.14: Análise de eventos em uma linha de tempo, adaptada de Stevens (2004)

Classificação de arquivos

A classificação de arquivos pode ser baseada em sua extensão ou de forma mais precisa com base no cabeçalho (alguns dos primeiros bytes) e rodapé (alguns dos bytes finais) desses. Essa classificação pode ser manipulada por meio da alteração da extensão ou de alguns bytes do cabeçalho e do rodapé, com o intuito de "disfarçar" o arquivo. A alteração da extensão é facilmente detectada pela verificação do cabeçalho, mas se ambos forem modificados, pode haver uma classificação incorreta ou nenhuma classificação é feita. A Figura 2.15 ilustra o procedimento de classificação e processamento de arquivos.

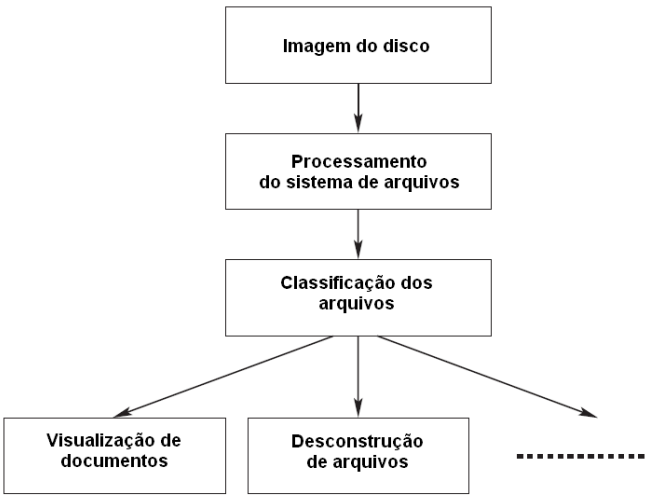

Figura 2.15: Processo de análise, adaptada de Vel (2004)

Pela figura, pode-se notar a importância da classificação corretado arquivo, pois ela

determina como o arquivo será visualizado e processado. Em arquivos contêineres como arquivos compactados (ZIP, RAR, etc.) é importante que a classificação seja correta para permitir interpretar e descompactar os arquivos neles contidos. Um bom mecanismo de classificação de arquivos também é muito útil na recuperação de arquivos apagados, pois permite determinar o tipo de arquivo de fragmentos de dados no espaço não alocado do disco.

Quanto mais detalhada a classificação, maiores são as possibilidade de agrupamento e filtragem de arquivos, simplificando o trabalho de análise dos dados. Devido à grande variedade de tipos de arquivo existentes, nem sempre as ferramentas conseguem classificar os arquivos corretamente, classificando-os como desconhecidos.

Vel (2004) apresenta uma abordagem que permite o aprendizado automático de classificação de arquivos baseados em suas subestruturas características de baixo nível. Um desempenho satisfatório foi obtido utilizando apenas os 256 primeiros bytes dos arquivos. Com a utilização de um número pequeno de bytes, o tempo de processamento e os requisitos de memória são baixos. Essa proposta pode, portanto, complementar a classificação já realizada pelas ferramentas atuais, nos casos em que os arquivos são classificados como desconhecidos.

Neste capítulo, além de uma visão geral de Informática Forense, foram apresentados os principais desafios enfrentados atualmente e diversos trabalhos correlatos. Diante desse cenário, propõe-se a utilização de SMA como uma forma inovadora de superar alguns desses desafios como o exame de grandes volumes de evidências e a viabilização do reuso de conhecimento adquirido de casos passados.

Capítulo 3

Sistemas Multiagente

Segundo Russell and Norvig (2003), o estudo da IA começou logo após a 2ª Guerra Mundial e o termo em si foi cunhado em 1956. As pesquisas em IA englobam um grande número de áreas e aplicações como robótica, planejamento automatizado, mineração de dados e suporte à decisão. De acordo com Weiss (1999), a IAD surgiu da metade para o fim da década de 1970, evoluindo e diversificando-se rapidamente. Ela é definida por Russell and Norvig (2003) como:

> o estudo, construção e aplicação de sistemas multiagente, ou seja, sistemas nos quais vários agentes inteligentes interagem em busca de um conjunto de objetivos ou da realização de um conjunto de tarefas.

Neste capítulo é apresentada uma revisão da teoria de Sistema Multiagente (SMA), uma área de estudo da IAD, com relação a organização, metodologias e padrões, protocolos de comunicação e ontologias. A Seção 3.5 é dedicada às questões de planejamento, enquanto a Seção 3.6 apresenta uma revisão de Raciocínio Baseado em Casos.

3.1 Agentes Inteligentes

Pela definição de Russell and Norvig (2003), um agente é qualquer elemento que possa ser visto percebendo seu ambiente por meio de sensores e atuando sobre ele por meio de atuadores. Um agente humano, por exemplo, percebe o ambiente utilizando seus sentidos e atua utilizando os membros do seu corpo. Um agente robótico pode utilizar câmeras de vídeo como sensores e um braço robótico como atuador. O mesmo pode ser imaginado para um agente de *software*.

Para Weiss (1999), um agente é uma entidade computacional, como um programa de *software* ou um robô, que além de perceber e atuar, o faz de maneira autônoma no sentido de que seu comportamento depende pelo menos parcialmente de sua própria experiência. A flexibilidade de comportamento e a racionalidade são alcançadas pelo agente por meio de processos chave, tais como resolução de problemas, planejamento, tomada de decisões e aprendizado. Kolp et al. (2006) enfatizam que os agentes inteligentes são por virtude de sua capacidades intencionais (habilidade de planejar e negociar) muito mais do que meros componentes de *software*.

Wooldridge (2002) afirma que não há definição universalmente aceita para o termo agente, embora haja consenso sobre a necessidade desse, por definição, ser autônomo.

Portanto, a definição oferecida em Wooldridge (2002) define um agente como "um sistema computacional que está situado em algum ambiente e que é capaz de ações autônomas nesse ambiente para que atinja seus objetivos".

A racionalidade do agente, segundo Russell and Norvig (2003), deve levar em consideração o que o agente pode fazer, sua percepção do ambiente (incluindo o que já foi percebido) e a medida de desempenho que avalia o seu sucesso. O agente ideal deve buscar maximizar seu desempenho com base nas informações da sua percepção e do conhecimento que possui. O projeto desse agente deve mapear as ações que ele tomará para qualquer sequência de percepção.

Assim, o projeto apropriado de um agente depende das suas percepções, ações, metas e do ambiente em que se encontra. O ambiente pode exigir mais dos agentes, dependendo de suas características como, por exemplo, o quão observável, determinístico, dinâmico e contínuo ele é. Na maioria dos domínios de complexidade razoável, um agente não terá completo controle sobre o ambiente (Wooldridge, 2002). Um agente também pode ter suas ações ou seu ambiente afetados por outros agentes, com os quais poderá interagir de maneira competitiva ou cooperativa para atingir seus objetivos.

Russell and Norvig (2003) apresentam uma classificação dos agentes inteligentes em quatro tipos básicos:

1. reativo simples: agem como reação à percepção atual, ignorando as percepções anteriores;
2. reativo baseado em modelos: mantém um estado interno com base no seu histórico de percepções e no estado do ambiente, mas ainda age de forma reativa;
3. baseado em objetivos: além das informações de estado, baseiam sua decisão atual nos seus objetivos, que descrevem situações que são desejáveis;
4. baseado na utilidade: além de considerar seus objetivos, considera também a utilidade de suas decisões e ações.

A Figura 3.1 apresenta o modelo de agente que mantém informações das suas percepções passadas e as considera na realização da próxima ação.

Segundo Sycara (1998), a capacidade de um agente inteligente é limitada pelo seu conhecimento, seus recursos computacionais e sua percepção. Logo, problemas complexos de maior escala estão acima das capacidades de um único agente inteligente. Para a resolução desses problemas, deve-se empregar vários agentes, que embora sozinhos não sejam capazes de resolver o problema, podem trabalhar conjuntamente para alcançar a solução.

Um Sistema Multiagente (SMA) é um sistema composto por diversos agentes inteligentes. Algumas das principais características dos SMA, segundo Sycara (1998), são a distribuição do controle, a descentralização dos dados e a comunicação assíncrona.

Viroli et al. (2007) fazem distinção entre duas classes principais de SMA:

- sistemas de resolução distribuída de problemas em que agentes-componentes são explicitamente projetados para cooperativamente atingir um objetivo;
- sistemas abertos em que agentes, não necessariamente co-projetados para compartilhar objetivos em comum, podem dinamicamente entrar e sair do sistema.

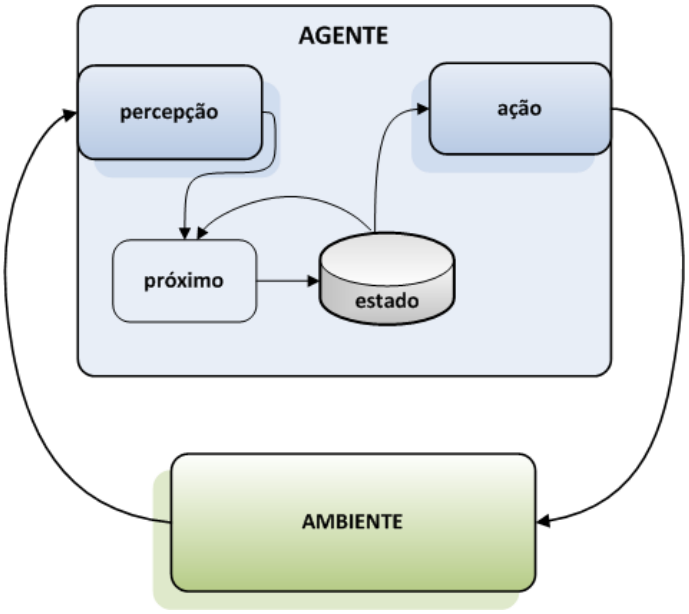

Figura 3.1: Modelo de agente que mantém informações de estado, adaptada de Wooldridge (2002)

No primeiro, todos os agentes são conhecidos *a priori* e todos os agentes supostamente são benevolentes uns com os outros e portanto, podem confiar um nos outros durante as interações. No segundo, a chegada dinâmica de agentes desconhecidos precisa ser levada em conta, bem como a possibilidade de comportamento auto-interessado no curso das interações.

Uma revisão conceitual dos aspectos de organização e comunicação em SMA é apresentada nas seções a seguir. Por fim, são apresentados alguns dos padrões estabelecidos pela *Foundation for Intelligent Physical Agents* (FIPA) e a *Java Agent DEvelopment Framework* (JADE), a plataforma de desenvolvimento de SMA utilizada neste trabalho.

3.2 Organização

A definição e especificação de um SMA requer um abordagem distinta da engenharia de software convencional. As metodologias de desenvolvimento devem levar em consideração a natureza dos agentes inteligentes e de suas interações. Um dos fatores chave a ser considerado é a organização dos agentes no SMA. A organização define a forma de interação entre os agentes e pode ser especificada de várias maneiras. Nesta seção, são apresentadas conceitos e propostas de organização.

Alguns SMA podem não apresentar uma organização definida. Nesse modelo os agentes são entidades individuais com seus próprios objetivos. O desempenho e comportamento global do sistema, nesse caso, são difíceis de prever. Vázquez-Salceda et al. (2005) enfatizam que em aplicações críticas como as utilizadas em hospitais ou na polícia, o comportamento global do sistema deve ser levado em conta e as características estruturais do domínio devem ser incorporadas. Isso provê algumas características ao SMA como estabilidade e previsibilidade organizacional e compromisso com objetivos e estratégias.

Segundo Horling and Lesser (2005), não há um tipo comum de organização adequado a todas as situações. Em alguns casos, um único estilo organizacional não é o suficiente e um conjunto de estruturas organizacionais distintas faz-se necessário. Alguns pesquisadores afirmam que não há organização perfeita para qualquer situação, devido às inevitáveis concessões (*tradeoffs*) que devem ser feitas e ao alto dinamismo do SMA.

Ainda assim, a definição de uma ou mais formas de organização para um SMA é essencial. Horling and Lesser (2005) afirmam que muitos pesquisadores demonstraram que o projeto organizacional aplicado a um SMA pode ter efeito significativo e quantitativo sobre a suas características de desempenho. A estrutura organizacional pode ser utilizada para limitar o escopo das interações entre os agentes, reduzir ou explicitamente aumentar a redundância de um sistema, formalizar objetivos de alto nível de que um agente isolado pode não ter ciência ou forçar certos mecanismos de coordenação para a execução eficiente de tarefas. Alguns exemplos de organização incluem hierarquias, coalizões, times, congregações, sociedades, federações, mercados e matrizes de agentes. A Figura 3.2 ilustra informalmente três desses exemplos.

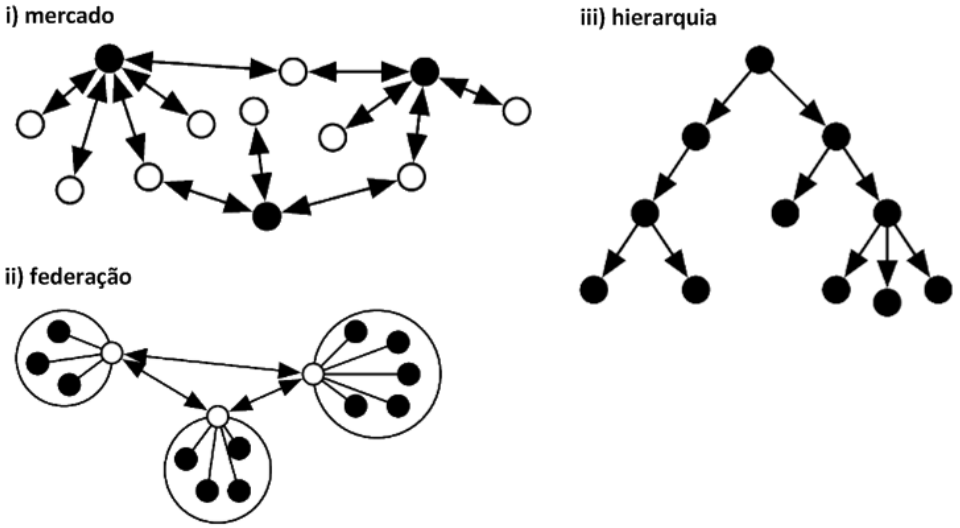

Figura 3.2: Ilustração de três tipos de organização, adaptada de Horling and Lesser (2005)

Cada uma das organizações mencionadas apresenta suas peculiaridades, vantagens e desvantagens. Mas em todas elas é importante notar a evidente relação com as organizações humanas, que são base para a maioria das propostas apresentadas a seguir. Viroli et al. (2007) defende que em um SMA, o comportamento autônomo e proativo dos seus agentes sugere que aplicações podem ser projetadas simulando o comportamento e a estrutura de organizações humanas.

3.2.1 Definição de organizações

De acordo com Viroli et al. (2007), uma perspectiva organizacional pode tornar o projeto do sistema menos complexo e mais fácil de gerenciar que metáforas mais tradicionais de sistemas concorrentes. Primeiramente, cada agente torna-se um local separado de controle, encarregado de cumprir seu papel e sendo totalmente responsável por ele. Em

segundo lugar, uma vez que agentes tipicamente englobam a maioria das funcionalidades que eles precisam para cumprir seu papel, a interdependência entre os componentes do sistema provavelmente será reduzida.

Para isso o conceito de papéis (*roles*) tem sido cada vez mais utilizado no projeto e especificação dos SMA. Em geral, o papel de um agente é simplesmente definido em termos de uma tarefa específica que o agente tem que realizar no contexto total da organização. Viroli et al. (2007) sugere uma noção de papel mais precisa, que dá ao agente uma posição bem definida na organização, com um conjunto de comportamentos esperados associados.

Vázquez-Salceda et al. (2005) propõe o conceito de estrutura social, que em uma organização descreve os papéis presentes, incluindo seus objetivos, direitos e requisitos, e possivelmente grupos e relações entre os papéis. Os papéis especificam as expectativas da sociedade com relação às atividades do agente na sociedade e por isso eles representam os principais elementos do conceito de estrutura social, que é composta pelos seguintes elementos:

- papéis – uma lista de definições dos papéis presentes na organização;
- hierarquia de papéis e dependências – um lista de triplas com dois papéis e o nome da relação entre eles;
- grupos – lista de conjuntos de papéis relacionados.

As descrições dos papéis devem identificar as atividades e serviços necessários para alcançar os objetivos de uma sociedade específica. Um papel é definido por seu nome, objetivos, subobjetivos, direitos, normas e regras que devem seguir, e o tipo de agente que pode assumir o papel. Os grupos fornecem um meio de referenciar coletivamente um conjunto de papéis. Os grupos são utilizados para especificar normas que se aplicam a todos os papéis em um grupo. Os elementos que definem um grupo são: nome do grupo, lista de papéis que compõem o grupo, normas e regras que se aplicam aos papéis (Vázquez-Salceda et al., 2005).

Na proposta de Viroli et al. (2007), outros três conceitos organizacionais que seriam necessários para complementar a especificação das organizações computacionais são introduzidos:

1. regras organizacionais – expressam requisitos gerais para a instanciação apropriada e execução do SMA;
2. estrutura organizacional – define a classe específica de organização e regime de controle aos quais os agentes/papéis têm que atender de forma que o SMA inteiro possa trabalhar de maneira eficiente e de acordo com os requisitos especificados;
3. padrões organizacionais – expressam estruturas organizacionais pré-definidas e amplamente utilizadas, que podem ser reutilizadas de sistema para sistema, como os catálogos de padrões utilizados no projeto de sistemas orientados a objetos.

Hannoun et al. (2000) propõe o *Model of Organization for multI-agent SystEms* (MOISE), um modelo de organização de SMA baseada em três conceitos: papéis (*roles*), que contêm o comportamento individual dos agentes; as relações organizacionais (*organizational links*) que regulam a interação social entre agentes; e grupos (*groups*) que aglutinam agentes que

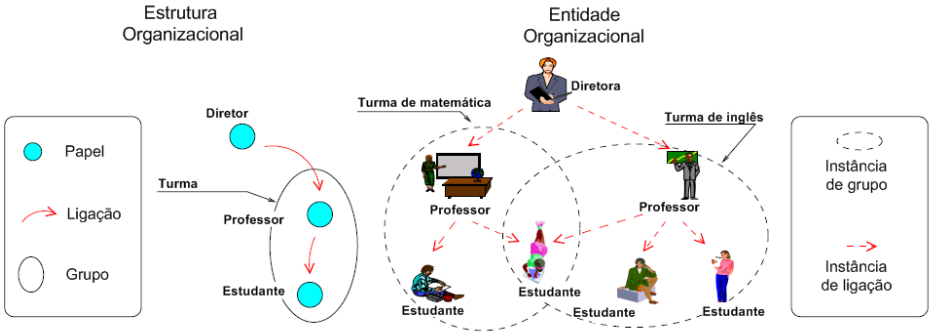

Figura 3.3: Ilustração do modelo de organização MOISE, adaptada de Hannoun et al. (2000)

interagem fortemente. A Figura 3.3 ilustra a proposta e o mapeamento de uma estrutura organização para uma entidade organizacional – uma instância da estrutura.

DeLoach et al. (2008) sugere a definição de *competências* como chave para determinar exatamente quais agentes podem desempenhar quais papéis dentro da organização. Uma competência (*capability*) seria uma entidades atômica utilizada para definir uma habilidade ou capacidade dos agentes. Ela pode ser usada para capturar uma habilidade como o acesso e o controle sobre recursos específicos, a habilidade de se comunicar com outros agentes, a habilidade de migrar para uma nova plataforma ou a habilidade de conduzir planos para alcançar objetivos específicos. Assim, um conjunto de competências em uma organização é a união de todas as competências requeridas pelos papéis ou possuídas pelos agentes da organização. A Equação 3.1, definida em DeLoach et al. (2008), apresenta essa definição, onde C é o conjunto de competências, A é o conjunto de agentes da organização e R é o conjunto de papéis.

$$\forall c : C(\exists a : A\ possui(a, c) > 0 \vee \exists r : R\ c \in requer(r)) \tag{3.1}$$

DeLoach et al. (2008) também define uma função rcf, $A \rightarrow [0..1]$, que descreve a habilidade de um agente realizar um papel específico. Quanto maior o valor de rcf, para um agente a do conjunto de agentes A da organização, maior é sua aptidão para desempenhar o papel. Esse valor é definido pelo usuário e computado em termos das competências exigidas para realizar o papel. Por fim, um conjunto de atribuição Φ, que consiste em conjunto de triplas agente-papel-objetivo (a, r, g), indica que um agente a foi designado para assumir o papel r para atingir um determinado objetivo g.

Em síntese, em uma organização computacional, para cada agente é atribuído um papel específico, isso é, uma tarefa ou responsabilidade bem definida no contexto global do sistema, que o agente tem que cumprir de maneira autônoma. Segundo Viroli et al. (2007), nesse caso as interações entre papéis não são só uma expressão de interdependência, elas também ajudam a caracterizar a posição do agente na organização.

Além de pesquisas centradas na forma de organização de um SMA em si, várias pesquisas enfatizam a necessidade de normas sociais, convenções ou leis que regulem o comportamento do agentes, especialmente em SMA abertos. Essas pesquisas visam a definição mais precisa dos papéis e das interações entre os agentes, garantindo a manutenção da

organização sem afetar drasticamente a autonomia dos agentes.

Pesquisas também têm sido realizadas para ampliar a capacidade de reorganização e adaptação nos SMA. Em geral, a reorganização ou adaptação é necessária quando um evento ocorre que muda o estado da organização atual, como mudanças nos objetivos e mudanças nos agentes. DeLoach et al. (2008) lembra que em aplicações dinâmicas em que o ambiente ou os agentes podem passar por modificação, um projetista dificilmente poderá considerar todas as situações possíveis.

O foco da pesquisa apresentada em Martin and Barber (2006) é dar aos agentes a capacidade de se adaptar a situações de mudança na organização do sistema por meio de *frameworks* adaptativas para tomada de decisão ou *Adaptive Decision-Making Frameworks* (ADMF). Isso permite aos agentes superar falhas e melhorar o desempenho modificando o padrão de controle de informação e relação de comunicação entre os agentes, bem como a distribuição de tarefas, recursos e competências. Segundo os autores, utilizando o ADMF, os agentes poderiam, por exemplo, superar a falha de um agente, reestruturando colaborativamente o processo de tomada de decisão ou contornar situações de baixo desempenho, permitindo aos agentes estabelecer novas colaborações que funcionem melhor.

DeLoach et al. (2008) define um modelo organizacional para sistemas computacionais adaptativos ou *Model for Adaptive Computational Systems* (OMACS). Nele, definem-se os requisitos de conhecimento da estrutura organizacional de um sistema e as capacidades que permitirão ao sistema reorganizar-se em tempo de execução e alcançar seus objetivos em face de mudanças no ambiente e nas capacidades dos agentes.

3.2.2 A organização hierárquica

A hierarquia ou organização hierárquica é talvez o exemplo mais antigo de projeto organizacional estruturado aplicado a SMA e arquiteturas distribuídas de IA (Horling and Lesser, 2005). Nesse caso, agentes são conceitualmente organizados em uma estrutura em forma de árvore, onde agentes que estão mais alto na árvore têm uma visão mais global que aqueles abaixo deles. Em uma interpretação mais estrita, interações não acontecem ao longo da árvore, mas apenas entre as entidades conectadas.

A instância mais simples dessa estrutura consiste em uma hierarquia de dois níveis, na qual os agentes do nível inferior são completamente controlados pelo nível superior, que produz uma visão global da informação resultante. Instâncias mais complexas têm múltiplos níveis e enquanto os dados fluem, as relações de autoridade e outras características ditadas pela organização podem não ser absolutas.

Segundo Horling and Lesser (2005), em uma hierarquia as decisões são tomadas por agentes que têm a informação necessária para raciocinar sobre a decisão e a autoridade organizacional para tomar a decisão. Cada nível atua como um filtro, explicitamente transferindo informações e implicitamente transferindo decisões para o topo da hierarquia, somente quando necessário. Kolp et al. (2006) lembra que em uma estrutura hierárquica, gerentes e supervisores nos níveis intermediários podem coordenar comportamentos ou tomar decisões táticas próprias, mas apenas no nível local.

A eficiência da hierarquia é derivada dessa noção de decomposição, pois a abordagem dividir-e-conquistar que ela gera permite ao sistema utilizar grupos maiores de agentes mais eficientemente e tratar de problemas de maior escala (Horling and Lesser, 2005). Esse

tipo de organização pode restringir agentes a um número de interações que é pequeno com relação à população total, permitindo o aumento do paralelismo e maior tratabilidade de ações de controle local e de tomada de decisões. Porque há menos dados potencialmente distrativos, agentes podem obter uma visão mais clara da informação pertinente a essas decisões.

Segundo Vázquez-Salceda et al. (2005), estruturas hierárquicas são adequadas para ambientes em que o propósito da sociedade seja a produção eficiente de alguns tipos de resultados ou bens ou o controle de um sistema de produção externo. Nesses ambientes, um controle confiável dos recursos e do fluxo de informação requer entidades centrais gerenciadoras, que também precisam de acesso rápido à visão global. Vázquez-Salceda et al. (2005) propõem a adição de dois papéis facilitadores: controladores, que monitoram e orientam o desempenho global do sistema ou de uma parte dele, e agentes de interface responsáveis pela comunicação entre o sistema e o mundo exterior.

Horling and Lesser (2005) alertam para o fato de que utilizar uma hierarquia pode levar a uma organização rígida e frágil, propensa a apresentar pontos únicos de falha com consequências potencialmente globais. Por exemplo, se o agente no topo da hierarquia falhar, a coesão de toda a estrutura pode ser comprometida. O agente pode ser substituído, mas pode custar muito para recuperar a informação concentrada possuída por seu predecessor.

Uma hierarquia também se torna suscetível a efeitos de gargalo, caso o escopo das decisões de controle ou o recebimento de dados não for eficientemente gerenciado. Considere, por exemplo, o que ocorreria se o agente no topo recebesse todos os dados produzidos por todos os agentes abaixo dele. Conforme citado por Horling and Lesser (2005), um solução para isso é aumentar as ligações internas na árvore para permitir comunicação mais direta, o que pode reduzir a latência das informações resultante do percorrimento de toda a estrutura.

Assim como na maioria da estruturas organizacionais, o formato da hierarquia pode afetar as características dos comportamentos globais e locais. Um estrutura muito plana onde os agentes têm um alto nível de conectividade pode levar a sobrecarga se os recursos forem limitados e consumidos como resultado dessas conexões. Reciprocamente, uma estrutura muito profunda pode reduzir a velocidade do sistema devido à atrasos gerados pela passagem de informação através dos múltiplos níveis.

Vázquez-Salceda et al. (2005) apresentam uma comparação de três modelos de organização com relação às características das interações entre agentes em cada um deles, conforme representado na Tabela 3.1. Note-se que nas organizações hierárquicas, o aspecto de coordenação é exercido por supervisão, isto é, os agentes estão envolvidos em relações dependentes de poder e agem de acordo com rotinas. Os agentes são em geral completamente cooperativos e a coordenação é alcançada por linhas de comando e controle bem definidas. As redes de agentes alcançam coordenação pelo interesse mútuo e a interdependência. Agentes mesmo quando auto-interessados, concordam em colaborar para atingir um objetivo mútuo que beneficie todos. A organização de mercado, por sua vez, apresenta o auto-interesse como característica principal, o que leva a uma relação predominante de competição, regulada e coordenada pelo próprio mercado.

Em Pinson and Moraïtis (1996), um SMA organizado hierarquicamente é aplicado ao problema de tomada de decisão. Uma das questões levantadas é sobre como obter coerência e coordenação entre decisões feitas por agentes posicionados em níveis hierárquicos

Tabela 3.1: Comparação de três modelos de organização, adaptada de Vázquez-Salceda et al. (2005)

	Mercado	**Rede**	**Hierarquia**
Tipo de sociedade	Aberta	Confiança	Fechada
Valores	Auto-interesse	Interesse mútuo	Dependência
Coordenação	Preço	Colaboração	Supervisão
Forma de relação	Competição	Interesse mútuo	Autoridade
Relação de dependência	Oferta	Pedido	Delegação
Resolução de conflitos	Negociação ou Arbitragem	Reciprocidade (Reputação)	Supervisão

diferentes. Os autores apresentam uma implementação baseada em *blackboards* para auxiliar gerentes a avaliar a viabilidade e coerência de um plano de ação. A proposta Pinson and Moraïtis (1996) inclui a divisão dos agentes em três níveis hierárquicos. O nível estratégico, o mais alto, decompõe objetivos globais em subobjetivos e coordena a alocação destes para o segundo nível, chamado de centro de decisão. Cada centro de decisão prepara propostas de ação para cada subobjetivo. Essas propostas serão executadas no último nível, o nível especialista, que propõe uma ou mais ações elementares para cada proposta. Nesse nível, cada especialista tem um domínio próprio de competência.

Com relação à utilização de agentes especilizados, Zoethout et al. (2008) apresentam um estudo comparativo do desempenho de agentes generalistas e especialistas em um ambiente auto-organizável, ou seja, os agentes têm liberdade de trabalhar nas tarefas que desejarem segundo sua especialidade ou motivação. Os autores oferecem várias conclusões interessantes:

- em geral o desempenho global é melhor em um ambiente no qual a variedade de tarefas é baixo;
- em casos em que não há variedade de tarefas, os agentes especialistas têm melhor desempenho que os agentes generalistas;
- em um ambiente de grande variedade de tarefas, não há oportunidade para o desenvolvimento de especialidades ou ganho motivacional, o que faz com que generalistas e especialistas tenham desempenho similar.

A organização hierárquica é utilizada na proposta deste trabalho por ser similar à estrutura comumente adotada pelas organizações policiais, nas quais há uma cadeia de comando bem definida em termos hierárquicos. Com isso, a definição de papéis do agentes apresenta grande similaridade aos papéis desempenhados pelos agentes humanos, o que facilita também a integração do SMA ao processo operacional da organização humana. A organização do sistema é apresentada em detalhes no Capítulo 4. Além disso, devido à natureza da aplicação do SMA proposto requisitos como previsibilidade organizacional e compromisso com objetivos são de extrema importância e são característicos da organização hierárquica.

3.2.3 A metodologia PASSI

A engenharia de SMA requer metodologias adequadas à natureza dos agentes inteligentes e de suas interações. Henderson-Sellers and Giorgini (2005) apresenta uma coleção e comparação de diversas metodologias como Tropos, Gaia, Prometheus, *Process for Agent Societies Specification and Implementation* (PASSI) e *Multi-agent Systems Engineering* (MaSE). A Figura 3.4 apresenta a ramificação dessas metodologias a partir do paradigma da orientação a objetos. Com base na avaliação dessas metodologias, decidiu-se pela utilização da metodologia PASSI para o desenvolvimento do SMA proposto neste trabalho.

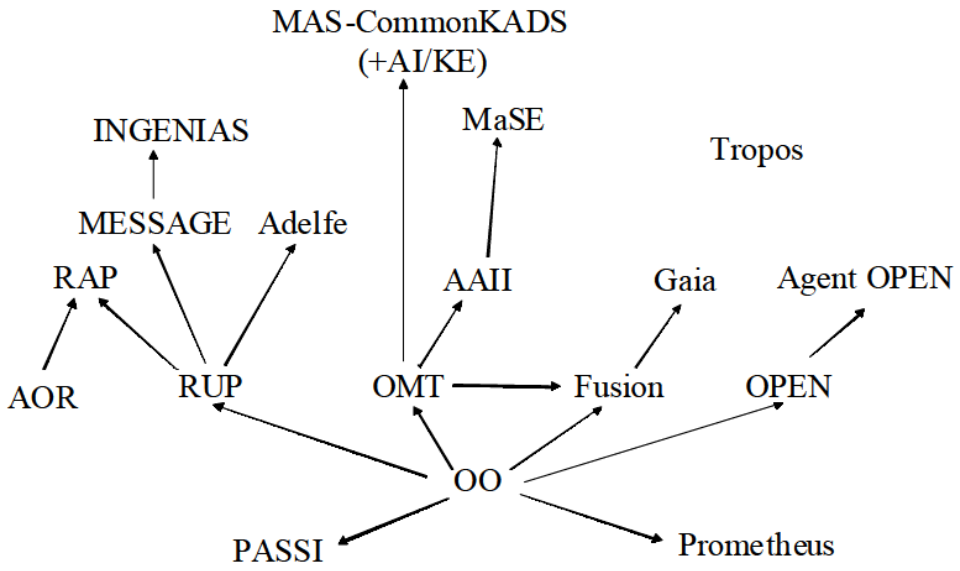

Figura 3.4: Metodologias de engenharia de software orientada a agentes (Henderson-Sellers and Giorgini, 2005)

O PASSI é uma das poucas metodologias que apresenta passos desde o levantamento de requisitos até a criação do código para o desenvolvimento de software multiagente (Cossentino and Potts, 2002). PASSI integra modelos de projeto e filosofia da engenharia de software orientada a objetos e de sistemas multiagente utilizando notação da *Unified Modelling Language* (UML). Outra vantagem apresentada é a adequação aos padrões da FIPA, que facilitam a transição do projeto para o código, especialmente quando utilizada a plataforma JADE, que também adere a esses padrões.

Segundo DeLoach et al. (2008), uma classe de agente é um modelo para um tipo de agente em um sistema e é análogo a uma classe de objeto na orientação a objetos. Um agente é uma instância da classe de agente. As classes de agente são definidas em termos dos papéis que desempenharão e das conversações das quais participarão.

No PASSI, um agente é uma unidade significativa do *software* tanto no nível do projeto quanto do código. Segundo essa visão, um agente é uma instância de uma classe de agente que é uma implementação de *software* de uma entidade autônoma capaz de buscar objetivos por meio de decisões, ações e relações sociais autônomas. Cossentino and Potts

(2002) afirma que seria mais natural descrever agentes em uma linguagem psicológica e social.

Um agente pode ocupar diversos papéis funcionais durante as interações com outros agentes para atingir seus objetivos, onde um papel é uma coleção de tarefas realizadas por um agente na busca de um subobjetivo. Uma tarefa, por sua vez, é definida como uma unidade significativa de comportamento individual ou interativo.

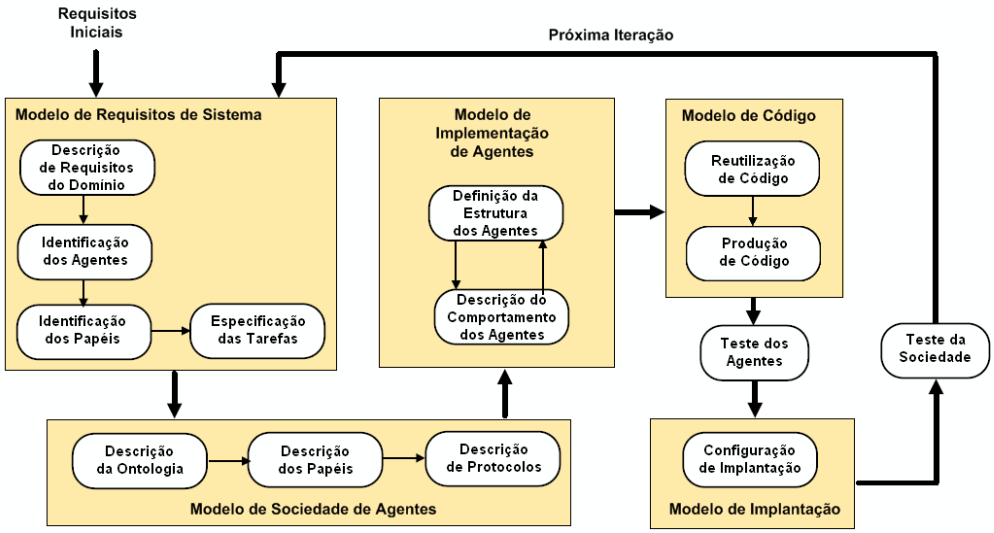

Figura 3.5: O método PASSI, adaptada de Henderson-Sellers and Giorgini (2005)

Os modelos e fases definidos no PASSI, apresentados na Figura 3.5, são:

1. modelo de requisito de sistema, composto de:
   - descrição do domínio: uma descrição funcional do sistema utilizando diagramas de casos de uso;
   - identificação dos agentes: separação de responsabilidades em agentes, representados por pacotes UML com uso de estereótipos;
   - identificação dos papéis: uso de diagramas de sequência para explorar as responsabilidades de cada agentes em cenários específicos de cada papel;
   - especificação de tarefas: especificação por meio de um diagrama de caso de uso e especificação auxiliar das capacidades de cada agente;
2. modelo de sociedade de agentes: um modelo das interações sociais e dependências entre agentes envolvidos na solução, que compreende três passos:
   - descrição da ontologia do domínio: use de diagrama de classes e restrições em *Object Constraint Language* (OCL) para descrever o conhecimento atribuído aos agentes individuais e a pragmática das suas interações;
   - descrição dos papéis: uso de diagramas de classe para representar papéis distintos exercidos pelos agentes, as tarefas envolvidas nesses papéis e capacidade de comunicação e dependência entre agentes;

- descrição de protocolos: uso de diagramas de sequência para especificar a gramática de cada protocolo de comunicação em termos de atos de comunicação.

3. modelo de implementação do agente: um modelo da solução de arquitetura em termos de classes e métodos, que requer:
   - definição da estrutura do agente utilizando diagramas de classe convencionais;
   - descrição do comportamento do agente utilizando diagramas de atividade ou de estado para descrever o comportamento de agentes individualmente;
4. modelo de código: um modelo da solução em nível de código;
5. modelo de implantação: modelo da distribuição das partes do sistema.

O PASSI é iterativo e possui dois tipos de iteração. O primeiro envolve todos os modelos do sistema e direcionada por novos requisitos. O segundo envolve apenas o modelo de implementação do agente. A Figura 3.6 apresenta esse segundo tipo de iteração, que deve ser considerado do ponto de vista multiagente e do agente isolado. As setas tracejadas representam as interdependências entre esses pontos de vista.

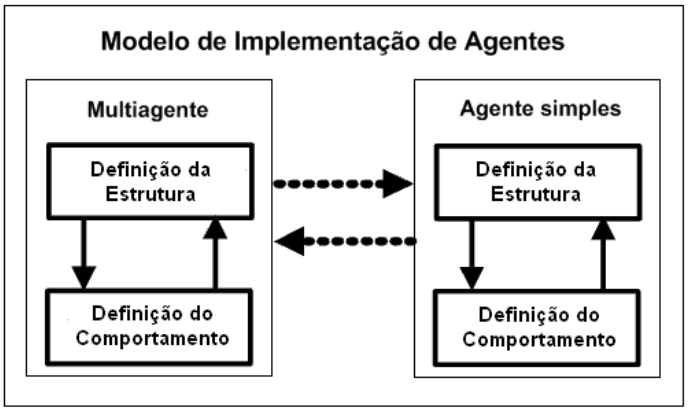

Figura 3.6: Modelo de implementação interativo de agentes, adaptada de Cossentino and Potts (2002)

Uma das características que destacam o PASSI é a definição de um descrição ontológica do sistema, que permite uma especificação mais apropriada das conversações entre os agentes. O método, no entanto, não define uma metodologia mais apurada para a definição dessa ontologia. Para tal, pode-se recorrer a metodologias específicas, conforme apresentado na Seção 3.3.

3.2.4 Comunicação

Russell and Norvig (2003) definem comunicação como troca intencional de informação realizado pela produção e percepção de um conjunto de sinais compartilhados. Um ato comunicativo (*speech act*) é uma ação do agente de produzir esses sinais.

Em um SMA, interagir com os demais agentes é essencial na realização dos objetivos. A comunicação nesse caso é crucial para conseguir que outro agente colabore na realização de uma tarefa, uma vez que os agentes são autônomos.

No início da década de 1990, foram definidas a *Knowledge Query and Manipulation Language* (KQML), uma linguagem para consulta e manipulação de conhecimento e um formato intercambiável de conhecimento, denominado *Knowledge Interchange Format* (KIF).

Um exemplo de diálogo utilizando KQML e KIF, apresentado em Wooldridge (2002), pode ser visto a seguir. Nele, o agente A envia uma mensagem para o agente B, utilizando a linguagem KIF e uma ontologia de nome `motors`. A mensagem é uma consulta, cujo identificador é `q1`, pedindo o valor do torque de `m1`. A resposta do agente B utiliza a mesma linguagem e ontologia e responde com uma expressão que diz que o torque de `m1` é igual a 12 kgf.

```
(evaluate
  :sender A
  :receiver B
  :language KIF
  :ontology motors
  :reply-with q1
  :content (val (torque m1)))
(reply
  :sender B
  :receiver A
  :language KIF
  :ontology motors
  :in-reply-to q1
  :content (= (torque m1) (scalar 12 kgf)))
```

Posteriormente, a FIPA propôs um padrão para a comunicação entre agentes, denominado *FIPA Agent Communication Language* (FIPA-ACL), que é semelhante ao KQML (Foundation for Intelligent Physical Agents (FIPA), 2002). A semântica, no entanto, foi especificada em maior detalhe, utilizando-se uma linguagem formal chamada *FIPA Semantic Language* (SL). Essa linguagem permite representar crenças, desejos e intenções dos agentes, bem como as ações que os agentes realizam em termos da linguagem SL. A semântica do padrão FIPA-ACL mapeia cada mensagem ACL à uma fórmula da SL, que define restrições que o remetente de uma mensagem deve satisfazer se quiser estar em conformidade com o padrão. Esse padrão é discutido posteriormente na Seção 3.4 e é utilizado na proposta deste trabalho na comunicação entre os agentes.

3.2.5 A Arquitetura *Blackboard*

Segundo van Liere et al. (1998), arquiteturas *blackboard* têm sido amplamente utilizadas na comunidade de IA como um tipo particular de modelo de resolução de problemas, pois permitem que vários agentes independentes, chamados de fontes de conhecimento (FC), compartilhem informações em uma área central denominada *blackboard*.

Para Jagannathan et al. (1989), o paradigma *blackboard* pode ser pensado como um modelo de resolução de problemas ou um *framework* para integrar vários métodos de

raciocínio em um único sistema. Ele é baseado na ideia de um grupo de especialistas independentes e cooperativos em volta de um quadro negro (*blackboard*) que contém uma solução em evolução para um dado problema. Cada especialista pode contribuir em algum aspecto para a solução do problema, mas não há uma ordem *a priori* para as contribuições dos especialistas.

Em outras palavras, os especialistas individuais devem aguardar até que dados ou eventos no *blackboard* forneçam uma indicação de que seu conhecimento pode contribuir para a solução do problema. Nesse momento, os especialistas relevantes podem colocar suas entradas no *blackboard*, permitindo que os outros especialistas se tornem ativos. Quando não houver mais nenhum especialista no grupo que possa se tornar ativo, o *blackboard* conterá então a solução final para o problema.

Para van Liere et al. (1998), o *blackboard* é como um esquema para organizar passos de raciocínio e conhecimento do domínio para construir uma solução para um problema particular. Na solução de um problema, cada fonte de conhecimento contribui com seu conhecimento específico para atingir o objetivo. O conhecimento é segmentado em módulos e um mecanismo de inferência separado é fornecido para cada módulo. A comunicação entre os módulos é feita lendo ou escrevendo no *blackboard*. Assim, o *blackboard* pode ser particionado de forma que contenha regiões com estruturas de dados diferentes, mas relacionadas. Dessa forma, aplicações podem organizar o espaço da solução em uma ou mais hierarquias dependentes da aplicação.

Corkill (2003) define os três principais componentes de um sistema *blackboard*, conforme ilustrado na 3.7:

- fontes de conhecimento (FC): são módulos computacionais independentes que juntos possuem o conhecimento necessário para resolver o problema;
- *blackboard*: o repositório compartilhado de dados que contém dados de entrada, soluções parciais, sugestões e outras informações úteis a solução do problema;
- componente de controle: gerencia o fluxo de execução para solução do problema, determinando qual FC é a mais apropriada para executar em um dado momento.

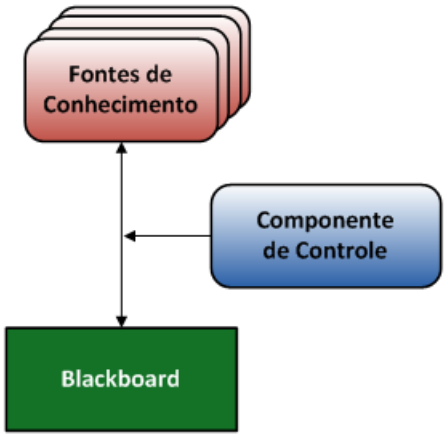

Figura 3.7: Principais componentes de um sistema *blackboard*, adaptada de Corkill (2003)

Jagannathan et al. (1989) afirma que não há nenhuma suposição fundamental sobre a estrutura interna da fonte de conhecimento, nem há qualquer suposição específica sobre o

componente de controle do *blackboard*. Corkill (2003) diz que as FC podem ser totalmente diversas em sua representação interna e forma de raciocínio e que são anônimas no sentido de não conhecerem as demais FC. Para van Liere et al. (1998), o modelo *blackboard* não especifica explicitamente o componente de controle. Ele meramente especifica um comportamento geral para resolução de problemas. O local de controle pode estar nas FC, no próprio *blackboard*, em um módulo separado ou em uma combinação desses.

Segundo van Liere et al. (1998), o objeto básico do *blackboard* é a variável, que encapsula toda a informação necessária para acessar um objeto do *blackboard*. Variáveis são definidas por quatro componentes: nome, tipo, descritor e dados ou lista de atributos. Nomes identificam unicamente a variável. O descritor da variável determina o tipo, tamanho e formato dos dados. O componente de dados é o contêiner de armazenamento dos dados puros. Os atributos são pares nome-valor que podem ser utilizados para descrever metadados da variável.

Buteau (1990) diz que as abordagens baseadas em *blackboard* têm sido utilizadas pois simplificam tanto o controle quanto a interação dos componentes de resolução de problemas, as FC. Ainda segundo o autor, o sucesso do paradigma básico tem levado ao desenvolvimento de sistemas *blackboard* independentes de domínio e também em ambientes distribuídos.

Nesses ambientes distribuídos, ao resolver um problema de domínio, um sistema de IAD realiza concorrentemente muitas ações de resolução de problemas. Cada ação local independente é ativada por dados, previamente gerados por elementos de solução locais ou elementos de solução previamente distribuídos e externamente gerados. Cada ação aplica alguma fonte de conhecimento de um domínio de problema, gerando ou modificando elementos da solução e distribuindo aqueles elementos conforme necessário (Buteau, 1990).

Buteau (1990) sugere a *Cooperative Architecture of Independent Blackboards* (CAIBL), uma arquitetura cooperativa de *blackboards* independentes, cujo foco primário é servir como um *framework* flexível para sistemas especialistas distribuídos trabalhando concorrentemente sobre vários problemas que requeiram controle dinâmico da comunicação de decisões.

3.3 Ontologia

Segundo Gruber (1993), uma ontologia é uma especificação explícita de uma conceitualização. Essa conceitualização é uma visão abstrata e simplificada do mundo que deseja-se representar. A ontologia tem então o papel de capturar o conhecimento de um domínio e fornecer uma compreensão comumente aceita dele (Staab and Maedche, 2000). Na prática, em um SMA uma ontologia define um vocabulário em comum que pode ser utilizado na comunicação entre os agentes (Wooldridge, 2002).

Segundo Noy and McGuinness (2001), uma ontologia cria um definição formal comum em um domínio de informação de uma certa área. Dessa forma, estruturas comuns de informação podem ser formadas, conhecimento pode ser reutilizado e suposições sobre um domínio podem ser feitas. A ontologia é, portanto, uma descrição explícita e formal dos conceitos de um domínio (chamadas também de classes), de propriedades descrevendo várias características e atributos de cada conceito (também chamadas de papéis ou *slots*) e restrições dessas propriedades (também chamadas de facetas ou *facets*). Uma

ontologia juntamente com um conjunto de instâncias individuais constitui uma base de conhecimento.

A visão proposta por Stumme et al. (2003) apresenta uma formalização do conceito de ontologia. Nela, uma ontologia O é uma estrutura da forma:

$$O := (C, \leq_C, R, \sigma, \leq_R) \tag{3.2}$$

que consiste de dois conjuntos disjuntos C e R, ou seja, $C \cap R = \emptyset$, que contêm respectivamente os conceitos e as relações; uma ordem parcial $\leq_C$, chamada de hierarquia de conceitos ou taxonomia; a função $\sigma : R \to C^+$, chamada de assinatura e uma ordem parcial $\leq_R$ chamada de hierarquia de relações.

Uma base de conhecimento KB, segundo essa visão de ontologia, contém asserções sobre instâncias de conceitos e suas relações e é da forma:

$$KB := (C_{KB}, R_{KB}, I, \iota_C, \iota_R) \tag{3.3}$$

Logo, KB consiste de um conjunto de conceitos (C_{KB}) e relações (R_{KB}) utilizados na base de conhecimento, no conjunto I de instâncias e nas funções de instanciação de conceitos (ι_C) e relações (ι_R). Algumas condições necessárias para que uma base de conhecimento seja consistente são a consistência da própria ontologia em que se baseia e que os conceitos e relações presentes na base de conhecimento estejam presentes na ontologia ($C_{KB} \subseteq C$ e $R_{KB} \subseteq R$).

Gruber (1993) destaca que as ontologias são tipicamente especificadas em linguagens que permitem abstração além de estruturas de dados e estratégias de implementação. Na prática, as linguagens de ontologias estão mais próxima em poder expressivo da lógica de primeira ordem do que das linguagens utilizadas para modelas bases de dados. Por isso, é dito que as ontologias encontram-se no nível semântico, enquanto os modelos de bases de dados estão no nível lógico ou físico.

3.3.1 Metodologias

Segundo Dileo et al. (2002), as metodologias existentes para projetar ontologias do domínio buscam descrever tudo sobre um domínio específico. No entanto, os autores apontam que isso não é apropriado para SMA, pois a ontologia do sistema deve especificar apenas a informação necessária para a execução apropriada do sistema.

Os agentes interagem por meio da troca de mensagens e essas mensagens frequentemente envolvem a passagem de parâmetros. Embora a comunicação possa ocorrer com a passagem de mensagens simples, em um sistema complexo é importante determinar a natureza dos parâmetros segundo informações do domínio, ou seja, determinar quais conceitos correspondem às mensagens e seus parâmetros.

Dileo et al. (2002) enfatiza que assim como é importante especificar um modelo de dados no processo de desenvolvimento de software tradicional, em um MAS esse modelo também deve ser especificado, porém considerando ainda os conceitos da ontologia do sistema. Em sua proposta, associada à metodologia MaSE de engenharia de SMA, a ontologia para o sistema é concebida em um passo adicional durante a fase de análise. Isso permite o uso de termos dos casos de usos e diagramas de sequência como possíveis

conceitos da ontologia. Assim, os conceitos da ontologia podem ser utilizados na definição de papéis e tarefas.

Dileo et al. (2002) lembra que ao reutilizar parte de um SMA, deve-se garantir que a ontologia utilizada anteriormente não tenha conflitos com a ontologia aplicada no novo sistema.

Para construir uma ontologia, o projetista precisa primeiro determinar o propósito e o escopo da ontologia e então coletar e analisar dados do domínio de informação para uso possível na ontologia. Finalmente, o analista constrói a ontologia inicial e refina, valida e amadurece o modelo até obter uma ontologia completa.

Segundo Noy and McGuinness (2001), não há uma forma "correta" ou metodologia para o desenvolvimento de ontologias. Em sua proposta, definem uma sequência de sete passos, listados a seguir:

1. determinar o domínio e o escopo da ontologia;
2. considerar a reutilização de ontologias existentes;
3. enumerar termos importantes na ontologia;
4. definir as classes e a hierarquia de classes;
5. definir as propriedades das classes (*slots*) e
6. definir as facetas dos *slots* como cardinalidades, restrições de tipos e valores.

Após a construção da ontologia do sistema, a metodologia de projeto de SMA deve permitir ao analista especificar objetos do modelo de dados como parâmetros em conversações entre os agentes, ou seja, explicitar os conceitos utilizados nas iterações em termos dos conceitos da ontologia.

3.3.2 Ferramentas

Uma linguagem muito utilizada atualmente para a definição de ontologias é a *Web Ontology Language* (OWL), recomendada pelo *World Wide Web Consortium* (W3C). Diversas ferramentas e APIs com suporte a OWL estão disponíveis para o desenvolvimento e manutenção de ontologias. Alguns exemplos são Protégé (`http://protege.stanford.edu`), *Karlsruhe Ontology Management Infrastructure* – KAON (`http://kaon.semanticweb.org`), OntoTrack (`http://www.informatik.uni-ulm.de/ki/ontotrack/`) e Jena (`http://jena.sourceforge.net/`). A Figura 3.8 apresenta a interface da ferramenta Protégé, utilizada neste trabalho.

Conforme apresentado na Seção 2.3, existem alguns poucos trabalhos relacionados à definição de ontologias no domínio das perícias de Informática, alguns com focos bastante distintos. Dessa forma, procurou-se neste trabalho definir uma ontologia de sistema, definindo-se apenas os conceitos necessários ao funcionamento do SMA no nível operacional, permitindo o uso adequado da informação trocada pelos agentes desse nível. Um maior detalhamento do uso da ontologia neste trabalho pode ser visto no Capítulo 4.

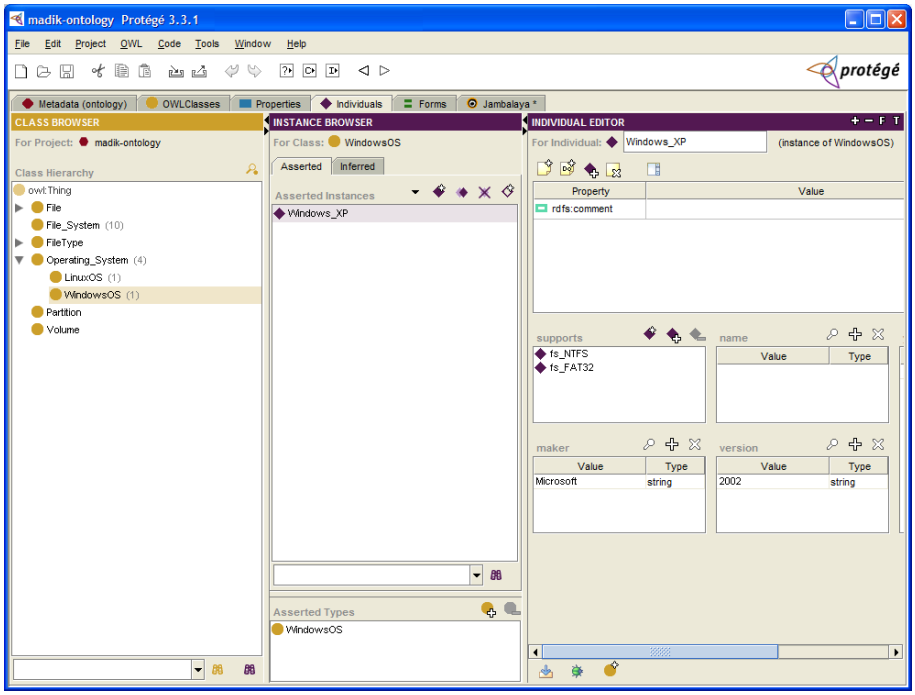

Figura 3.8: Inteface da ferramenta Protégé

3.4 Padrões FIPA

A FIPA especifica padrões para o desenvolvimento de agentes inteligentes, destinados a facilitar a adoção e interoperabilidade de tecnologias e sistemas baseados em agentes. Os padrões FIPA especificam os agentes comumente envolvidos em um SMA, protocolos e linguagens para a comunicação e interação entre agentes, ontologias para a gerência do sistema, dentre outros elementos.

A *FIPA Agent Management Specification* (Foundation for Intelligent Physical Agents (FIPA), 2002) define a arquitetura de uma plataforma que provê a infraestrutura básica para criação, comunicação, migração, registro e administração dos agentes. Para isso, define os seguintes componentes, ilustrados pela Figura 3.9:

- Agente: principal elemento da plataforma de agentes; ele utiliza os serviços da plataforma e pode acessar software ou usuários externos; cada agente deve possuir um proprietário e alguma forma de identidade, como o *Agent IDentifier* (AID), que identifica unicamente um agente.
- Facilitador de Diretórios ou *Directory Facilitator* (DF): componente obrigatório da plataforma de agentes segundo a especificação da FIPA; ele fornece a lista de serviços oferecidos no sistema por outros agentes, também chamado de páginas amarelas em alusão às páginas amarelas de uma lista telefônica.
- Sistema Gerenciador de Agentes ou *Agent Management System* (AMS): componente obrigatório e de forma semelhante ao DF fornece uma listagem dos agentes e seus endereços, chamada também de "páginas brancas". Neste sistema os agentes devem se registrar com o AMS para receber um AID válido. Além disso, o AMS exerce

um controle sobre o acesso e uso da plataforma, podendo haver apenas um AMS em uma plataforma de agentes.

- Sistema de Transporte de Mensagens ou *Message Transport System* (MTS): define o método padrão de comunicação entre agente de diferentes plataformas.
- Plataforma de Agentes ou *Agent Platform* (AP): provê a estrutura física para a execução do SMA, o que inclui *hardware*, sistema operacional, os componentes de gerenciamento citados (AMS, DF, MTS), os agentes (não é padronizado pela FIPA).
- Software: inclui todos elementos não-agentes acessíveis aos agentes da plataforma.

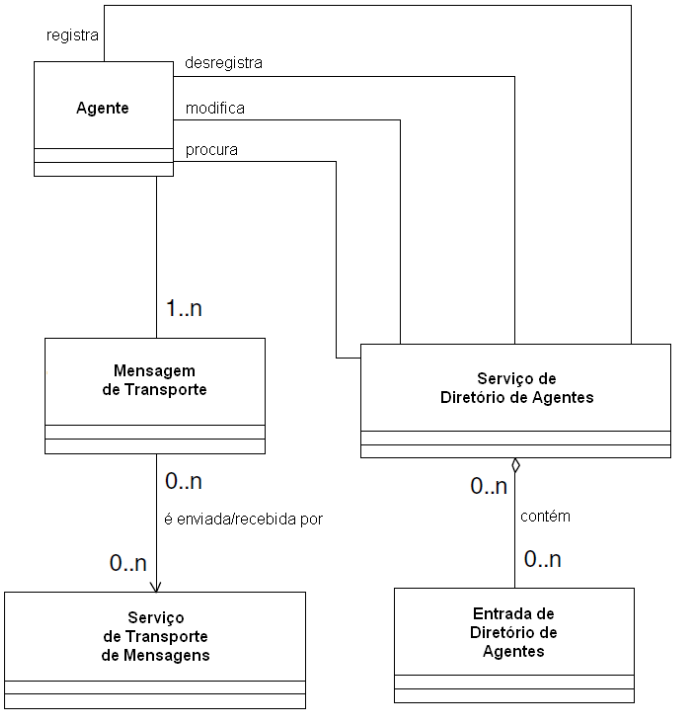

Figura 3.9: Modelo de referência FIPA da plataforma de agentes, adaptada de Foundation for Intelligent Physical Agents (FIPA) (2002)

Para a comunicação entre agentes, a FIPA definiu o padrão FIPA-ACL (*Agent Communication Language*), que especifica um conjunto mínimo de tipos de mensagens, chamados de atos comunicativos. O ato comunicativo, ou *performative*, representa a vontade do agente sobre a informação contida na mensagem. Alguns exemplos são o `query-if`, que pergunta a um agente se uma proposição é verdadeira ou falsa; `not-understood`, indicando que o emissor não compreendeu uma mensagem anterior e `request`, que solicita a realização de alguma ação.

Juntamente com informação contida na mensagem, outros elementos fazem parte do padrão FIPA-ACL, tais como agente emissor (*sender*), agente receptor (*receiver*), conteúdo e controle da conversação. A Figura 3.10 apresenta a estrutura da mensagem definida no padrão FIPA-ACL (Bellifemine et al., 2007).

O exemplo a seguir, presente em Foundation for Intelligent Physical Agents (FIPA) (2002) apresenta alguns dos elementos de uma mensagem ACL:

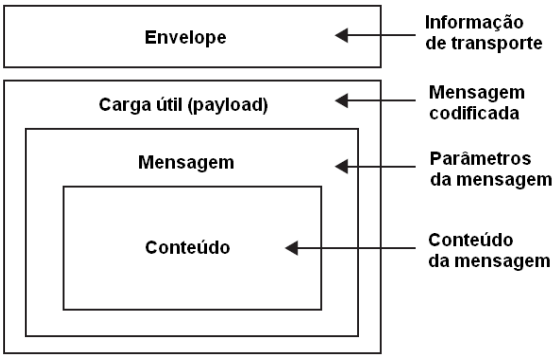

Figura 3.10: Estrutura da mensagem ACL, adaptada de Bellifemine et al. (2007)

```
(inform
  :sender agent1
  :receiver agent2
  :content (price good2 150)
  :language sl
  :ontology hpl-auction)
```

Neste trabalho, procurou-se utilizar uma plataforma de desenvolvimento de SMA que aderisse aos padrões da FIPA. A seção seguinte apresenta as características da plataforma JADE utilizada e sua relação com os padrões citados.

3.4.1 A plataforma JADE

O *Java Agent DEvelopment Framework* (JADE), desenvolvida pelo TILab (*Telecom Italia Lab*), oferece um *framework* para o desenvolvimento de aplicações multiagente. Segundo Bellifemine et al. (2007), o objetivo da plataforma é simplificar o desenvolvimento de SMA garantindo a conformidade com os padrões por meio de um conjunto de serviços e agentes que seguem as especificações da FIPA como serviços de diretório ou transporte de mensagens e um conjunto de protocolos de interação. JADE é desenvolvido completamente na linguagem Java. A última versão utilizada até o término deste trabalho foi a versão 3.6.1 de 04 de novembro de 2008.

A comunicação na plataforma JADE é realizada pela troca de mensagens. A plataforma de agentes pode ser distribuída em vários *hosts*, como, por exemplo, diversos computadores conectados em rede. Em cada um desses computadores, um contêiner em execução em uma máquina virtual Java – *Java Virtual Machine* (JVM) – fornece suporte para a execução dos agentes. A um conjunto de contêineres é dado o nome plataforma. Cada agente possui sua própria *thread* de execução e vários agentes podem existir em um mesmo contêiner. Um desses contêineres é dito o contêiner principal (*Main Container*) e nele estão os agentes principais AMS e DF. A Figura 3.11 apresenta um diagrama UML ilustrando as relações descritas.

Todos os componentes definidos pela especificação da FIPA para gerência de agentes apresentados na seção anterior estão disponíveis na plataforma JADE. Devido à conformidade com as especificações FIPA, um agente JADE pode interagir com agentes construídos

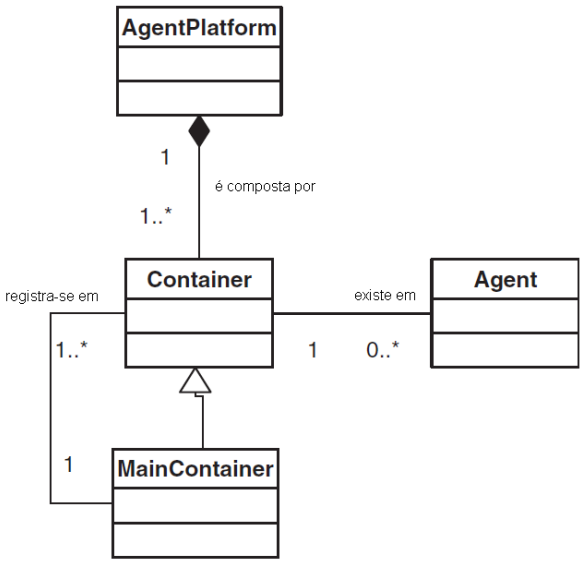

Figura 3.11: Diagrama UML dos elementos da arquitetura JADE, adaptada de Bellifemine et al. (2007)

em outras *frameworks* que também sigam os padrões FIPA. A Figura 3.12 apresenta as relações entre os componentes da arquitetura JADE (Bellifemine et al., 2007).

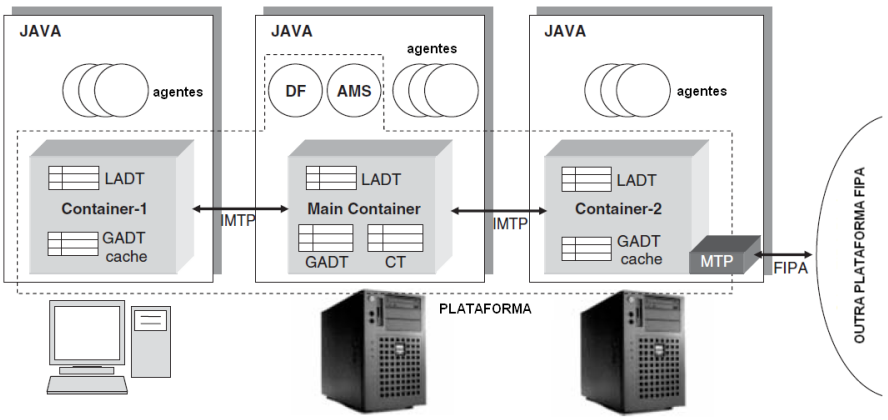

Figura 3.12: Relação entre elementos da arquitetura JADE, adaptada de Bellifemine et al. (2007)

Como pode ser observado na Figura 3.12, além dos agentes especiais AMS e DF, o contêiner principal mantém também duas estruturas importantes para o gerenciamento da plataforma:

- tabela de contêineres ou *Container Table* (CT): registro de todos os contêineres que formam a plataforma de agentes, incluindo seus endereços de transporte de mensagens;
- tabela global de descritores de agentes ou *Global Agent Descriptor Table* (GADT): registro de todos os agentes na plataforma, incluindo seu estado e localização.

Os demais contêineres mantém um *cache* local da GADT, para evitar que o contêiner principal torne-se um gargalo de desempenho nas consultas à GADT principal. Além disso, todos os contêineres mantém uma tabela local de descritores de agentes, a *Local Agent Descriptor Table* (LADT). Assim, para entregar uma mensagem, o contêiner consulta primeiro o endereço do destinatário na LADT. Caso o endereço não seja encontrado, ele consulta o cache local da GADT e por último solicita ao contêiner principal que consulte a GADT.

Pode-se notar que o contêiner principal, por ser único, torna-se um ponto de falha geral da plataforma, ou seja, qualquer falha no seu funcionamento inutiliza a plataforma. Para isso, JADE disponibiliza um serviço de replicação do contêiner principal como mecanismo de tolerância a falhas.

A utilização da plataforma JADE neste trabalho deve-se ao fato da mesma fornecer os serviços e agentes básicos para a gerência de um SMA, seguindo os padrões FIPA além de: (i) ser desenvolvido em uma linguagem com boa portabilidade, (ii) possuir boa documentação e uma comunidade ativa de usuários e desenvolvedores e (iii) ser software gratuito e de código aberto, distribuído sob a licença *GNU Lesser General Public License* (LGPL).

Com a definição da plataforma a ser utilizada neste trabalho, passou-se ao estudo da resolução distribuída de problemas em SMA, com ênfase nas questões de planejamento.

3.5 Planejamento

Segundo Ghallab et al. (2004), um problema é uma divergência entre um estado particular e o estado ideal desejado. A resolução de problemas é a diminuição ou eliminação dessa divergência. E o planejamento é o processo de encontrar os espaços necessários para modificar o estado inicial para o estado final desejado. Logo, o planejamento pode ser visto como parte do processo de resolução de problema que é completado pelo processo de execução, sendo que o próprio planejamento pode ser alterado durante a execução.

Atividades de planejamento distribuído envolvem um grupo de agentes no processo de planejamento. O planejamento distribuído está intimamente relacionado com a resolução distribuída de problemas, pois é em si um problema e também uma forma de resolver problemas (Weiss, 1999). A especificação e o funcionamento do esquema de planejamento do SMA proposto neste trabalho é apresentado no Capítulo 4.

Apesar da maioria das discussões sobre planejamento envolverem a geração de planos, a maioria dos sistemas reais requer sistemas capazes de intermear planejamento com execução, monitoramento, recuperação de falhas e revisão de planos (Ghallab et al., 2004).

Segundo Nau (2007), nas pesquisas de planejamento automatizado tem se assumido tradicionalmente que o planejador é um programa monolítico que soluciona o problema sozinho. Mas em aplicações mais complexas, o planejador é parte de um sistema maior em que outros agentes existem, sejam humanos ou artificiais. Portanto, em um SMA, o planejador deve levar em consideração o que os agentes estão buscando realizar, bem como a existência de intenções conjuntas (Wooldridge, 2002).

Uma proposta de taxonomia para o planejamento distribuído em um SMA, descrita em Wooldridge (2002) inclui:

- planejamento centralizado de planos distribuídos: um sistema centralizado de planejamento desenvolve um plano para um grupo de agentes; esses agentes receberão o plano e o executarão de forma distribuída;
- planejamento distribuído: um grupo de agentes trabalha cooperativamente para formular um plano centralizado, cada um contribuindo em diferentes aspectos do plano; a execução, no entanto, não cabe a esses agentes.
- planejamento distribuído de planos distribuídos: um grupo de agentes trabalha para formar planos individuais de ação, dinamicamente coordenando as suas atividades; nesse caso, os agentes podem ser auto-interessados e pode surgir a necessidade de um mecanismo de resolução de conflitos, como o uso de protocolos de negociação.

(Durfee and Lesser, 1991) apresentam uma proposta, denominada *Partial Global Planning*, em que os agentes comunicam seus planos locais para construir planos que são parcialmente globais. Os planos especificam interações entre ações para evitar os resultados limitados de planos puramente locais. A proposta também busca conservar as funcionalidades de tempo real, que podem ser perdidas no planejamento global.

Uma estratégia para a resolução distribuída de problemas é a divisão de tarefas. Para Russell and Norvig (2003), o planejamento é a tarefa de elaborar um sequência de ações para alcançar um objetivo e para isso a decomposição do problema deve ser realizada, criando assim subobjetivos que após serem alcançados devem ter seus resultados combinados para obter o resultado final. As fases da divisão de tarefas, apresentadas em Weiss (1999), são:

- decomposição da tarefa: gerar o conjunto de tarefas a ser distribuído, o que pode incluir dividir tarefas grandes em subtarefas que possam ser resolvidas por diferentes agentes;
- alocação: designar as subtarefas para os agentes apropriados;
- cumprimento de tarefas: cada um dos agentes cumpre suas subtarefas, o que pode envolver mais decomposições até o ponto em que um agente sozinho possa resolver a subtarefa;
- síntese dos resultados: os agentes enviam os resultados de suas subtarefas para o agente apropriado que tem o conhecimento de como compor esses resultados para obter a solução do problema original.

Wooldridge (2002) apresenta uma ilustração do processo de resolução distribuída de problemas. O autor mostra o processo de forma mais global, composto apenas de três passos, ilustrados na Figura 3.13.

O compartilhamento de resultados é outra estratégia para a resolução distribuída de problemas. Os agentes podem melhorar o desempenho dos resultados do grupo em termos de (Weiss, 1999):

- confiança: utilizando os resultados de outro agente para corroborar seus próprios resultados, aumenta-se o nível de confiança nos resultados obtidos pelo grupo;
- completude: soluções distintas podem ser combinadas, de forma que a abrangência da solução do grupo é maior;

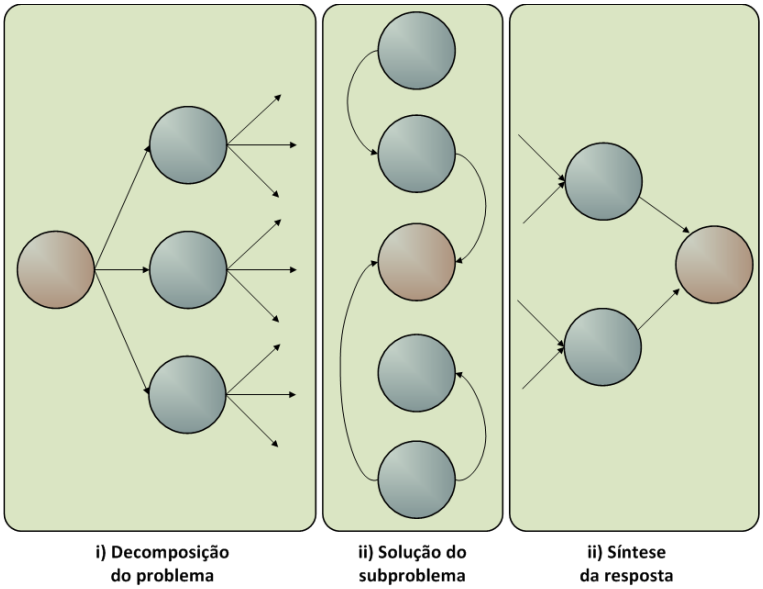

Figura 3.13: Passos da resolução distribuída de problemas, adaptada de Wooldridge (2002)

- precisão: com os resultados compartilhados, os agentes podem ajustar seus resultados para que a solução global tenha maior precisão;
- oportunidade: com a resolução em paralelo de subtarefas e o compartilhamento dos resultados, a solução global é alcançada mais rapidamente, mesmo que um só agente fosse capaz de resolver a tarefa maior sozinho.

3.5.1 Definição

Segundo Ghallab et al. (2004), a maioria das abordagens de planejamento utiliza uma formalização baseada em máquinas de estados que pode ser definidas pela quádrupla:

$$\sigma = (S, A, E, \gamma) \tag{3.4}$$

onde $S = \{s_1, s_2, \ldots\}$ é um conjunto finito ou recursivamente enumerável de estados, $A = \{a_1, a_2, \ldots\}$ é um conjunto finito de ações, $E = \{e_1, e_2, \ldots\}$ é um conjunto finito de eventos e γ é a função de transição do tipo:

$$S \times A \times E \rightarrow 2^S \tag{3.5}$$

A diferença de um evento para uma ação, é que o primeiro não está sobre o controle daquele que planeja, podendo ocorrer esporadicamente. Nesse caso podem ocorrer transições de estado decorrentes somente de ações ou somente de eventos, de forma que um evento e ação neutros são necessários e definidos como ϵ e $no-op$ respectivamente.

Acompanhando essa definição, um plano pode ser definido como um conjunto de ações que levam a estados de objetivo dentro do subconjunto S de estados. Pode levar também à satisfação e manutenção de uma condição ou estar associada à maximização de uma função de utilidade.

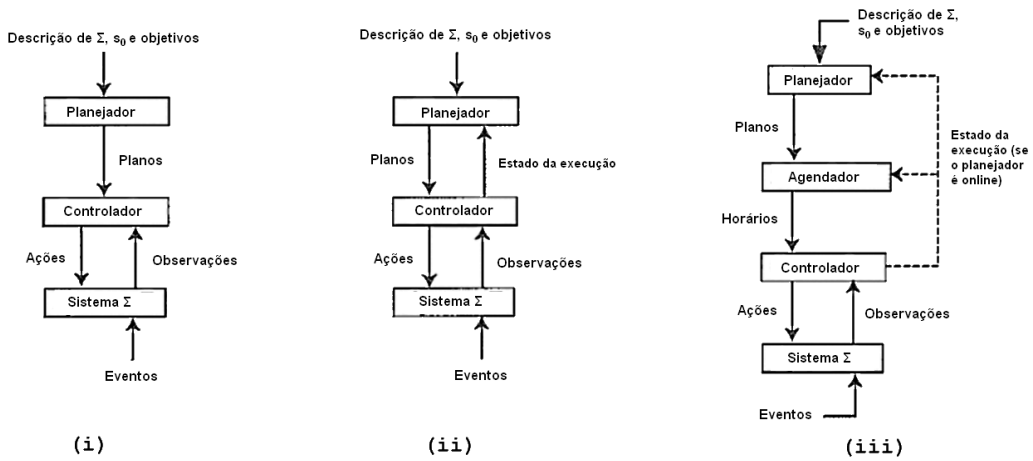

Figura 3.14: Modelo conceitual de planejamento *offline*, *online* e com um agendador separado, adaptada de Nau (2007)

A Figura 3.14 apresenta um modelo conceitual de três formas de planejamento: (i) *offline*, (ii) *online* e (iii) com o uso de um agendador (*scheduler*) separado (Nau, 2007). Nas três formas nota-se a presença de três componentes principais:

- planejador: sintetiza o plano para o controlador;
- controlador: realiza as ações de acordo com o plano, levando em conta o estado atual e
- sistema de transição σ, que se movimenta segundo a função de transição γ, apresentada na Equação 3.5.

Entre o planejamento e a execução podem haver modificações no estado de σ (modificações no mundo real), que podem dificultar o trabalho do controlador. Um modelo mais realista intercala planejamento e ação com supervisão e revisão de planos (Ghallab et al., 2004). O planejamento dinâmico (ou *online*) é obtido com a comunicação do estado de execução entre controlador e planejador, como pode ser visto na situação `ii` da Figura 3.14. Um componente agendador separado do planejador também pode ser utilizado, reduzindo a complexidade do planejador (Nau, 2007).

A *Planning Domain Definition Language* (PDDL) é uma linguagem específica para a descrição de domínios e problemas de planejamento, que é descendente de várias outras linguagens e formalismos (McDermott, 1998). A PDDL permite definir um domínio em termos dos predicados existentes, das ações possíveis, da estrutura de ações compostas e dos efeitos das ações. Os exemplos a seguir, apresentados por Fox and Long (2003) ilustram algumas das características da PDDL 2.1, que é uma extensão da PDDL para tratamento temporal de expressões em diferentes domínios.

O primeiro trecho de código a seguir é uma definição do domínio do problema. Nela devem ser descritos os tipos de objetos que compõem o domínio, os predicados aplicáveis a eles, as ações possíveis e os requisitos exigidos do planejador para resolver problemas desse domínio.

```
(define (domain vehicle)
```

```
    (:requirements :strips :typing)
    (:types vehicle location fuel-level)
    (:predicates (at ?v - vehicle ?p - location)
    (fuel ?v - vehicle ?f - fuel-level)
    (accessible ?v - vehicle ?p1 ?p2 - location)
    (next ?f1 ?f2 - fuel-level))

    (:action drive
        :parameters (?v - vehicle ?from ?to - location
            ?fbefore ?fafter - fuel-level)
        :precondition (and (at ?v ?from)
            (accessible ?v ?from ?to)
            (fuel ?v ?fbefore)
            (next ?fbefore ?fafter))
        :effect (and (not (at ?v ?from))
            (at ?v ?to)
            (not (fuel ?v ?fbefore))
            (fuel ?v ?fafter))
    )
)
```

No exemplo, o domínio `vehicle` (veículo) define três tipos: `vehicle` (veículo), `location` (localização) e `fuel-level` (nível de combustível). O predicado `at`, por exemplo, associa uma localização a um veículo. Apenas uma ação, `drive` (dirigir) é definida nesse domínio. Além dos seus parâmetros, uma ação também tem pré-condições que devem ser atendidas para que a ação possa ser realizada. Após sua realização, os efeitos esperados também são definidos. Nesse caso, um veículo dirige até uma determinada localização, se ela for acessível do ponto de origem, e tem seu nível de combustível reduzido.

O trecho de código PDDL a seguir apresenta um exemplo de problema do domínio `vehicle` a ser solucionado. A definição de um problema inclui o domínio do problema, os objetos envolvidos, a situação inicial definida por meio de predicados e o objetivo a ser alcançado.

```
(define (problem vehicle-example)
    (:domain vehicle)
    (:objects
        truck car - vehicle
        full half empty - fuel-level
        Paris Berlin Rome Madrid - location)
    (:init
        (at truck Rome)
        (at car Paris)
        (fuel truck half)
        (fuel car full)
        (next full half)
        (next half empty)
        (accessible car Paris Berlin)
```

```
        (accessible car Berlin Rome)
        (accessible car Rome Madrid)
        (acessible truck Rome Paris)
        (accessible truck Rome Berlin)
        (accessible truck Berlin Paris)
    )
    (:goal (and (at truck Paris)
        (at car Rome))
    )
)
```

Nesse exemplo, o objetivo é que ambos os veículos `car` e `truck` saiam de suas origens e cheguem aos destinos desejados. O veículo `truck`, por exemplo, possui nível de combustível `half` (metade), o que permite que ele dirija apenas um trecho. Sua origem é `Rome` e seu destino `Paris`. Nessas condições o objetivo pode ser atingido, pois o predicado `(acessible truck Rome Paris)`, permite que `truck` viaje diretamente entre essas localidades. Como efeito, `truck` chega em `Rome` e seu nível de combustível agora é `empty` (vazio). Logo, um exemplo de solução para esse problema seria a sequência de ações:

```
(drive truck Rome Paris half empty)
(drive car Paris Berlin full half)
(drive car Berlin Rome half empty)
```

3.5.2 Planejamento hieráquico

Conforme apresentado anteriormente, a resolução distribuída de problemas envolve a decomposição e divisão de tarefas entre agentes. Isso pode ser realizada de maneira hierárquica, para melhor lidar com a complexidade dos problemas (Russell and Norvig, 2003). No planejamento hierárquico, a cada nível da hierarquia o problema é reduzido a problemas menores por meio da decomposição de ações. O processo continua até que a complexidade das ações seja reduzida a um nível básico, chamado por Russell and Norvig (2003) de ações primitivas. Tais ações primitivas podem então ser realizadas sem dificuldade por um agente.

Uma *Hierarchical Task Network* ou rede de tarefas hierárquicas (HTN) é semelhante ao planejamento clássico, no entanto, o objetivo não é atingir um estado-objetivo e sim realizar um conjunto de tarefas. Uma das entradas do sistema é um conjunto de métodos que informa como decompor uma tarefa em um conjunto de subtarefas menores. Esse processo de decomposição é realizado recursivamente até que uma "tarefa primitiva" seja alcançada. Segundo (Ghallab et al., 2004), a HTN representa a técnica de planejamento mais utilizada na prática, porque correspondem à forma de pensar do especialista, como se descrevesse uma "receita" para resolver o problema.

Para definir uma HTN é necessário acrescentar algumas definições àquelas do planejamento clássico apresentado anteriormente. O primeiro conceito é o de tarefa. Cada tarefa é uma expressão da forma $t(r_1, \ldots, r_k)$ em que cada r_i é termo e t é um símbolo. Uma tarefa t é primitiva se t for um operador. Uma rede de tarefas (*task network*) é grafo acíclico $w = (U, E)$ em que U é um conjunto de nós e E é o conjunto de arestas.

A principal desvantagem do uso de HTNs, segundo Ghallab et al. (2004) é a necessidade de definir não só um conjunto de operadores como um conjunto de métodos para o domínio.

Em Weiss (1999) é apresentado um algoritmo para o planejamento hierárquico distribuído que permite aos agentes representar seus comportamentos planejados localmente em múltiplos níveis de abstração, os quais podem ser usados para resolver conflitos no planejamento. O algoritmo é apresentado a seguir:

1. Iniciar o nível de abstração atual como o nível de abstração mais alto.
2. Agentes trocam descrições de planos e metas de interesse no nível atual.
3. Planos sem conflitos potenciais são removidos. Se o conjunto restante for vazio, o protocolo termina. Caso contrário, determinar se é possível resolver os conflitos no nível atual ou se um nível mais profundo é necessário.
4. Se os conflitos tiverem que ser resolvidos em um nível mais profundo, definir o nível de abstração atual para o próximo nível e definir os planos e metas de interesse para o refinamento de planos com conflitos potenciais. Ir para o passo 2.
5. Se os conflitos tiverem que ser resolvidos neste nível:
   (a) Os agentes formam uma ordem total. O primeiro agente é o atual superior.
   (b) O atual superior envia seus planos aos demais.
   (c) Os outros agentes modificam seus planos para trabalhar apropriadamente de acordo com os planos do superior. Antes de confirmar seus planos com o superior atual, o agente verifica se seu plano não tem conflitos com nenhum dos superiores anteriores.
   (d) Uma vez que mais nenhuma mudança seja necessária entre os planos dos agentes inferiores, o superior atual, tornar-se um superior anterior. O próximo agente na ordem total torna-se o superior. Volta-se ao passo 5b. Se não houver mais agentes, o protocolo termina e os agentes têm seus planos coordenados.

Se o número de abstrações for finito e as modificações dos planos pelos agentes for restringida (evitando padrões cíclicos de geração de planos), é garantido que o algoritmo termina (Weiss, 1999). A resolução do problema em níveis mais altos de abstração reduz o tempo de planejamento e o custo de comunicação. A desvantagem é que em níveis mais detalhados, a precisão da solução pode ser maior.

Cox and Durfee (2003) apresentam um algoritmo para descobrir pontos de potencial cooperação e coordenação entre agentes planejadores que utilizam a representação hierárquica de planos, para evitar ineficiências na execução provenientes de interferências entre as ações dos agentes e oportunidades de cooperação perdidas.

Agendamento

Geralmente o plano fornecido para um SMA inclui um conjunto de ações estruturado ou parcialmente ordenado que não especifica um agendamento (*schedule*) das ações. O planejamento tem foco no que fazer, enquanto o agendamento no quando e como fazer (Ghallab et al., 2004). A Figura 3.15 apresenta essa ideia.

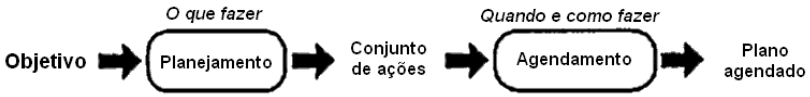

Figura 3.15: Planejamento e agendamento, adaptado de Ghallab et al. (2004)

O problema de agendamento de recursos é uma classe de problema bastante conhecida. Algumas classes de problema bem conhecidas são *flow-shop*, *open-shop* e *job-shop*. Em geral, os problemas de agendamento são problemas de otimização combinatorial difíceis. Logo, técnicas de aproximação são as mais utilizadas na prática (Ghallab et al., 2004). Um problema de agendamento é especificado pelo seguinte:

- um conjunto de recursos e sua disponibilidade futura;
- um conjunto de ações que precisam ser realizadas e seus requisitos de recursos;
- um conjunto de restrições sob essas ações e recursos;
- uma função de custo.

Uma abordagem possível para restrições temporais e de recursos é processá-las durante uma fase de crítica. Uma crítica pode disparar um processo de *backtrack* se as restrições não puderem ser satisfeitas. Isso é feito juntamente com o gerenciamento das variáveis de restrição por meio das ligações na rede de tarefas.

Na seção seguinte, é apresentado o conceito de Raciocínio Baseado em Casos (RBC), que pode ser combinado às questões de planejamento apresentadas. Segundo Marling et al. (2002), Raciocínio Baseado em Casos (RBC) foi utilizado no planejamento tanto clássico quanto hierárquico. Um exemplo dessa integração é o sistema de planejamento `PRODIGY/ANALOGY`, proposto em Veloso (1994). Nele os casos são utilizados para guiar o processo de busca de soluções do sistema de planejamento, o que representou ganhos pela diminuição do espaço de busca, que em princípio é exponencial com relação ao tamanho dos planos (Veloso, 1994).

Zimmerman and Kambhampati (2003) apresentam um estudo sobre a aplicação de técnicas de aprendizagem na melhoria do processo de planejamento no qual comenta algumas das vantagens e desvantagens de várias abordagens como o RBC. Dentre as vantagens citadas está o potencial de cortar grande parte do esforço de planejamento em situações em que problemas similares surgem frequentemente. As desvantagens citadas são o grande espaço de armazenamento necessário devido ao crescimento da base de casos, a sobrecarga e dificuldade da recuperação de casos similares e o custo de revisão dos planos que pode ser alto.

3.6 Raciocínio baseado em casos

A utilização de raciocínio baseado em casos (RBC) baseia-se no princípio de que problemas parecidos têm soluções parecidas. Mais do que isso, cada novo problema enfrentado tem o potencial de beneficiar-se de soluções anteriores ou de aumentar a base de problemas conhecidos, fornecendo um aprendizado incremental e contínuo. Aamodt and Plaza (1994)

afirma que vários resultados empíricos apontaram o papel dominante do uso de situações específicas experimentadas no passado, na resolução de problemas por seres humanos. Um exemplo é a solução de problemas por analogia. Aamodt and Plaza (1994) lembra que o RBC, no entanto, deve ser considerado uma forma de analogia intradomínio.

O principal conceito associado ao RBC é o conceito de caso, que pode ser descrito como uma experiência passada composta por três elementos: o estado inicial ou descrição do problema, uma solução, que apresenta os passos necessários para resolver o problema e um estado final que é representado por um conjunto de objetivos.

Segundo Marling et al. (2002), existem basicamente dois tipos de RBC: (i) interpretativo e (ii) de resolução de problemas. O interpretativo envolve o uso de casos passados, tipicamente chamados de precedentes, para criar uma análise e justificativa para a interpretação de um novo caso. O RBC para resolução de problemas envolve a adaptação de soluções de um problema antigo para alcançar os requisitos da nova situação.

O RBC para resolução de problemas pode ser divido em dois tipos mais específicos: o transformacional e o derivacional. No primeiro, soluções passadas são usadas diretamente. No segundo, o próprio processo de resolução de problema pela qual as soluções são derivadas é reutilizado.

Segundo Marling et al. (2002), todo uso do RBC segue um conjunto básico de passos:

1. analisar um novo caso;
2. baseado nessa análise, recuperar casos passados relevantes da base de casos;
3. baseado em um métrica de similaridade, ordenar os casos de acordo com o quão relevante ou úteis são com respeito ao novo caso;
4. selecionar um ou mais casos para serem utilizados na solução do novo caso;
5. criar a solução para o novo caso;
6. testar explorar a solução proposta e
7. se apropriado, adicionar o novo caso e sua solução à base de casos de forma que possa ser recuperado para uso futuro.

De maneira similar, Aamodt and Plaza (1994) define o ciclo do RBC, ilustrado na Figura 3.16, como um conjunto de quatro fases seqüenciais:

1. recuperação (*retrive*): realizada após a definição do problema. Consultando a base de casos passados, uma medida de similaridade deve ser aplicada para definir qual caso se aproxima mais do caso atual;
2. reuso (*reuse*): prevê a adaptação do caso base selecionado ao caso atual. Isso é feito através de ajustes diversos no caso recuperado, que podem incluir aproximação de valores numéricos, adição ou eliminação de passos, dentre outros. Desse processo surge a sugestão de solução.
3. revisão (*revise*): a solução sugerida é aplicada. Um especialista deve avaliar os seus resultados e ajustar a solução;
4. retenção (*retain*): a solução revisada pode ser armazenada como um novo caso, expandindo assim a base de conhecimento de casos.

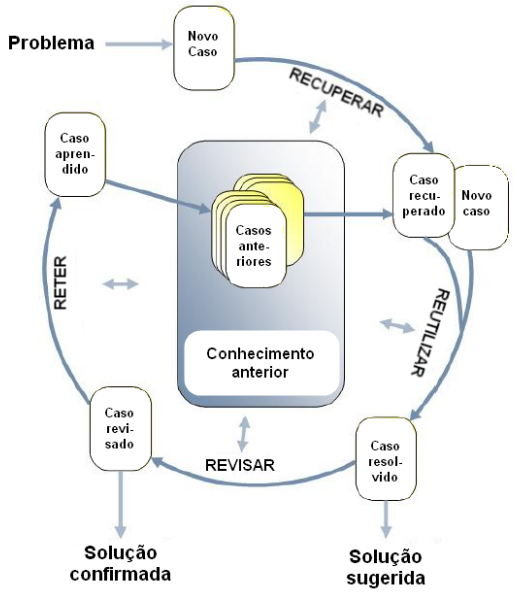

Figura 3.16: O ciclo do raciocínio baseado em casos, adaptada de Aamodt and Plaza (1994)

Kolodner (1993) enfatiza que é essencial para o uso de RBC a garantia de que os casos sejam similares o bastante ao novo caso para que se considere o reuso no novo problema.

Segundo Corchado et al. (2007), sistemas de RBC também têm sido aplicado com sucesso em técnicas de argumentação. Os autores afirmam que armazenar experiências passadas como casos é um meio de gerenciar informações incompletas, incertas ou inconsistentes. Um dos trabalhos presentes em Corchado et al. (2007) apresenta um agente que utiliza o RBC como mecanismo de raciocínio juntamente com o modelo deliberativo *Desire-Belief-Intent* (DBI). O objetivo da utilização de RBC juntamente com DBI é facilitar o aprendizado e adaptação e obter um maior grau de autonomia do que em sistemas DBI puros. Para isso, os casos devem ser representados por meio de desejos, crenças e intenções dentro do ciclo de RBC.

Marling et al. (2002) apresentam um levantamento dos trabalhos de integração do RBC em outros paradigmas. Apontam ainda que várias tarefas são beneficiadas pela integração do RBC como interpretação, argumentação ou planejamento. Ainda segundo os autores, a maioria das abordagens de RBC para planejamento que não assumem a existência de uma teoria completa do domínio utilizam uma analogia transformacional para a adaptação do plano, embora a analogia derivacional também possa ser usada.

A proposta deste trabalho leva em consideração a possibilidade de aplicação futura do RBC para facilitar a reutilização de conhecimento no planejamento de atividades do SMA proposto, uma vez que os exames periciais de crimes por computador apresenta uma grande ocorrência de problemas e soluções semelhantes, o que favorece o uso do RBC.

Capítulo 4

Proposta do Trabalho

Este trabalho propõe a utilização de um SMA no exame pericial de sistemas computacionais, sejam eles compostos por um computador isolado ou parte de um conjunto de computadores, sistemas e mídias de armazenamento de dados. O objetivo da utilização de uma abordagem multiagente é permitir a captura do conhecimento especializado dos examinadores em agentes inteligentes e autônomos que, trabalhando de forma distribuída e cooperativa, sejam capazes de sugerir a melhor ação a ser tomada para uma evidência encontrada no sistema examinado. O sistema é denominado *Multi-Agent Digital Investigation toolKit* (MADIK)[1].

Com a aplicação do SMA proposto, espera-se obter maior eficiência na realização dos exames periciais, com aumento da utilização de recursos computacionais ociosos, redução no tempo despendido e melhor aproveitamento da especialidade dos peritos na realização de exames mais complexos. O sistema também visa fornecer mecanismos de retenção e reutilização do conhecimento obtido de casos anteriores, que podem auxiliar os peritos menos experientes em seu trabalho e futuramente pode permitir a aplicação de técnicas de mineração de dados e descoberta de conhecimento.

MADIK não é proposto como um substituto das ferramentas forenses existentes. Conforme apresentado na Seção 2.1.3, o trabalho pericial é dividido em várias fases, das quais a principal é a fase do exame pericial, que pode ser dividida ainda em outras três subfases: (i) levantamento, (ii) extração e (iii) exame dos dados.

A proposta deste trabalho concentra-se quase exclusivamente na fase do exame pericial, com ênfase na subfase do exame dos dados, uma vez que os procedimentos associados às subfases de levantamento e extração de dados têm suporte mais amplo de ferramentas específicas. O exame dos dados, no entanto, permite uma grande amplitude de análise que nem sempre dispõe de ferramentas adequadas. A área em destaque na Figura 4.1 representa a inserção do MADIK no fluxo do trabalho pericial.

Essa concentração na fase do exame pericial, no entanto, não restringe a extensão futura da proposta para uso nas demais fases. Como trabalhos futuros, essas restrições podem ser reduzidas ou eliminadas por meio da integração com outras ferramentas forenses ou da expansão da aplicação do SMA a outras fases do exames pericial.

[1] A sigla é um trocadilho com a palavra *magic* e deve ser pronunciada da mesma forma.

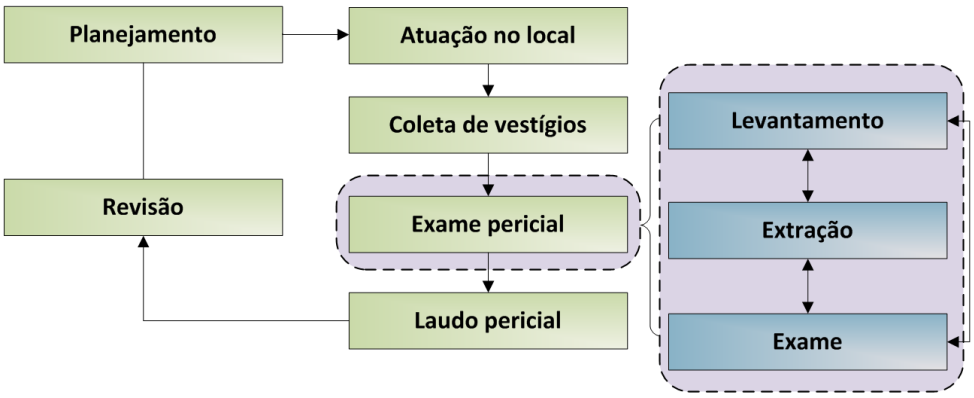

Figura 4.1: Inserção do MADIK proposto no trabalho pericial

4.1 Arquitetura e organização

A arquitetura proposta neste trabalho consiste em um SMA organizado hierarquicamente em quatro níveis, que utiliza uma estrutura de *blackboard*, conforme apresentada na Seção 3.2.5, para compartilhamento de dados entre os agentes. O sistema interage com o usuário por meio de uma camada de apresentação com três interfaces distintas para controle, monitoramento e revisão, detalhadas na Seção 4.5.3. No nível físico, interage com bases de dados e com o conteúdo das evidências examinadas. A Figura 4.2 ilustra essa arquitetura em três camadas. A organização interna do SMA é apresentada pela Figura 4.3.

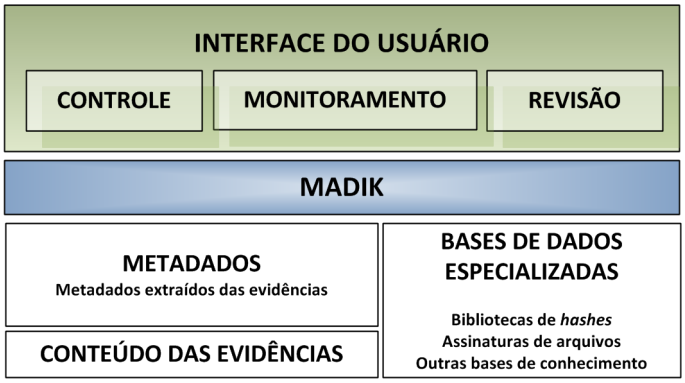

Figura 4.2: Arquitetura do sistema proposto

A organização hierárquica visa dar maior estabilidade e previsibilidade ao SMA, simplificando a coordenação dos agentes, a distribuição dos recursos e a resolução de conflitos. O nível mais baixo, chamado de nível especializado, é composto apenas por agentes que realizam atividades periciais específicas, como as citadas na Seção 2.3. No nível imediatamente superior, chamado operacional, é realizado o planejamento e coordenação das atividades do nível especializado. Analogamente, o nível seguinte, chamado tático, faz o mesmo com relação ao nível operacional. Por fim, no topo da hierarquia, o nível estratégico controla o nível tático. Os agentes responsáveis pelas atividades de planejamento e coordenação nas três camadas superiores são chamados gerentes.

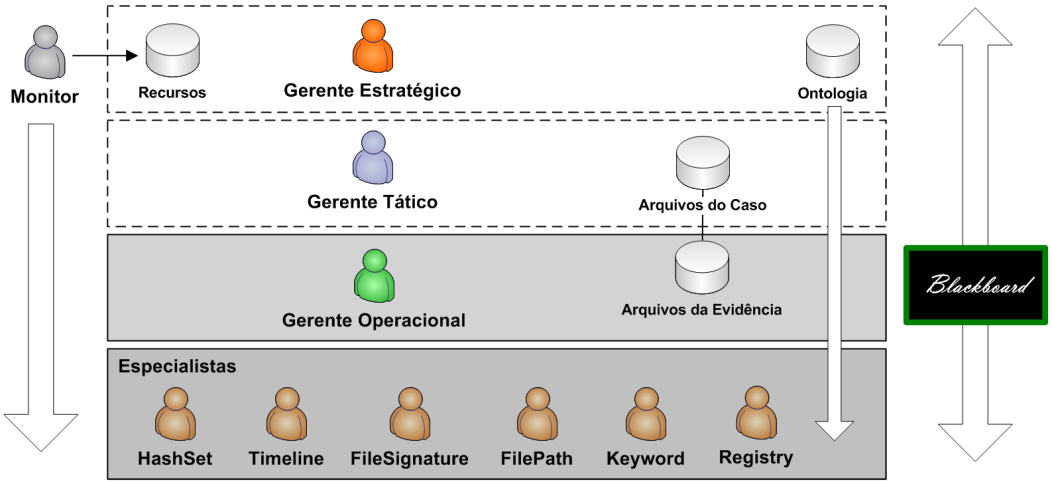

Figura 4.3: Organização do SMA proposto

A comunicação entre os agentes segue estritamente a hierarquia da organização, ou seja, um agente só se comunica com seu gerente imediato e um gerente só se comunica com seu gerente superior. Conforme discutido na Seção 3.2, uma das vantagens dessa restrição é a facilidade de controle organizacional e a clareza da cadeia de comando. Outra motivação para o uso desse modelo de organização é o fato das organizações policiais serem fundamentadas em estruturas hierárquicas. A Figura 4.4 apresenta a relação entre os níveis hieráquicos e os papéis desempenhados no sistema.

Figura 4.4: Relação entre os níveis hierárquicos e os papéis do sistema.

No nível mais alto da hierarquia são gerenciados os objetivos estratégicos da organização. Do ponto de vista do exames periciais, isso significa a priorização do atendimento dos casos pendentes, ou seja, a determinação de quais exames devem ser realizados primeiramente. Diversos fatores influenciam tal priorização, como antiguidade da solicitação de exame, risco de prescrição do crime, existência de suspeito preso e outros fatores esporádicos como repercussão pública ou situações de emergência. As características que descrevem um caso, no escopo deste trabalho, são as seguintes:

- principal crime ou tipo de crime atribuído aos suspeitos, como crime financeiro, fraude em licitação, exploração sexual de crianças e adolescentes, dentre outros;
- tempo transcorrido desde a solicitação do exame;
- risco da perda do prazo legal para impor pena ao criminoso (prescrição);
- existência de suspeito preso cujo resultado do laudo possa significar sua libertação ou a manutenção da prisão;
- presença de fatores emergenciais ou externos como grande repercussão pública ou ordem superior explícita de priorização.

No nível tático, trabalha-se com o conjunto de casos prioritários em atendimento no momento. Nesse nível, para cada caso é atribuído um gerente tático, cujo objetivo é coordenar os exames periciais das evidências associadas àquele caso. No âmbito do caso, portanto, é ele quem determina qual evidência deverá ser examinada primeiramente, segundo o seu potencial para a investigação. Esse potencial é determinado com base nos seguintes fatores:

- tipo de evidência - o tipo de mídia de armazenamento como disco rígido, *pen drive*, cartão de memória, CD, DVD, dentre outros;
- relação de proximidade com o suspeito - indica se o computador pertence ou é utilizado diretamente pelo suspeito; inclui-se aqui o grau de importância do suspeito na investigação;
- perfil de utilização - detalha o tipo de evidência segundo seu perfil de utilização; um disco rígido, por exemplo, pode desempenhar o papel servidor de correio eletrônico, de sistema gerenciador de banco de dados, de servidor de arquivos; um cartão de memória pode ser coletado de uma câmera digital ou de uma reprodutor de arquivos multimídia (como um *MP3 player*);
- capacidade de armazenamento e taxa de ocupação - pode ser levado em conta para determinar o potencial da evidência em termos da quantidade de dados existente na mídia e do espaço livre que pode ser examinado em busca de arquivos apagados;
- prioridade especial - indica que uma determinada evidência, por algum motivo, deve receber atenção especial.

Ressalta-se que os detalhes do caso estão acessíveis ao gerente tático para ajustar a priorização dos fatores citados, como, por exemplo, o tipo de crime em investigação. A seguir, é apresentado um exemplo de um caso e de duas evidências relacionadas:

- Identificação: caso 8765/2009
- Tipo de crime: exploração sexual infantil de crianças e adolescentes
- Suspeito preso: sim
- Data da solicitação: 20/12/2008
- Risco de prescrição: não

- Situação especial: não
- Evidências:

    1. disco rígido, 160 GB de capacidade nominal de armazenamento, com taxa de ocupação de 45%, coletado do computador pessoal localizado na residência do principal suspeito e
    2. cartão de memória, 2 GB de memória, 95% de taxa de ocupação, encontrado no carro do mesmo suspeito em uma câmera digital.

Os casos e evidências possuem ainda uma descrição do seu estado de atendimento:

- pendente - aguardando o início dos exames;
- em andamento - os exames estão em fase de realização;
- suspenso - os exames foram iniciados, mas por algum motivo foram suspensos e
- terminado - o exame foi completamente concluído segundo o planejamento do gerente responsável.

Examinando uma evidência encontra-se um grupo de especialistas subordinados a um único gerente operacional. Ou seja, para cada evidência selecionada para exame pelo gerente tático é associado um gerente operacional, que determina quais especialistas são os mais adequados para examinar aquela evidência. Nesse nível, o gerente operacional tem a sua disposição não só informações quanto à natureza da evidência, mas também informações sobre o caso, vindas do nível estratégico. No nível especializado, os agentes especialistas designados pelo gerente operacional também terão acesso a todas essas informações, além do conteúdo da evidência para a realização dos exames.

Cada agente especialista possui um mecanismo de inferência baseado em uma ou mais regras de produção. Além da regra em si, é armazenado no *blackboard* o perfil da regra, ou seja, informações adicionais sobre os requisitos e funcionamento daquela regra. Essas características das regras são muito importantes, pois são levadas em consideração durante o planejamento e principalmente na alocação de recursos. Os requisitos especificados pelas regras são:

- sistema operacional - pode ser especificado se houver alguma necessidade específica;
- ambiente - para regras que incluam a execução de scripts ou código em outro ambiente como Perl, Python ou Java; por exemplo, a versão mínima da JVM;
- requisitos mínimos de CPU e memória RAM;
- nível de acesso aos arquivos - é dividido em três categorias: `metadata`, `partial content`, `full content`; no primeiro apenas dados obtidos do sistema de arquivos são utilizados, o que significa que não há necessidade de acesso ao conteúdo dos arquivos no servidor de arquivos; os nível seguintes requerem acesso ao conteúdo parcial ou completo;
- estimativa de tempo - média de tempo para analisar um arquivo levando em consideração o nível de acesso e

- abrangência da regra - aplica-se ao caso em que uma determinada regra necessita de acesso apenas a um conjunto restrito e específico de arquivos, ao conjunto completo ou se não há restrições; essa característica permite ao gerente operacional distribuir a execução da regra sem maiores restrições e permite também que ele dê preferência para a alocação nas mesmas máquinas de tarefas que trabalhem com o conjuntos de arquivos, permitindo a utilização de um *cache* local, evitando a retransmissão de arquivos.

Portanto, pode-se notar que a cadeia de comando desde o gerente estratégico percorre toda a hierarquia, bem como as informações fluem de um nível para o outro. Com essa organização dos agentes, busca-se obter um sistema escalável, que possa tratar do caso mais simples onde é necessário analisar apenas algumas evidências isoladas, até os casos mais complexos que envolvam uma grande quantidade de casos. A Figura 4.5 ilustra a estrutura da organização durante a execução de dois casos simultâneos.

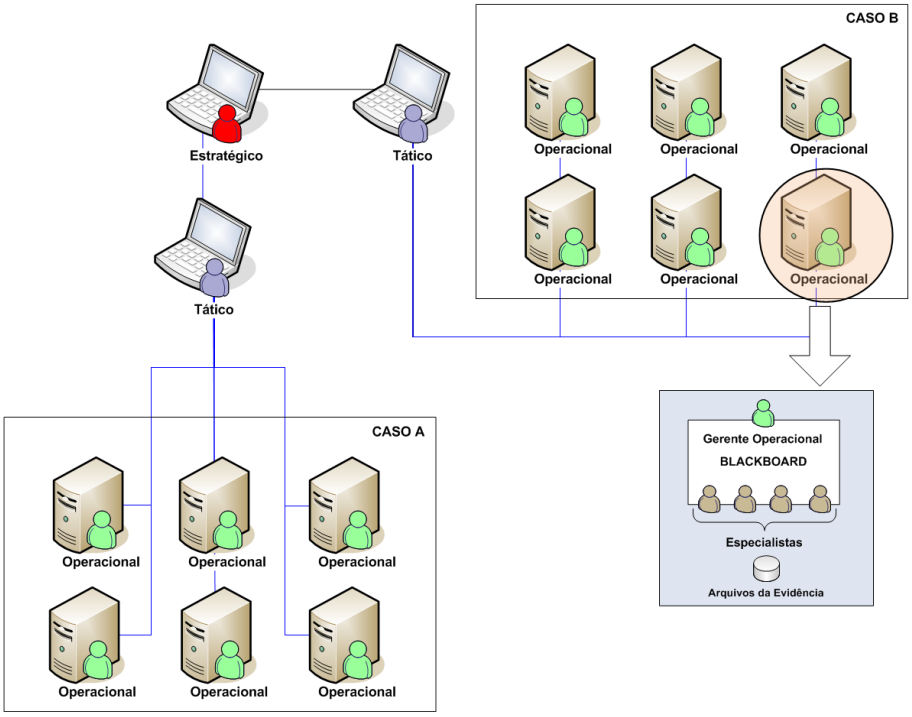

Figura 4.5: Estrutura da organização durante o exame pericial

Complementando a arquitetura é essencial destacar o *blackboard*, que está à disposição de todos os agentes especialistas e gerentes que estejam trabalhando em um mesmo caso. Assim, todas as descobertas e conclusões dos agentes especialistas são colocadas em um mesmo local, o que permite posteriormente o trabalho de cruzamento e correlação dos resultados. Esse trabalho é coordenado pelo nível tático, que possui a visibilidade de um caso como um todo. Utilizando o *blackboard* os agentes especialistas também podem consultar os resultados de outros agentes no intuito de complementar seus resultados ou evitar a repetição de tarefas.

Na Figura 4.3, duas bases de dados são mostradas, uma relativa a uma única evidência, chamada de dados locais e outra chamada de dados do caso, que é uma visão do

conjunto de dados locais de cada evidência que está disponível para o nível tático. No nível estratégico, há ainda uma base ontológica que contém as ontologias necessárias ao funcionamento da organização. O gerente estratégico, portanto, é responsável pela propagação de atualizações nessas ontologias, garantindo que todos os níveis da hierarquia estejam utilizando um mesmo vocabulário comum.

Ao longo da hierarquia da organização existe um serviço provido por agentes de monitoramento que está à disposição do nível estratégico, o que permite que o topo da organização esteja ciente das condições de trabalho de cada um dos agentes, independentemente da comunicação que vem dos gerentes dos níveis inferiores. Esta quebra de hierarquia é apenas aparente, pois não interfere na realização do trabalho de cada um dos níveis, apenas fornece informação sobre a saúde do SMA, que é de importância estratégica para o planejamento global da organização.

4.2 Ciclo de execução do SMA

Nesta seção é detalhado o ciclo de execução do SMA proposto. Esse ciclo define as ações e objetivos do sistema em cada um dos níveis da hierarquia, bem como define as interações e eventos que podem ocorrer durante a execução das atividades.

Primeiramente, o SMA inicia-se com o estabelecimento da organização, o que ocorre pelo nível mais alto da hierarquia. Logo, um gerente estratégico é o agente iniciador de toda a hierarquia, que instancia o *blackboard* e todas as ontologias necessárias. O gerente estratégico localiza-se no contêiner principal da plataforma de agentes. Em cada um dos demais contêineres, um agente de monitoramento é instanciado. A partir daí, os agentes de monitoramento informarão periodicamente o gerente estratégico sobre a disponibilidade e estado dos recursos do contêiner.

Com o estabelecimento dessa plataforma inicial, o gerente estratégico pode receber novos casos para atendimento ou continuar o trabalho nos casos já pendentes. O gerente estratégico inicia então a sua atividade de planejamento, que segue os seguintes passos:

1. definir um conjunto de trabalho com alguns dos casos de maior prioridade;
2. contabilizar os recursos disponíveis para cada caso;
3. instanciar os gerentes táticos nos contêineres de maior grau de disponibilidade;
4. atribuir a cada um deles o objetivo de atender um dos casos do conjunto de trabalho;
5. solicitar aos gerentes o seu planejamento para que possa distribuir os recursos;
6. distribuir os recursos disponíveis segundo os planos recebidos do nível tático e
7. coordenar o trabalho dos gerentes táticos e realizar ajustes no plano.

O objetivo final do gerente estratégico é atender todos os casos pendentes, procurando utilizar o máximo dos recursos disponíveis e mantendo em atendimento sempre os casos de maior prioridade, segundo os fatores citados na Seção 4.1. Esses passos são, portanto, realizados iterativamente até que todos os casos tenham sido atendidos.

No terceiro passo, os gerentes táticos são instanciados nos contêineres que apresentam os melhores recursos e que estão disponíveis a mais tempo. O objetivo é evitar que

contêineres mais sujeitos à interrupções sejam utilizados para os gerentes, pois o custo de substituição de um agente aumenta quanto mais alta for sua posição na hierarquia.

O passo 5 requer que os gerentes táticos realizem suas atividades de planejamento. Os passos são semelhantes ao nível estratégico:

1. definir um conjunto de trabalho com algumas das evidências de maior prioridade do caso atribuído;
2. instanciar os gerentes operacionais no mesmo contêiner em que se encontra;
3. atribuir a cada um deles o objetivo de conduzir o exame de uma das evidências do conjunto de trabalho;
4. solicitar aos gerentes o seu planejamento para que possa escolher os recursos;
5. receber o planos dos gerentes operacionais, sintetizá-los e encaminhá-los ao gerente estratégico;
6. receber os recursos e entregá-los aos gerentes operacionais e
7. coordenar o trabalho dos gerentes operacionais e realizar ajustes no plano.

O primeiro passo envolve o uso de RBC, que é descrito ainda neste capítulo. No passo 2, os gerentes são instanciados no mesmo contêiner do gerente tático porque o gerente ainda não tem autorização de utilizar qualquer outro recurso da plataforma. Após o recebimento dos recursos, os gerentes operacionais podem ser migrados para os novos contêineres.

Nesse nível, o passo 4 requer que cada gerente operacional realize seu planejamento. Note-se que nesse ponto ambos os gerentes tático e estratégico aguardam informações para finalizar o planejamento. Os passos também são semelhantes:

1. consultar os agentes disponíveis na plataforma e seu desempenho nesse tipo de caso e evidência;
2. ordenar as regras de produção dos agentes que deseja utilizar, segundo sua ordem de preferência;
3. informar ao gerente tático sua preferência;
4. aguardar o recebimento dos recursos;
5. instanciar os agentes especialistas nos recursos recebidos e
6. coordenar o trabalho dos agentes especialistas e realizar ajustes no plano.

O passo 2, remete à utilização de RBC, conforme descrito posteriormente neste capítulo. É interessante notar que os recursos recebidos podem não atender a preferência do gerente. Nesse caso, ele deve instanciar os agentes na ordem que for possível. Os pedidos de recursos não atendidos vão para uma fila de espera. Para executar as regras que não puderam ser atendidas no momento, ele deverá aguardar a próxima rodada de planejamento, que levará em consideração essa fila. A definição de prioridade para recebimento dos recursos da fila segue a prioridade do caso e da evidência.

Ao serem instanciados, os agentes especialistas recebem as informações do caso e da evidência, bem como qual regra de produção devem utilizar. Com base nessas informações, ele podem iniciar o seu trabalho imediatamente. O agente especializado pode realizar um planejamento próprio, determinando, por exemplo, um subconjunto de arquivos que deseja examinar primeiro. É importante que o agente especializado utilize essas informações no intuito de maximizar o volume de descobertas relevantes no menor tempo possível.

Ao concluir o seu trabalho, o agente especialista comunica o seu gerente, que pode agora executar uma nova regra utilizando o recurso agora disponível. Essa nova regra pode envolver o instanciamento de outro tipo de agente e a dispensa do agente anterior. Se o recurso ocupado for particularmente escasso, provavelmente haverá registros de solicitação dele em uma fila de espera e, portanto, ele deverá ser liberado e cedido a outro gerente.

4.3 Utilização do *blackboard*

Durante o exame dos arquivos contidos em uma evidência, os agentes especialistas inserem suas recomendações no *blackboard*, as quais podem ser de quatro tipos:

- `ignorar` - indica que o arquivo não precisa ser examinado em detalhe;
- `incluir` - indica que um arquivo deveria ser incluído no relatório do caso, porque foi encontrado algum dado relevante nele ou pela natureza do arquivo em si;
- `alertar` - é o maior nível de atenção para um arquivo; indica a necessidade de exame prioritário e
- `informar` - serve para informar que características extraídas do arquivo examinado podem auxiliar na decisão do especialista humano e nas atividades de correlação; pode ser adicionado um viés positivo ou negativo à recomendação (com um sinal de `+` ou `-`), na direção de incluir ou ignorar o arquivo, respectivamente.

A estrutura do *blackboard* inclui as seguintes informações:

- `arquivo` - identificação do arquivo examinado;
- `agente` - o nome do agente que fez a recomendação;
- `regra` - regra de produção utilizada;
- `recomendação` - é a recomendação de fato feita pelo agente; segundo os quatro tipos definidos; quando as recomendações divergem, há uma situação de conflito que deve ser resolvida pelo gerente operacional;
- `descrição` - uma explanação da recomendação, que serve para esclarecê-la para o perito revisor do caso e
- `tempo gasto` - tempo em milissegundos necessário para examinar o arquivo; é utilizado posteriormente para determinar o desempenho dos agentes.

A Tabela 4.1 apresenta exemplos de recomendação de vários tipos inseridas no *blackboard*.

Tabela 4.1: Exemplos de recomendação no *blackboard*

Arquivo	**Agente (Regra)**	**Recomendação/Descrição**
DSC00881.JPG	FileSignatureAgent (DigitalPhoto)	Informar (+): possível imagem de câmera digital
kernel32.dll	HashSetAgent (KnownSystemFile)	Ignorar: arquivo do sistema operacional Microsoft Windows XP
pthcxxx.jpg	HashSetAgent (KnownChildPorn)	Alertar: possível imagem de pornografia infantil
contrato.doc	KeywordAgent (CustomKeywordList)	Informar (+): palavra-chave encontrada no arquivo

4.3.1 Correlação de evidências

O nível tático, por possuir uma visão global de um caso, é o responsável pelas atividades de correlação de informações. O objetivo dessas atividades é buscar informações que permitam relacionar duas ou mais evidências.

A correlação é realizada pelos agentes especializados que possuem regras de produção para tal fim. Nesse caso, o gerente tático designará um gerente operacional para coordenar as atividades de correlação. O funcionamento é semelhante ao exame de uma evidência, porém engloba os arquivos de todas as evidências do caso, bem como tudo que foi inserido no *blackboard*.

Por envolver informações provenientes de várias evidências, os trabalhos de correlação são iniciados apenas após o término das análises conduzidas pelos gerentes operacionais do caso.

Algumas possibilidades de regras de correlação podem ser citados:

- determinar um conjunto de arquivos comuns a várias evidências, permitindo ignorar sistemas e aplicativos em instalações padronizadas, que não têm relevância para a investigação; isso reduz o volume de arquivos a ser examinado pelo especialista humano;
- encontrar arquivos que tenham sido compartilhados entre os suspeitos, o que deve ser apresentado prioritariamente ao examinador;
- relacionar mensagens de correio eletrônico encontrados em computadores distintos como indício de comunicação e relação entre suspeitos, como ilustra a Figura 4.6;
- descobrir mídias removíveis como *pen drives*, CDs e cartões de memória que foram utilizados no sistema ou
- encontrar dados pessoais (números de CPF, telefone e CEP), atalhos e arquivos recentes que apontam para outras evidências.

4.3.2 Avaliação das recomendações

Após a execução dos agentes, várias recomendações são inseridas no *blackboard*. Antes de serem apresentadas ao examinador humano para revisão, elas devem ser avaliadas pelo

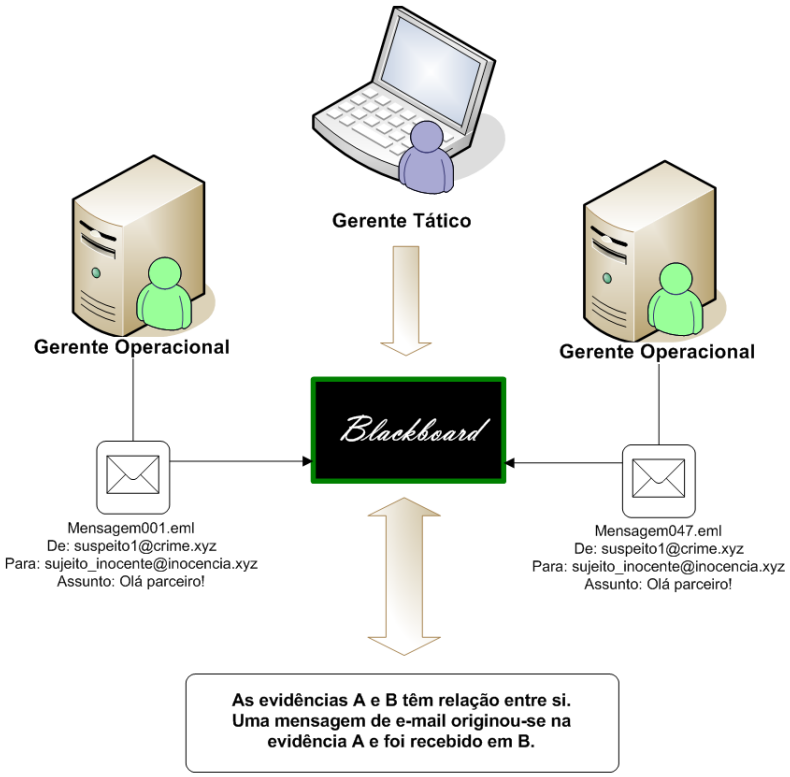

Figura 4.6: Correlação de duas mensagens de *e-mail* em evidências distintas

gerente operacional. Caso exista algum conflito nas recomendações dos agentes, o gerente operacional resolverá o conflito e decidirá o resultado final. Isso é feito com base na confiabilidade do agente em casos anteriores de natureza semelhante.

O nível de confiança em um especialista é estabelecido por meio da fase de revisão, na qual o especialista humano concorda ou discorda das recomendações finais apresentadas pelo gerente operacional. Os agentes cujas recomendações iniciais estavam em consonância com a decisão do especialista têm seu nível de confiança aumentado, enquanto os demais perdem um pouco dessa confiança. Lembrando mais uma vez que a natureza do caso é sempre levada em consideração. Com isso, espera-se que quanto mais o sistema for utilizado, mais precisas serão as recomendações. O estudo dos erros e acertos dos agentes especialistas podem permitir a descoberta de padrões de sucesso ou falha, auxiliando na melhoria das regras de produção.

Para simplificar o trabalho de revisão pelo especialista humano, os arquivos com menor índice de conflitos e com mais recomendações de inclusão e alertas (arquivos mais interessantes para a investigação) estarão sempre no topo da lista. Assim sendo, todo arquivo inicia o exame com o mesmo nível de importância e recebe um viés negativo ou positivo ao longo da execução do SMA.

O índice de confiabilidade média T das recomendações para um arquivo f é obtido a partir dos cálculos da Equação 4.1, considerando as recomendações r_i feitas pelos agentes a_i em uma escala numérica e a confiabilidade da regra de produção utilizada $C(a_i)$, em casos semelhantes. Essa é uma proposta inicial para o ranqueamento das recomendações visando uma futura introdução de mecanismos de reputação dos agentes.

$$T(f) = \frac{C(a_1) \times r_1(f) + C(a_2) \times r_2(f) + \ldots + C(a_n) \times r_n(f)}{\sum_{i=1}^{n} C(a_i)} \tag{4.1}$$

A transformação da recomendação em um valor numérico segue a escala definida na Tabela 4.2 a seguir. Com a realização dos cálculos e o uso dessa escala, é possível ordenar os arquivos do caso segundo a confiabilidade e relevância crescente ou decrescente dos arquivos do caso, facilitando a revisão pelo examinador humano:

Tabela 4.2: Escala numérica das recomendações dos agentes

Recomendação	**Viés**	**Valor numérico**
ignorar	negativo	-2
informar	negativo	-1
informar	positivo	+1
incluir	positivo	+2
alertar	positivo	+5

Se um arquivo não é examinado por nenhum agente especialista, ele é incluído em um conjunto de arquivos não examinados. Esse conjunto deverá ser examinado diretamente pelo próprio especialista humano. Essa situação não é desejada, pois quanto maior for esse conjunto, menor será o benefício da redução no volume de arquivos, propiciado pelo SMA. Portanto, esse conjunto define a abrangência do exame realizado pelo gerente operacional. O exame desse conjunto é importante para a descoberta de novas regras de produção, que aumentem essa abrangência.

A Tabela 4.3 apresenta o exemplo de avaliação dos resultados para um mesmo arquivo f avaliado por três agentes. Perceba que um dos agentes utiliza mais de uma regra sobre o mesmo arquivo.

Tabela 4.3: Exemplo da avaliação das decisões dos especialistas

Agente (a)	**Regra (r)**	**Recomendação**	**Confiabilidade**	$C(a) \times r$
Agente 1	Regra 1	Informar (+)	0,90	0,90
Agente 2	Regra 2	Ignorar	0,35	-0,70
Agente 3	Regra 5	Incluir	0,65	1,30
Agente 3	Regra 4	Informar (+)	0,80	0,80

Nesse exemplo há conflito entre a recomendação do Agente 2 e as demais recomendações. O conflito é resolvido utilizando a Equação 4.1. O valor de $T(f)$ nesse caso é de 0,85. Por ser um valor positivo, o gerente operacional sugere a inclusão do arquivo, embora a confiabilidade não seja muito alta. Nesse caso, outros arquivo com uma confiabilidade maior seriam apresentados primeiro ao perito revisor. Caso o revisor concorde com a recomendação final, as regras concordantes, de valor positivo, têm um aumento na sua confiabilidade, enquanto a confiabilidade da Regra 2 do Agente 2 tem sua confiabilidade diminuída. Caso o revisor discorde da decisão, o oposto ocorre.

4.4 Coordenação e planejamento no SMA

O trabalho de coordenação da execução é simplificado pela hierarquia, uma vez que cada nível coordena apenas o nível imediatamente inferior. As principais atividades são:

- controle de execução das tarefas: no nível operacional, por exemplo, significa acompanhar o andamento das tarefas atribuídas a cada especialista;
- requisição de recursos: solicitação de recursos necessários para a execução de uma regra;
- liberação de recursos: o recurso pode ser liberado porque não é mais necessário, porque não pode mais ser utilizado ou porque foi requisitado pelo nível hierárquico superior.

No nível especializado, durante a execução, os agentes especialistas podem ser interrompidos por determinação de um nível superior, provavelmente devido a alguma necessidade de redistribuição de recursos. Isso exige que antes de iniciar qualquer trabalho, os agentes especialistas observem no *blackboard* se parte do trabalho já foi feito anteriormente, evitando repetição de exames.

Dois protocolos são utilizados amplamente na composição dos demais diálogos do SMA, são eles *FIPA-Request* e *FIPA-Contract-Net*, ambos padronizados pela FIPA (Foundation for Intelligent Physical Agents (FIPA), 2002), apresentados nas Figuras 4.7 e 4.8, respectivamente.

Alguns do eventos que envolvem atos de comunicação específicos incluem: notificação de novo caso, notificação de nova evidência, remoção de recurso, oferta de recursos para os gerentes, interrupção de execução e monitoramento de recurso.

4.4.1 Protocolo de Planejamento

O protocolo de planejamento inicia-se com o nível superior da hierarquia, após este avaliar as prioridades dos casos pendentes e observar o conjunto de recursos disponível. Ele emite um pedido de plano aos gerentes táticos informando aos mesmos que eles possuem um determinado número de recursos. Observe que os recursos são todos tratados de maneira igual, considerando-se apenas o número de máquinas disponíveis.

Assim sendo, o gerente tático ao ser informado que dispõe de um número X de recursos para atender o caso que é de sua responsabilidade, repete o processo para o nível operacional, também já informando a cada gerente operacional o número de recursos disponível para cada um. Cabe lembrar que também é informado para os níveis inferiores as características do caso e da evidência.

Com isso, o gerente operacional seleciona um conjunto de agentes, mais especificamente um conjunto de regras de produção, e envia a sua ordem de preferência de alocação de recursos ao gerente tático. Todos têm conhecimento dos recursos disponíveis devido ao uso do *blackboard*. Assim todos os recursos disponíveis estão expostos, mesmo que o gerente tenha o conhecimento de que não necessariamente ele receberá o recurso solicitado.

Dessa forma, o gerente tático possui os planos de todos os seus gerentes operacionais. Com esses planos em mãos ele define quais desses recursos serão entregue para cada gerente operacional segundo a prioridade da evidência atribuída. Por fim, a solicitação de recursos é apresentada ao gerente estratégico, que realiza o seguinte procedimento:

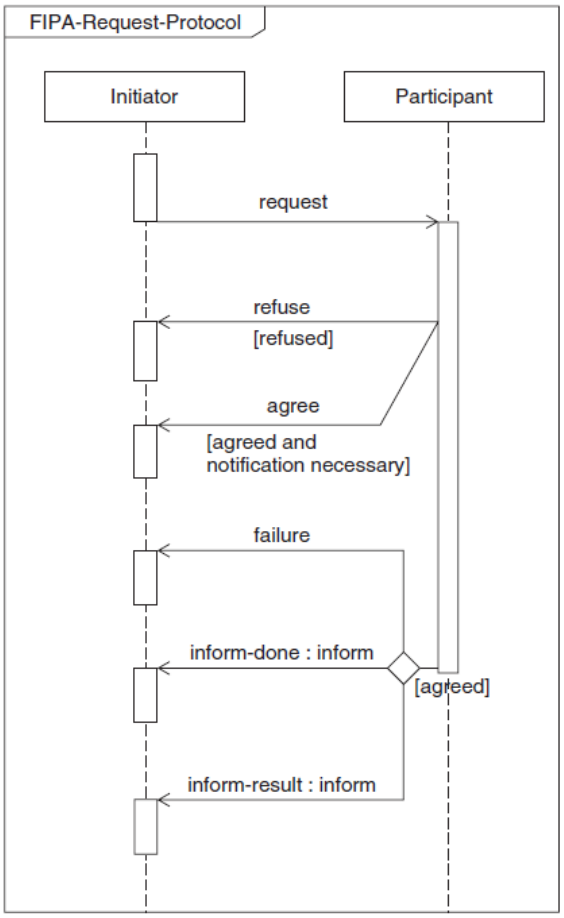

Figura 4.7: Protocolo FIPA-Request (Bellifemine et al., 2007)

1. do caso mais importante para o menos importante, atenda a primeira requisição de recurso;
2. em seguida, atenda à segunda requisição; se ela não puder ser atendida por ausência de um recurso, coloque-a em uma fila de espera.
3. repita o processo para as demais requisições.

Quando um recurso é liberado, a fila de espera é consultada. Caso alguém esteja esperando o recurso é realizada uma comunicação e a troca de recursos é feita com o gerente. O gerente deve liberar um recurso para receber o da fila de espera. Observe que a fila de espera já se encontra ordenada por prioridade dos casos, evidências e regras de produção.

Durante a execução um recurso pode tornar-se indisponível. O perdedor do recurso então envia uma mensagem ao seu gerente tático, que por sua vez envia um mensagem ao gerente estratégico, que insere o pedido de reposição na fila de espera. Com a chegada de um novo recurso, a fila é consultada e se o recurso for adequado, ele é entregue ao gerente solicitante e retirado da fila.

Essa solução visa reduzir a necessidade de replanejamento a cada mudança de recurso com vistas a manter a estabilidade dos recursos entregues inicialmente para cada gerente,

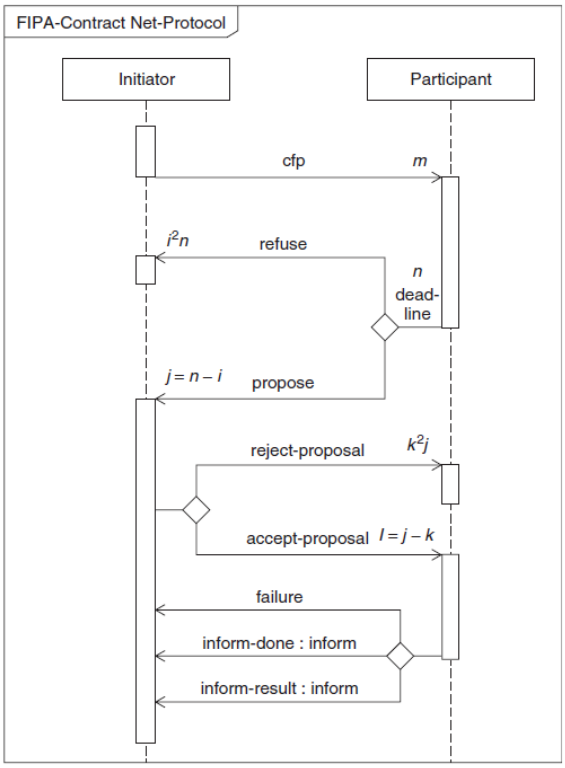

Figura 4.8: Protocolo FIPA-Contract-Net (Bellifemine et al., 2007)

especialmente se o sistema estiver em execução em uma grande rede com muitas conexões e desconexões de recursos.

No entanto, é necessário que haja previsão de uma ordem de replanejamento global caso a fila de espera contenha um grande percentual de pendências de um caso de maior prioridade, o que significa que o caso não está recebendo a prioridade devida. Nesse caso um replanejamento geral pode afetar drasticamente os demais casos, que deverão ceder alguns de seus recursos para o caso de maior prioridade.

O replanejamento deve ocorrer sempre que uma nova evidência ou caso de maior prioridade for adicionado ao sistema. Os níveis superiores de gerência devem estar cientes desses eventos e observar se afetam as atividades atuais ou seus passos subsequentes. Um caso novo de maior prioridade, por exemplo, pode significar a interrupção no trabalho de vários gerentes operacionais e de seus especialistas. Todos os agentes da plataforma devem ter ciência da possibilidade de interrupção, registrando no *blackboard* toda a informação necessária para continuar o trabalho posteriormente.

No nível operacional, quando um especialista termina o trabalho e o recurso é liberado, o gerente verifica se o recurso pode ser utilizado na próxima regra de produção do plano. Se sim, ele mantém o recurso, caso contrário ele solicita a sua inclusão na lista de espera. O procedimento é repetido para todas as regras seguintes. Com isso, o gerente busca utilizar o recurso que lhe foi dado, enquanto um outro recurso, necessário para a execução de outra regra mais importante não está disponível.

Quando um novo recurso é incluído no sistema, o mesmo atenderá primeiro à fila de espera. Gerentes que perderam mais recursos ao longo da execução são os primeiros

beneficiados. Com a utilização desse protocolo, a alocação de recursos não é sempre realizada de maneira ótima, mas é feita de maneira rápida e envolve pouca troca de mensagens. Portanto, são necessárias apenas duas mensagens básicas: uma pedindo os planos e uma segunda com os resultados dos pedidos.

Futuramente, os recursos poderiam ser diferenciados por seu nível de disponibilidade, desempenho, confiabilidade e proximidade física e lógica na rede. Além disso, a aplicação de outros algoritmos e o estudo de questões de otimização e reescalonamento também seriam muito valiosos.

4.4.2 Cálculos de prioridade e distribuição de recursos

Os diferentes gerentes nos diferentes níveis da arquitetura realizam planejamento diferentes de acordo com as prioridades da organização e a disponibilidade e capacidade dos recursos. A parte inicial do planejamento consiste em definir o conjunto de trabalho composto por alguns dos casos mais importantes. Esta seção apresenta esses cálculos.

O gerente estratégico calcula, baseado nas características dos casos, um valor de prioridade de 1 (prioridade mais baixa) à 5 (prioridade mais alta). O peso de cada uma das características na definição dessa prioridade pode ser ajustado. Em geral, a antiguidade do caso é o fator determinante na prioridade, ou seja, o caso mais antigo é atendido primeiro como em um fila. Em seguida, o gerente estratégico seleciona um conjunto de trabalho C com um número arbitrário dos casos de maior prioridade e calcula a prioridade relativa do caso (RCP) em relação a soma das prioridades desse conjunto de casos (SCP), apresentado na Equação 4.2.

$$RCP(c) = \frac{P(c)}{SCP(C)} \tag{4.2}$$

A função $P(c)$ retorna a prioridade do caso c. O denominador $SCP(C)$ é calculado segundo a Equação 4.3 e representa a soma das prioridades dos casos no conjunto de trabalho C.

$$SCP(C) = \sum_{i=1}^{n} P(c_i), \quad \text{onde } c \in C \tag{4.3}$$

Com o cálculo do RCP, os recursos disponíveis são divididos entre os casos, conforme a Equação 4.4, onde $R(c)$ indica a quantidade de recursos no conjunto R reservados ao caso c.

$$R(c) = RCP(c) \times |R| \tag{4.4}$$

Um gerente tático, ao receber a tarefa de examinar um caso c e um número de recursos $R(c)$ disponível para fazê-lo, tem que realizar cálculo semelhante para selecionar um subconjunto de trabalho E de evidências prioritárias do caso c, com base nas características da evidência. Essa prioridade é calculada a partir dos resultados obtidos nos casos anteriores, que indicam qual tipo de evidência fornece os melhores resultados em um determinado tipo de caso. Após calcular a prioridade da evidência $P(e)$, calcula-se a sua prioridade relativa REP no subconjunto de trabalho utilizando a Equação 4.5.

$$REP(e) = \frac{P(e)}{SEP(E)} \tag{4.5}$$

Na Equação 4.6, $SEP(E)$ representa a soma das prioridades das evidências contidas no subconjunto de trabalho E que contém as evidências prioritárias.

$$SEP(E) = \sum_{i=1}^{n} P(e_i), \quad \text{onde } e \in E \tag{4.6}$$

Após o cálculo do REP de cada evidência do subconjunto de trabalho, o número de recursos R disponíveis para o nível operacional examinar uma evidência é obtido pela Equação 4.7.

$$R(e) = REP(e) \times |R(c)|, \quad \text{onde } c \text{ é o caso a que } e \text{ pertence} \tag{4.7}$$

Como exemplo, considere que o gerente estratégico descubra 40 computadores a sua disposição para realizar os exames e que existam três casos para examinar: (i) um de baixa prioridade, (ii) um com prioridade normal e (iii) um com prioridade muito alta. O gerente estratégico realiza cálculos como os mostrados na Tabela 4.4.

Tabela 4.4: Exemplo de cálculo de recursos para três casos

Caso	**Prioridade**	**RCP**	**Recursos**
Caso 1	LOW (2)	2/10 = 0,2	8
Caso 2	NORMAL (3)	3/10 = 0,3	12
Caso 3	VERY HIGH (5)	5/10 = 0,5	20

Os gerentes táticos responsáveis por cada caso são então informados dos recursos disponíveis. Considerando o gerente tático responsável pelo exame do caso 2, que recebeu 12 computadores de acordo com a Tabela 4.4. O gerente tático realizará os cálculos mostrados na Tabela 4.5.

Tabela 4.5: Cálculos do caso 2 realizados pelo gerente tático.

Caso	**Prioridade**	**REP**	**Recursos**
Evidência 1	NORMAL (3)	3/7 = 0,43	5,14
Evidência 2	NORMAL (3)	3/7 = 0,43	5,14
Evidência 3	VERY LOW (1)	1/7 = 0,14	1,71

Nesse exemplo há um resultado fracionário. O gerente pode aplicar uma política específica para resolver isso. No protótipo desta proposta, as partes fracionárias são acumuladas e depois distribuídas para a evidência de maior prioridade. Nesse caso, as Evidências 1 e 2 receberiam 5 computadores, enquanto a evidência 3 apenas 1. A soma das frações totaliza um recurso que é entregue à Evidência 1.

A definição do tamanho dos conjuntos de trabalho C, de casos, e E de evidências, deve ser ajustada pelo especialista humano com base no número médio de recursos disponíveis no sistema, pois um número muito pequeno de recursos e muito grande de evidências resultaria em recursos muito fracionados. Embora um grande número de evidências estejam

sendo examinados em paralelo, o avanço dos exames de cada evidência em si seria mais lento. O mesmo se aplica ao processamento dos casos.

4.4.3 Monitoramento

Cada plataforma em execução possui um agente de monitoramento que atualiza periodicamente o *blackboard* com informações de recursos disponíveis para o SMA como sua localização, versão do JADE, JVM e sistema operacional, capacidade de memória, disco e CPU, nível de carga e outros. Periodicamente o gerente estratégico verifica o *blackboard* e procura por plataformas que não atualizaram os dados nos últimos 60 segundos.

Essas plataformas são colocadas em uma lista especial. Embora haja uma suspeita de que o recurso foi perdido, o gerente estratégico não é o responsável direto por isso. Um erro no agente de monitoramento, por exemplo, poderia dar a impressão de que o recurso foi perdido, embora o mesmo esteja sendo utilizado. O que acontece nesse caso é que o recurso será utilizado até ser liberado pelo agente especialista. Após ser liberado, o mesmo não será redistribuído. O gerente estratégico deve esperar a comunicação de perda de recurso pelo processo que já foi apresentado anteriormente. Isso permite que o gerente estratégico prepare-se para atender novas requisições de recursos.

A efetiva detecção de perda de um recurso só é feita entre um gerente e um agente subordinado. O gerente recebe notificações de tempos em tempos sobre o andamento das tarefas de seus subordinados. Caso ele não receba essa notificação ou uma mensagem pedindo a dispensa do agente, o gerente verifica o *blackboard* de recursos para determinar se o problema é com a plataforma ou com seu subordinado. Caso não haja registro, o recurso é dado como perdido. Nesse caso, após a comunicação de perda de recurso, o gerente estratégico deve verificar na sua lista de recursos perdidos para confirmar a perda do recurso. Caso exista atualização naquele recurso, significa que o mesmo está disponível, mas o gerente que tem a posse do recurso não está conseguindo utilizá-lo. Nesse caso o recurso será redistribuído. Caso contrário, o recurso será eliminado e o gerente recebe um lugar na fila de espera por um recurso de reposição.

Um novo recurso, por sua vez, é detectado diretamente pela comunicação entre o agente de monitoramento e o gerente estratégico, que assim toma conhecimento da existência de uma novo contêiner JADE. Isso evita que recursos sejam inseridos diretamente no *blackboard*, sem o estabelecimento de um protocolo e a adoção de uma ontologia comum.

4.4.4 Tolerância a falhas

A tolerância a falhas do SMA proposto é baseada nos recursos providos pela plataforma JADE, de replicação do contêiner principal. Na eventualidade de uma falha nesse contêiner, o mesmo pode ser recuperado em outro local. Esse mecanismo não provê tolerância a falhas de funcionamento dos agentes JADE individuais.

No evento de uma falha em um agente do tipo gerente, o procedimento básico realizado pelos agentes subordinados é completar suas tarefas. O superior hierárquico do gerente ausente comunicará aos agentes ociosos que eles estão dispensados e assim os recursos ocupados por estes podem ser liberados. O gerente de nível superior, na ausência de um dos seus subordinados, poderá instanciar um novo, levando em consideração os recursos disponíveis.

O gerente estratégico por ser único é extremamente importante para o sistema. No entanto, sua ausência pode não ser notada por um longo período devido ao procedimento de monitoramento descrito anteriormente. Ou seja, após a atribuição de casos aos gerentes do nível tático, o trabalho do gerente estratégico seria coordenar a execução, principalmente com relação aos recursos. Este estando ausente, não haverá manejo de recursos após o plano inicial. É possível que apenas após o tratamento dos casos designados inicialmente é que se notaria a ausência do mesmo.

Caso o gerente estratégico ou a plataforma em que este se encontra sofra uma falha, o mesmo pode ser instanciado novamente. Para isso, um protocolo de recuperação de comando sobre o nível tático precisa ser executado. O mesmo protocolo pode ser usado nos demais níveis hierárquicos, para que um gerente recupere o comando dos seus subordinados.

O trabalho de Costa (2008) apresenta um mecanismo de monitoramento e recuperação de falhas para o MADIK, no nível dos agentes especialistas.

4.5 Protótipo

Esta seção apresenta detalhes da implementação do protótipo do SMA sob vários aspectos.

Como *framework* para a implementação da plataforma de agentes, foi escolhida a plataforma JADE, apresentada na Seção 3.4.1. JADE implementa a arquitetura proposta pela FIPA para definição de um SMA e atua como um *middleware*, fornecendo serviços de comunicação e localização de agentes, além de ferramentas para a movimentação de agentes entre plataformas. JADE também tem suporte ao uso de ontologias e define diversos protocolos de interação como `Request-Reply`, `Contract-Net`, `Auction`, dentre outros definidos pela FIPA. A implementação do *blackboard* foi feita utilizando o banco de dados `PostgreSQL 8.3` e o *framework* de persistência `Hibernate`.

4.5.1 Projeto

Todos os agentes do SMA proposto são descendentes indiretos da classe `jade.core.Agent`, provida pela plataforma JADE. Algumas características de registro de agentes e mecanismos de interação com o *blackboard* comuns a todos os agentes foram condensadas em uma classe intermediária chamada `BasicAgent`. Assim, todos os agentes do MADIK descendem dessa classe. A Figura 4.9 apresenta um diagrama de classes contendo os principais agentes do SMA proposto e alguns exemplos de agentes especialistas.

Cada um dos níveis da hierarquia possui um tipo de agente. Os agentes nos níveis estratégico, tático e operacional são chamados *gerentes*. Cada um desses níveis, portanto, conta com seu gerente específico, definidos pelas seguintes classes do pacote `madik.agents.managers`:

- `StrategicManager`, o gerente estratégico;
- `TacticalManager`, o gerente tático;
- `OperationalManager`, o gerente operacional.

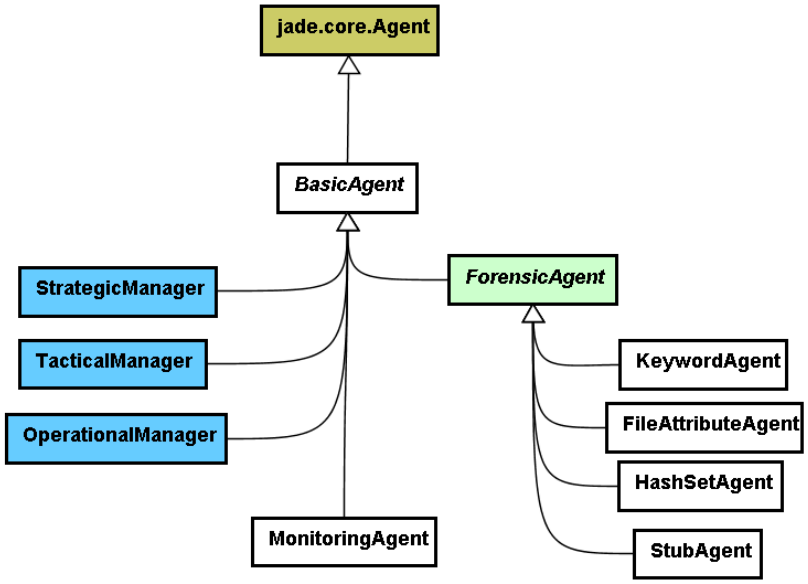

Figura 4.9: Diagrama de classes dos agentes no MADIK

O agente de monitoramento é definido pela classe `MonitoringAgent`. Ele é executado independentemente dos demais e serve como referência para o gerente estratégico dos recursos disponíveis.

Todos os agentes especialistas são baseados na classe *ForensicAgent*, que implemente uma parte comum a todos os especialistas, principalmente no que diz respeito à relação com o gerente operacional. A Seção 4.5.4 apresenta em maior detalhe as classes que implementam especificamente os especialistas.

A Figura 4.10 apresenta de maneira simplificada o modelo de dados utilizado para a gerência da aplicação, para a gerência de casos e planejamento e para o armazenamento e revisão de resultados. Os nomes das colunas foram omitidos para simplificar a visualização. Utilizando a *framework* de persistência `Hibernate`, cada uma dessas tabelas é mapeada para uma classe correspondente no pacote *madik.entities*.

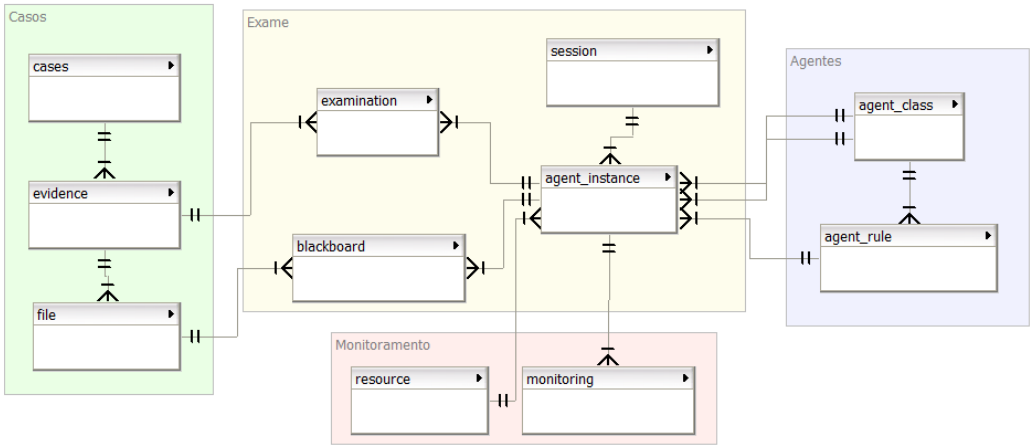

Figura 4.10: Modelo entidade-relacionamento da base de dados do MADIK

4.5.2 Definição da ontologia

A definição de ontologia para uma aplicação de SMA em exames periciais de sistemas computacionais envolve uma grande diversidade de domínios com conceitos criminais, periciais, computacionais e de sistemas multiagente. O escopo da ontologia, portanto, foi limitado neste momento ao nível do especialista, podendo ser expandido futuramente. A Figura 4.11 apresenta os conceitos relacionados aos agentes especialistas definidos. Em destaque na figura, conceitos de arquivo e tipo de arquivo que são utilizados extensamente pelos agentes especialistas.

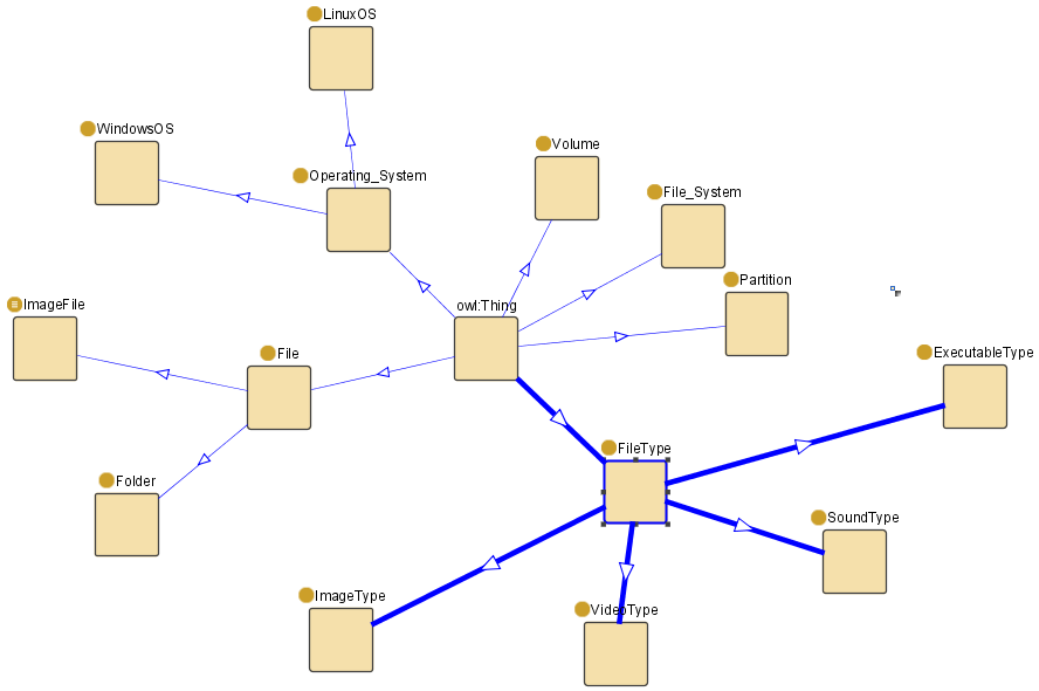

Figura 4.11: Árvore de conceitos da ontologia

Com relação à reutilização de ontologias existentes, a plataforma JADE provê as ontologias relacionadas à gerência da plataforma de agentes, podendo ser integralmente reutilizadas. Conforme discutido na Seção 3.3, ontologias que buscam descrever tudo sobre um domínio específico não são apropriadas para SMA, pois uma ontologia do sistema deve especificar apenas a informação necessária para a execução apropriada do SMA. As ontologias existentes encontradas, apresentadas na Seção 2.3, apresentam nível de detalhe dos conceitos muito distinto do nível do sistema proposto e, por isso, não foram reutilizadas nesse trabalho.

A Figura 4.12 ilustra uma hierarquia das classes contendo algumas instâncias, desenvolvida utilizando o *software* Protégé.

4.5.3 Interface

A camada de apresentação do sistema foi divida em quatro partes, cada uma relacionada a uma atividade distinta. As seções seguintes apresentam essas interfaces.

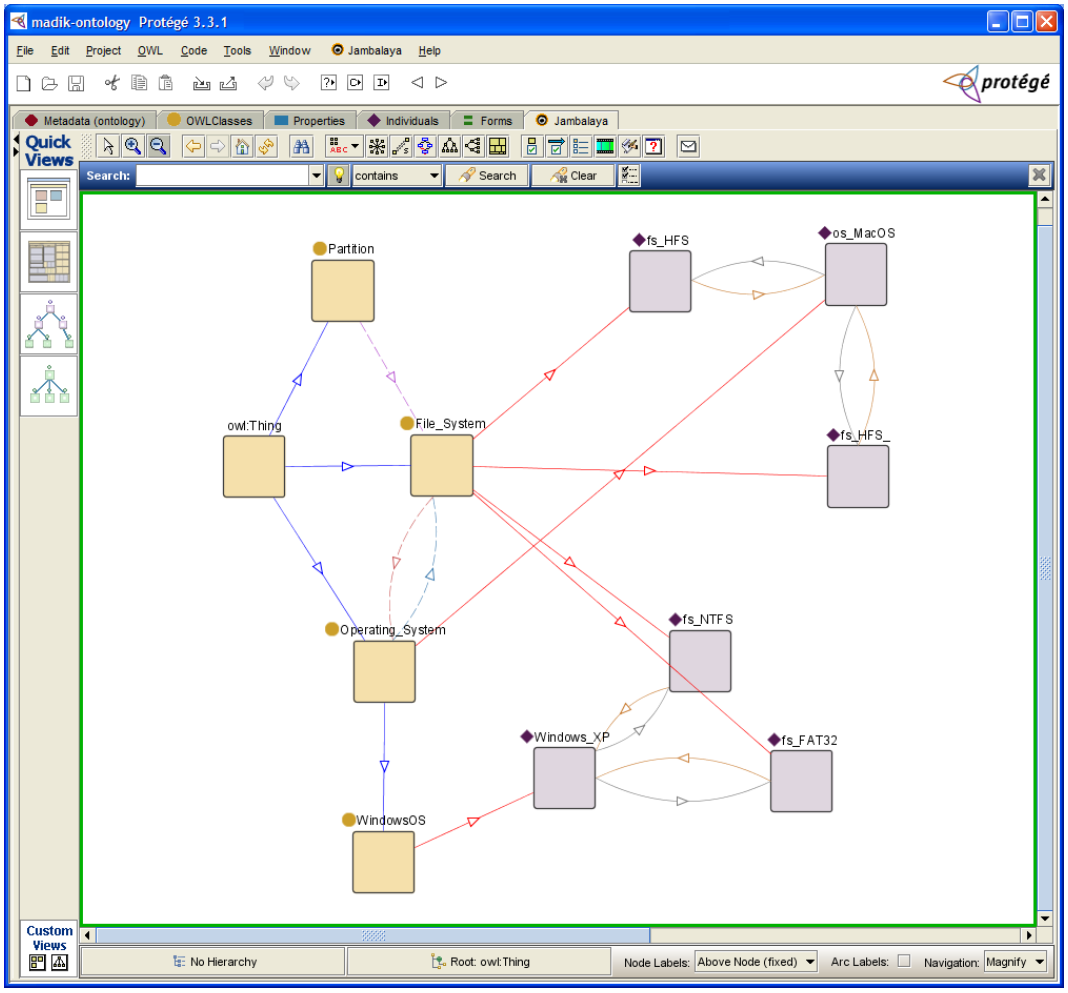

Figura 4.12: Ilustração de alguns conceitos e instâncias da ontologia

Controle

A interface de controle, implementada em Java, permite o controle da plataforma de agentes por meio da interação com os agentes especiais AMS e DF da plataforma JADE. Ela é composta por duas partes, uma associada ao contêiner principal da plataforma JADE a ao gerente estratégico, e a outra associada aos contêineres secundários e seus respectivos agentes de monitoramento. As Figuras 4.13 e 4.14 apresentam, respectivamente, os protótipos de cada uma das interfaces.

A interface de controle principal exibe a hierarquia dos agentes em execução e sua localização nos contêineres da plataforma JADE. A interface do componente de monitoramento apresenta apenas as informações coletadas pelo agente de monitoramento no contêiner em que se encontra.

Monitoramento

Esta é uma interface *web*, desenvolvida utilizando-se a *framework* Ruby on Rails, destinada ao monitoramento dos trabalho da plataforma. Ela apresenta as informações coletadas pelos agentes de monitoramento em cada uma das plataformas do sistema, bem como

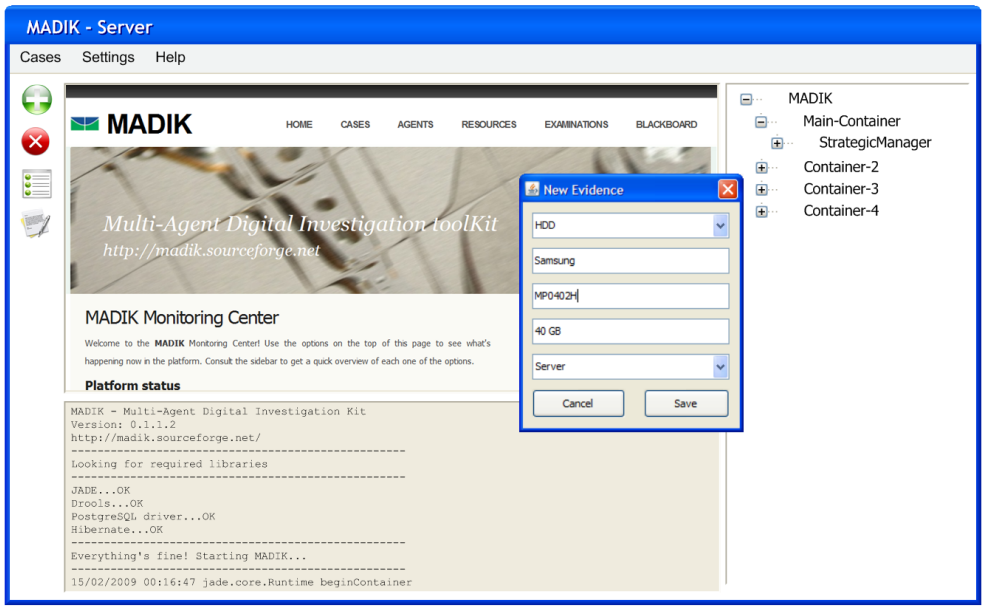

Figura 4.13: Interface de controle principal

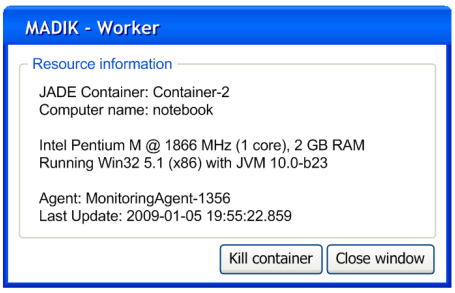

Figura 4.14: Interface do componente de monitoramento

detalhes do planejamento. A interface não se destina ao controle de qualquer aspecto da plataforma, sendo utilizado para isso a interface específica de controle. As Figuras 4.15 e 4.16 apresentam duas telas da interface de monitoramento.

Revisão

Conforme descrito anteriormente, após o trabalho dos agentes em um caso, inicia-se a fase de revisão. Nessa fase, o especialista humano pode acessar e revisar cada uma das recomendações dos agentes, concordando com a recomendação final sugerida pelo gerente operacional ou corrigindo-a. A presença de conflitos na decisão dos agentes é destacada por um ícone de exclamação amarelo, indicando que a atenção do especialista é necessária. Em casos em que uma recomendação do tipo `alertar` foi feita, um ícone de exclamação vermelho é exibido, indicando que esses itens devem ser revisados primeiro.

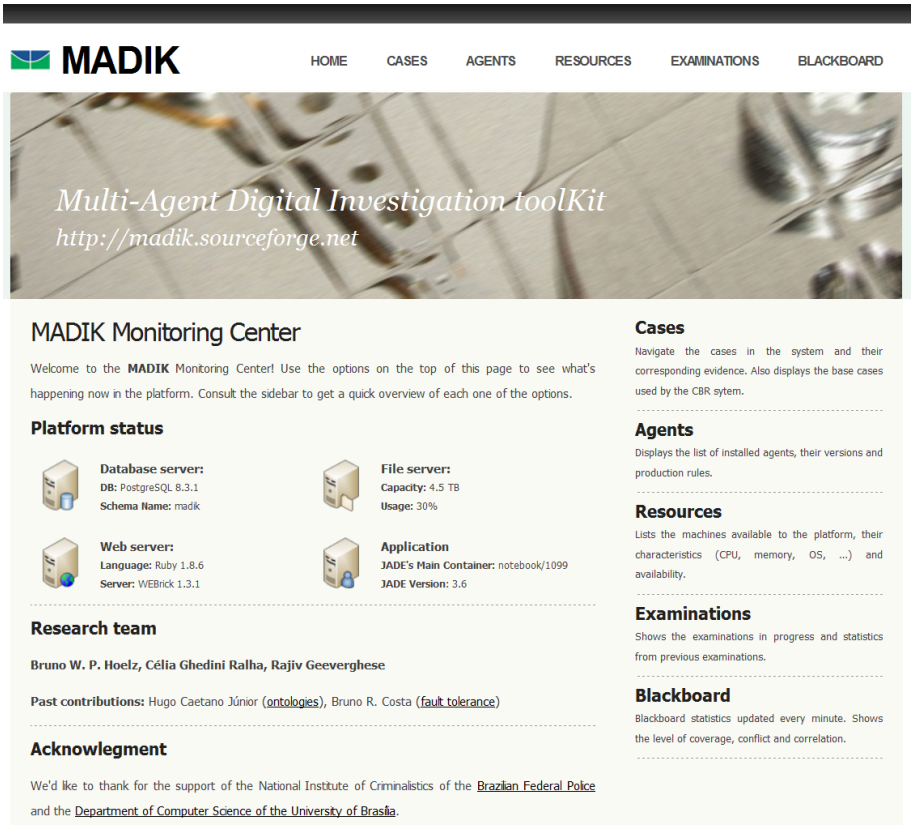

Figura 4.15: Tela inicial da interface de monitoramento

4.5.4 Agentes especialistas

Os agentes especialistas são os responsáveis pelo exame em si dos arquivos contidos nas evidências de um caso. Para isso, utilizam uma ou mais regras de produção. As regras podem ser definidas em termos de código Java, consulta na linguagem *Structured Query Language* (SQL) ou utilizando o motor de inferência `Drools` (`http://www.jboss.org/drools/`). Os agentes especialistas podem também trabalhar como *wrappers* de ferramentas forenses existentes, controlando suas entradas, interpretando suas saídas e colocando suas recomendações no *blackboard*, de forma semelhante ao sugerido por Alinka et al. (2006).

No protótipo implementado as regras de produção foram direcionadas para exames em discos rígidos contendo o sistema operacional Microsoft Windows XP, que corresponde à grande maioria da demanda atual de exames periciais na PF. A seções seguintes apresentam os agentes implementados e uma série de regras de produção propostas para eles.

As regras implementadas são simples e de aplicação ampla, com o objetivo de demonstrar a viabilidade do SMA proposto. Considerando a variedade de exames e situações possíveis, a definição e implementação de novos agentes e de novas regras de produção para os agentes existentes é um objetivo contínuo. A Seção 2.3 apresenta vários estudos de exames periciais específicos que futuramente podem ser agregados a esta proposta.

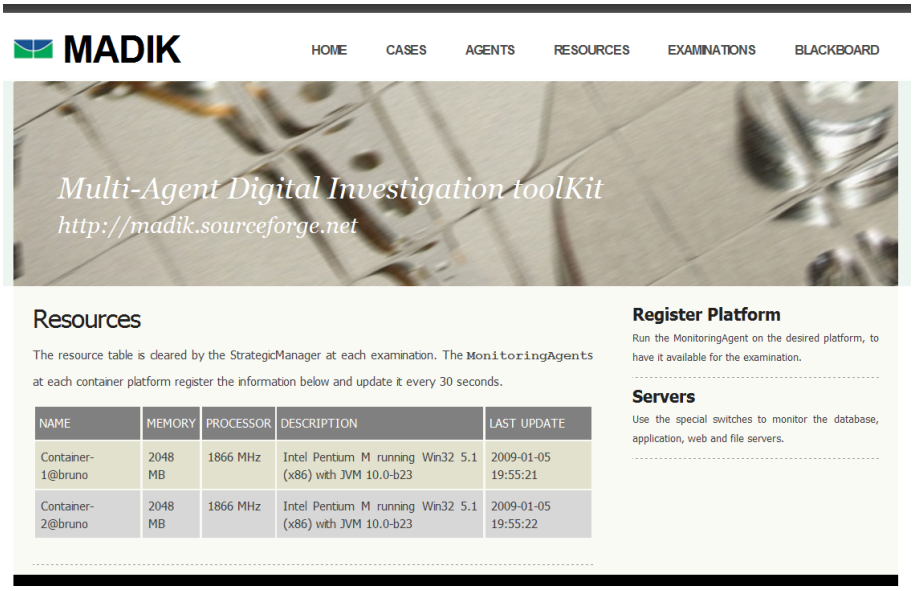

Figura 4.16: Listagem de recursos na interface de monitoramento

HashSetAgent

Este agente utiliza uma base de *hashes* MD5 de arquivos conhecidos, que auxilia na identificação de arquivos de interesse para o caso ou na eliminação de arquivos reconhecidamente irrelevantes para a investigação, conforme discutido na Seção 2.3. A seguir são apresentadas algumas das regras desse agente.

- `EmptyFile`: identifica arquivos vazios;
- `DuplicateFile`: identifica arquivos duplicados;
- `KnownSystemFile`: identifica arquivos conhecidos do sistema operacional Microsoft Windows;
- `KnownP2PFile`: identifica arquivos conhecidos de programas P2P;
- `KnownMalicious`: identifica arquivos maliciosos como cavalos-de-tróia ou vírus;
- `KnownChildPorn`: identifica arquivos conhecidos de pornografia infantil;
- `KnownApplication`: identifica aplicativos conhecidos, que são amplamente instalados
- `UnknownFrequentFile` (regra de correlação): identifica arquivos comuns a várias evidências que não são arquivos conhecidos; isso pode indicar arquivos que foram transferidos entre os suspeitos;
- `CustomSet`: regra que utiliza um conjunto pré-definido de valores de *hash* para identificar arquivos de interesse.

Ao contrário de abordagens que comparam todos os valores de *hash* conhecidos com todos os arquivos, o `HashSetAgent` procura aplicar apenas a regra mais adequada à natureza do caso. Em um caso de fraude financeira, por exemplo, a regra `KnownP2PFile` é

utilizada para ignorar arquivos, enquanto em um caso de pornografia infantil é utilizado para identificar o possível uso desses programas para envio e recebimento de material pornográfico envolvendo menores de idade. A Figura 4.17 ilustra o funcionamento do `HashSetAgent`.

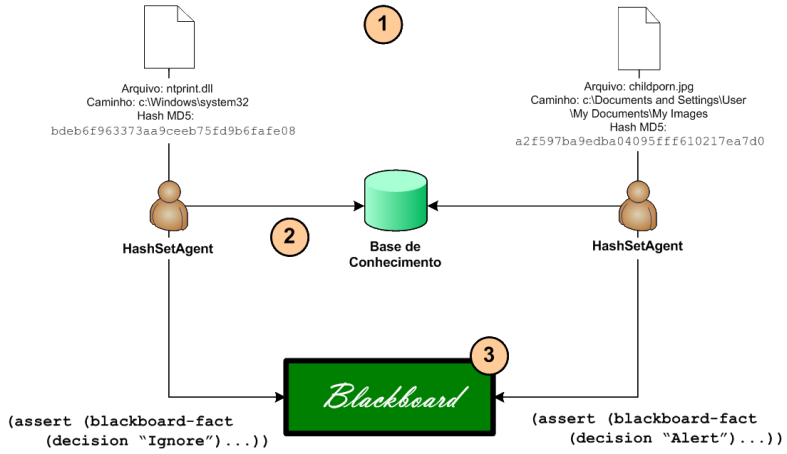

Figura 4.17: Funcionamento do HashSetAgent

TimelineAgent

Este agente analisa as datas de criação, acesso e modificação de arquivos para identificar eventos no sistema de arquivo como a instalação de novos programas, atualização do sistema, criação de arquivos de usuário ou navegação na Internet. Na grande maioria dos casos, muitos desses eventos não são relevantes para a investigação. A eliminação, portanto, dos arquivos envolvidos reduz em grande quantidade o volume de material a ser examinado.

- `RecentlyModified`: identifica arquivos relacionados à atividade do usuário como, por exemplo, arquivos da pasta "Meus documentos" do sistema operacional Microsoft Windows XP, modificados dias antes da apreensão do computador;
- `RecentlyCreated`: identifica arquivos relacionados à atividade do usuário criados nos últimos dias de uso do computador;
- `ApplicationInstall`: identifica pontos de instalação de programas, que representam uma concentração de arquivos criados em um mesmo diretório;
- `SystemUpdate`: identifica pontos de atualização do sistema, que implicam em uma grande modificação de arquivos do sistema operacional;
- `CustomPeriod`: identifica arquivos compreendidos em um ou mais períodos pré-determinados.

FileAttributeAgent

Este agente examina atributos diversos dos arquivos como seu nome, localização e tamanho. Com isso pode identificar grupos de arquivos que podem ser ignorados ou que são de interesse para o caso. Nos estudos de caso com o protótipo inicial do SMA, este agente era chamado `FilePathAgent`.

- `SmallInternetImage`: identifica imagens de Internet pequenas;
- `TaxRelated`: identifica arquivos de declaração de impostos;
- `Cookies`: identifica arquivos de *cookie* do navegador de Internet;
- `BigMultimidia`: identifica arquivos multimídia de tamanho grande;
- `MailMessage`: identifica arquivos relacionados a *e-mails*;
- `UserFile`: extrai planilhas e documentos das pastas de usuário do sistema;
- `SourceCode`: arquivos relacionados a códigos-fonte de programas em várias linguagens de programação;
- `MailExchange` (correlação): identifica troca de *e-mails* entre evidências.

FileSignatureAgent

Este agente compara as assinaturas de tipos de arquivos com a extensão encontrada. Com isso ele é capaz de encontrar arquivos cuja extensão tenha sido modificada para ocultar o propósito real do arquivo. Além disso, ele registra cabeçalhos e extensões desconhecidas para revisão do especialista. Com isso, o agente pode associar determinadas extensões ou cabeçalhos de interesse aos diversos tipos de casos, ignorando as demais extensões. Utiliza também regras para identificar padrões de nomenclatura dos arquivos (prefixos, sufixos e numerações sequenciais).

- `BadSignature`: a extensão do arquivo não corresponde ao seu cabeçalho;
- `UnknownSignature`: a extensão e o cabeçalho são desconhecido;
- `DigitalPhoto`: arquivo com prefixo característico de fotografia digital;
- `HomeBanking`: identifica arquivos associados à sítios de serviços bancários na Internet;
- `Webmail`: identifica arquivos relacionados à serviços de correio eletrônico pelo navegador de Internet.

KeywordAgent

Este agente é capaz de procurar por ocorrências de padrões de bytes ou palavras em arquivos utilizando codificações diversas (`ASCII`, `UTF-8`, `Unicode`).

- `GenericWordlist`: busca uma lista de palavras genéricas com relação ao tipo de crime investigado; exemplo: nomes e gírias para drogas em uma caso de tráfico de drogas;

- `FeatureExtraction`: extrai informações características de arquivos tais como números de CPF e endereços de *e-mail*;
- `RegexSearch`: realiza uma busca segundo uma expressão regular pré-definida;
- `CustomKeyworkList`: realiza um pesquisa por palavras-chave fornecidas especificamente para o caso.

WindowsRegistryAgent

Este agente é capaz de examinar os arquivos de registro do sistema operacional Microsoft Windows, extraindo o valor das chaves e examinando-os segundo suas regras de produção. Embora as regras implementadas não utilizem nenhuma informação além das existentes nos arquivos de registro, o agente poderia acessar quaisquer outros arquivos do caso para corroborar suas descobertas.

- `UserActivity`: obtém informações de buscas realizadas na Internet, programas executados pelo usuários, dispositivos removíveis utilizados (como *pen drives*) e outras informações de utilização do sistema;
- `UserPasswords`: obtém senhas de correio eletrônico, sítios na Internet e outras armazenadas no registro;
- `SystemInformation`: extrai informações do registro como data de instalação e versão do sistema;
- `ApplicationConfiguration`: extrai configurações de aplicativos específicos armazenadas no registro.

A Figura 4.18 apresenta alguns dos componentes necessários à implantação do SMA. Os principais componentes são:

- servidor de aplicação principal para abrigar o contêiner principal da plataforma JADE e a interface de monitoramento;
- servidor de banco de dados para armazenar os dados do caso e os resultados do *blackboard*;
- servidor que fornece acesso ao conteúdo completo das evidências;
- computadores para utilização como contêineres secundários, onde os demais agentes trabalham;

4.5.5 Implantação

Além desses componentes, é necessário um elemento de integração com uma ferramenta pericial que realize o processamento inicial das evidências. Para isso, uma solução inicial é fornecida pela classe auxiliar `FTKUtil` que exporta os dados de uma caso processado com a ferramenta *Forensic ToolKit* (FTK), para que possa ser usado pelos agentes do MADIK. Para o acesso ao conteúdo integral das evidências, pode ser utilizada outra ferramenta forense, o Guidance EnCase (`http://www.guidancesoftware.com/products/ef_index.asp`), com o uso do módulo *Physical Disk Emulator* (PDE).

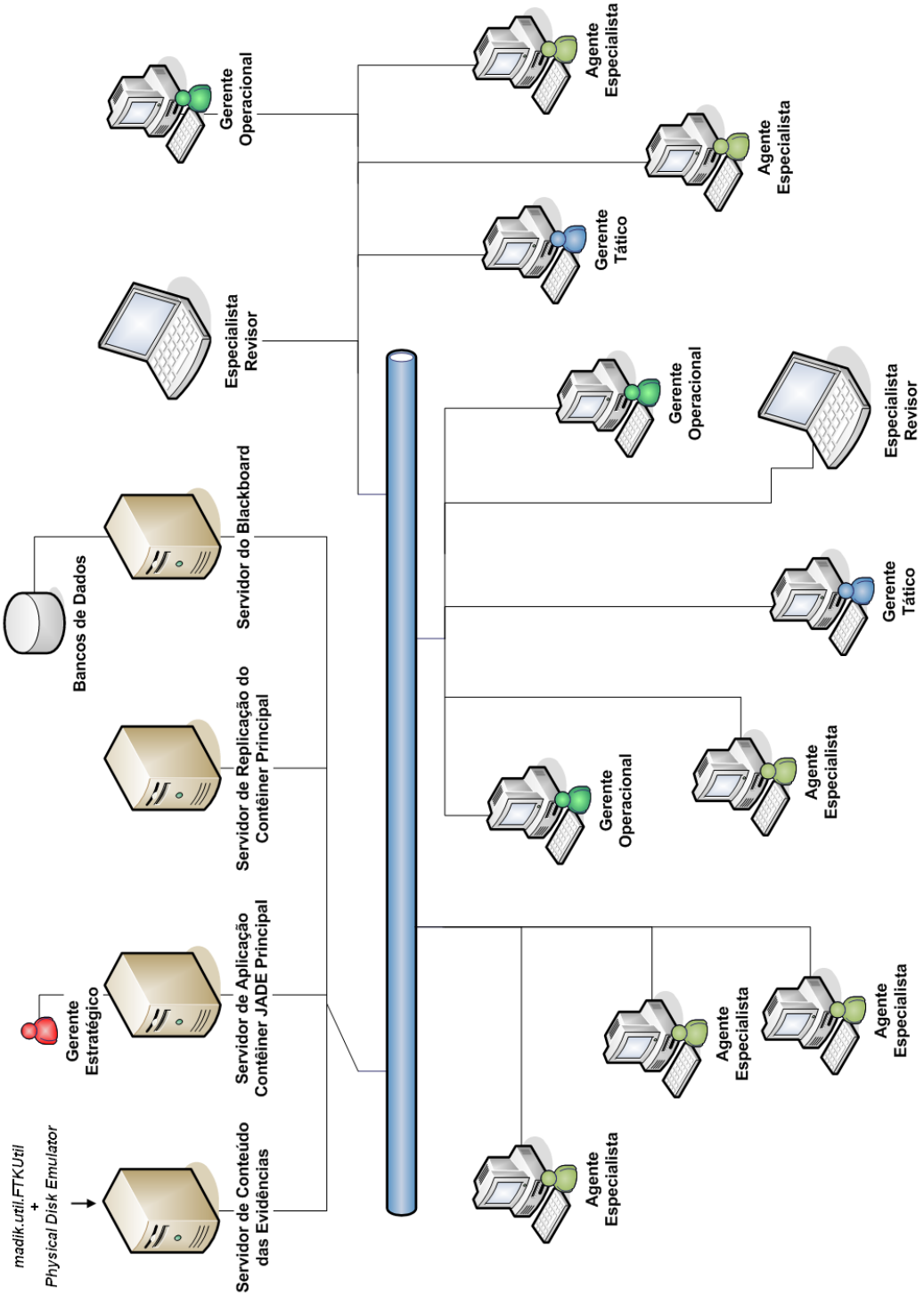

Figura 4.18: Implantação do MADIK

Capítulo 5

Experimentos e Resultados

Durante o desenvolvimento do MADIK, vários experimentos foram realizados para avaliar a viabilidade e o desempenho do protótipo por meio de uma abordagem incremental. Este capítulo apresenta esses experimentos e seus respectivos resultados, avaliando sempre o impacto sobre a evolução do protótipo. Serão apresentados também estudos de caso utilizando o protótipo em casos reais com volumes significativos de evidências.

5.1 Experimento 1

O primeiro protótipo do SMA proposto foi implementado com o intuito de avaliar a relação entre dois níveis hierárquicos (operacional e especialista), avaliando os custos de comunicação, a escalabilidade, o nível adequado de coordenação das atividade e os ganhos de distribuição. Apenas um tipo de especialista foi implementado, o `HashSetAgent`, cujo único objetivo era identificar arquivos conhecidos que pudessem ser ignorados (como arquivos da instalação padrão do sistema operacional) ou que necessitassem de atenção especial do especialista (como imagens contendo pornografia infantil). Os três primeiros experimentos a seguir foram realizados em um único contêiner JADE em execução em uma máquina isolada.

No primeiro experimento, testou-se um número variado desses agentes sob o comando de um gerente operacional. O gerente divide a tarefa de examinar os arquivos de um disco rígido entre os especialistas. Para isso, o gerente divide o conjunto total de arquivos em blocos. Em seguida, cada agente recebe um bloco para examinar. Ao terminar o exame do bloco, o agente comunica ao gerente que terminou e se ainda existirem blocos pendentes, ele receberá mais um bloco. O processo é repetido até que todos os blocos tenham sido examinados.

Nesse protótipo, um agente especial chamado `BlackboardAgent` trabalha como elemento de controle do *blackboard*, de forma que toda informação escrita a ser escrita no *blackboard* passa por ele. O `BlackboardAgent` utiliza o motor de inferência Jess (Hill, 2003) para armazenar as informações do *blackboard* na forma de fatos na sua memória de trabalho. Nesse modelo, consultas ao *blackboard* também são solicitadas a esse agente, que realiza o raciocínio e coloca os resultados no próprio *blackboard*.

A Tabela 5.1 apresenta o tempo médio de execução do primeiro experimento com esse protótipo. Nele, apenas um bloco é examinado por agente, ou seja, o tamanho total dos arquivos é dividido igualmente entre os agentes. O número total de arquivos é de 50 mil.

Portanto, no cenário com apenas um agente especialista, este examinará todos os arquivos sozinhos, com dois agentes, cada um examinará 25 mil arquivos e assim por diante. O objetivo desse experimento era verificar os impactos da distribuição de trabalho no tempo total de exame e na coordenação pelo gerente operacional.

Tabela 5.1: Tempo de execução com um bloco por agente

Número de Agentes	**Tamanho de Bloco**	$\mathbf{T}_{min}$	$\mathbf{T}_{max}$
1	50.000	50s	50s
2	25.000	40s	42s
4	12.500	29s	43s
8	6.250	28s	45s
16	3.125	5s	48s

Pode-se notar na Tabela 5.1 que nesse cenário com muitos agentes examinando apenas um bloco de tamanho igual, há uma tendência de reduzir o tempo mínimo T_{min}, enquanto o tempo máximo não decresce na mesma proporção. Isso ocorre, pois um agente termina seu trabalho bem mais cedo que os demais, devido à natureza do próprio exame. Se um `HashSetAgent` recebe um bloco em que poucos arquivos estejam em sua base de arquivos conhecidos, este não precisará enviar muitas mensagens para o `BlackboardAgent`. Essa situação indica que um gerente poderia aproveitar esse tempo ocioso dos agentes com a melhor divisão dos blocos. A Figura 5.1 apresenta visualmente os resultados obtidos.

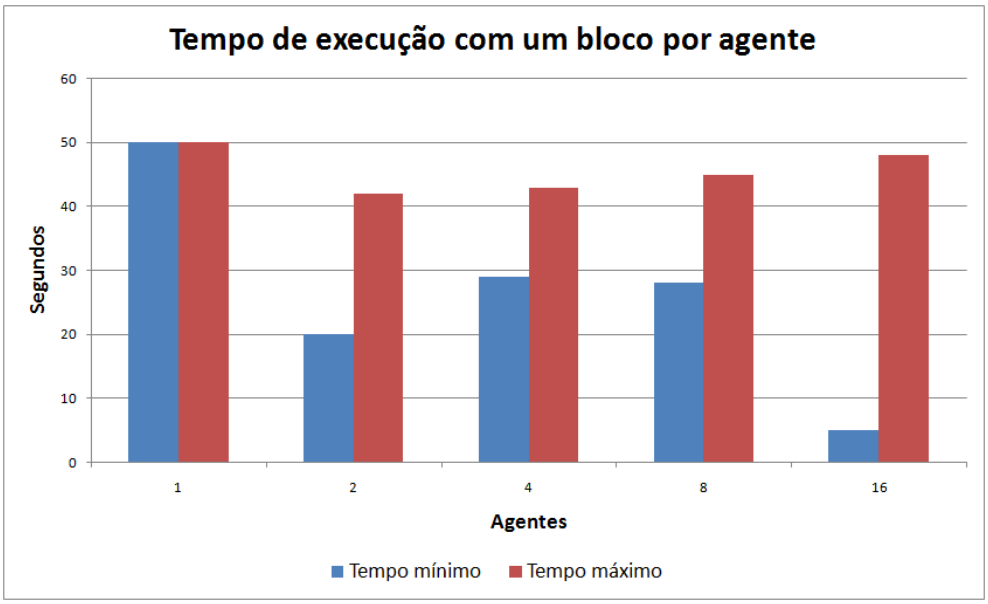

Figura 5.1: Tempo de execução com um bloco por agente

5.2 Experimento 2

No segundo experimento, utilizou-se um tamanho de bloco pequeno com um número variado de agentes especialistas `HashSetAgent`. A Tabela 5.2 apresenta os resultados desse

experimento. Além de observar ganhos com relação à melhor distribuição do trabalho e uso dos recursos, buscou-se observar o impacto do aumento do número de blocos e, consequentemente, do número de mensagens trocadas entre o gerente operacional e o agente especialista, sobre o desempenho do sistema. Isso porque, quanto menor o tamanho do bloco, maior o número de mensagens trocadas no sistema. No protocolo de comunicação utilizado, para cada "diálogo" entre gerente e agente, três mensagens foram utilizadas:

1. gerente atribui um bloco a um `HashSetAgent`;
2. `HashSetAgent` informa o gerente do término do trabalho e solicita nova tarefa;
3. o gerente atribui um novo bloco ao especialista ou o dispensa, caso não haja mais trabalho.

Tabela 5.2: Tempo de execução com fixação do tamanho do bloco

Número de Agentes	**Tamanho de Bloco**	T_{min}	T_{max}
1	3.125	58s	58s
4	3.125	28s	42s
8	3.125	29s	40s

Pode-se observar que o tempo utilizado na troca de mensagens afeta significativamente o tempo de execução do sistema. Com oito agentes e um tamanho de bloco de 3.125 arquivos, os resultados foram um pouco melhores, considerando o tempo máximo T_{max}, do que com quatro agentes, mas não houve ganho significativo com relação ao T_{min}.

A Figura 5.2 ilustra os resultados do segundo experimento, combinado ao resultado obtido no primeiro experimento com o uso de 16 agentes, que também examinaram blocos de 3125 arquivos.

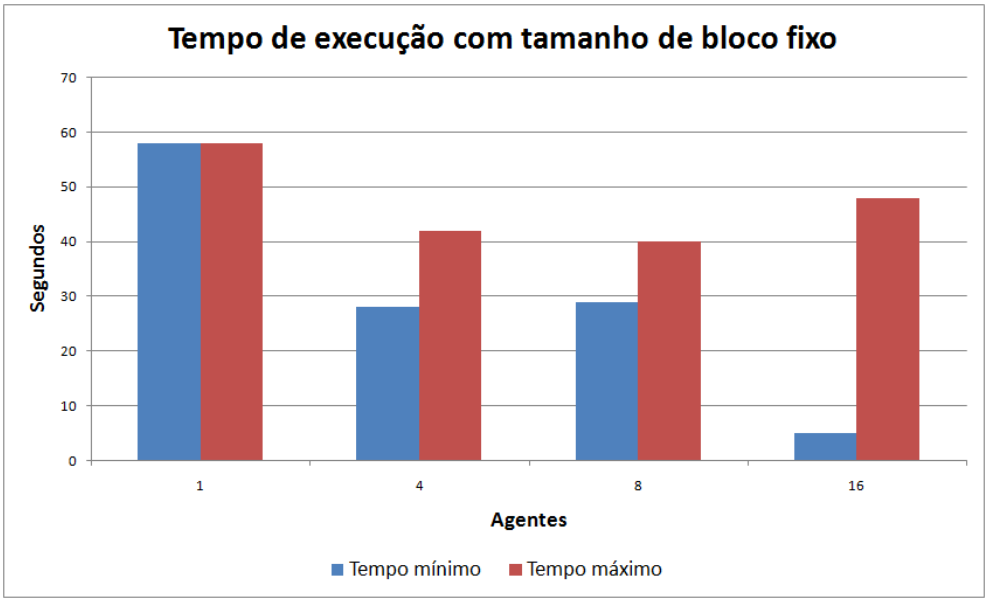

Figura 5.2: Tempo de execução com tamanho de bloco fixo

Esses resultados mostram que apesar do tempo mínimo registrado com 16 agentes ter sido bem menor que dos demais, o tempo máximo foi superior ao cenário com quatro e oito agentes, o que sugere que a quantidade de oito agentes é ideal para blocos de 3125 arquivos.

O impacto da troca de mensagens pode ser visto mais claramente comparando-se o tempo gasto utilizando apenas um agente. Na Tabela 5.1, o agente examina 50.000 arquivos em 50 segundos, enquanto na Tabela 5.2, examina os 3125 arquivos em 58 segundos. No primeiro caso, o agente recebe apenas um bloco para exame e troca 3 mensagens com o seu gerente, enquanto no segundo, recebe 16 blocos e troca 48 mensagens.

Os resultados mostram a necessidade de controlar dinamicamente o número de agentes e o tamanho dos blocos para melhorar o desempenho. Para comparar os resultados obtidos pelo especialista, os mesmos dados foram processados utilizando a ferramenta forense FTK 1.70.1. A base de arquivos conhecidos utilizada pelo programa nesses testes também foi a mesma utilizada pelo agente especialista `HashSetAgent`. O tempo médio de execução da ferramenta foi de 58 segundos, o mesmo obtido com apenas um agente especialista e blocos de 3125 arquivos. Com oito agentes, o resultado obtido de 40 segundos apresenta uma redução de aproximadamente 32% no tempo total necessário para realizar essa análise, o que demonstrou a viabilidade da aplicação de uma SMA para o exame de sistemas periciais.

5.3 Experimento 3

Em um terceiro experimento, semelhante ao primeiro, um número variado de agentes especialistas examina apenas um bloco, porém nesse caso com um tamanho de bloco de 160.000 arquivos. A Tabela 5.3 apresenta o tempo de execução desse experimento. Note-se que o tempo mínimo T_{min} de execução diminui abruptamente de 64 segundos com um agente para cinco segundos com quatro ou oito agentes e para seis segundos com 16 agentes. É interessante notar que o tempo máximo T_{max} aumenta de 66 para 92 segundos, causado pelo aumento nos custos de comunicação entre os agentes especialistas e o agente do *blackboard*. Como mais arquivos foram analisados, mais mensagens foram trocadas entre esses agentes. O desvio padrão observado variou de 0,5 a dois segundos.

Tabela 5.3: Tempo de execução com um bloco grande por agente

Número de Agentes	**Tamanho de Bloco**	T_{min}	T_{max}
1	160.000	64	66
2	80.000	49	50
4	40.000	5	75
8	20.000	5	80
16	10.000	6	92

Em comparação com o primeiro experimento, observou-se que os tempos não aumentaram na mesma proporção que o tamanho do caso. O primeiro experimento demorou relativamente mais que o terceiro, embora o tamanho do terceiro fosse mais que três vezes maior. A Figura 5.3 ilustra essa diferença. Os tempos do terceiro experimento foram

ajustados proporcionalmente com relação ao primeiro experimento, ou seja, os tempos apresentados na Tabela 5.3 foram reduzidos na proporção entre os tamanhos de bloco nos dois experimentos.

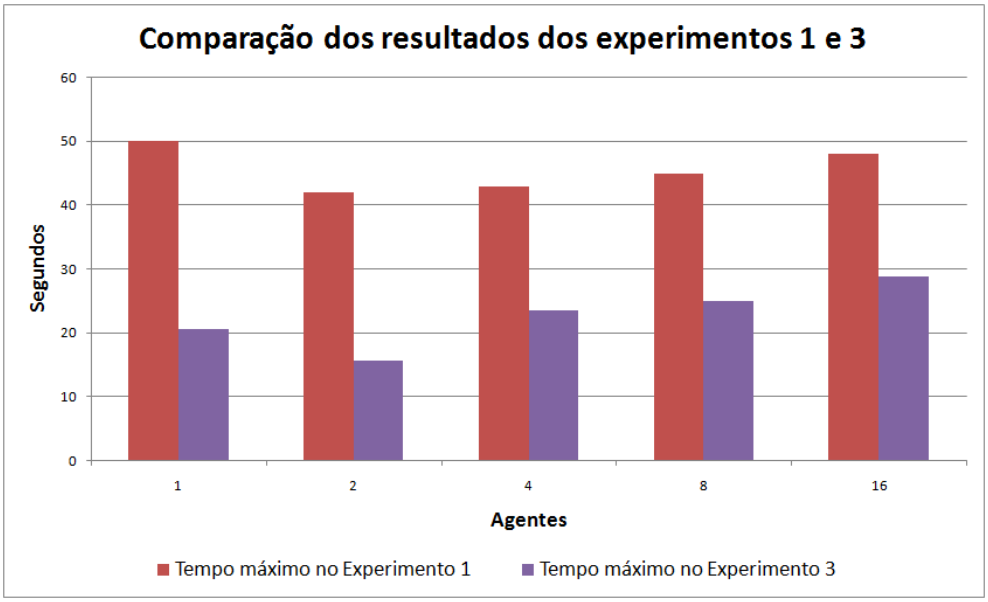

Figura 5.3: Comparação dos resultados dos experimentos 1 e 3

Uma preocupação surgiu com relação ao `BlackboardAgent`, que poderia representar um gargalo no acesso ao *blackboard*, caso o tamanho e o número de agentes especialistas aumentassem muito. Modificações foram então realizadas no protótipo, eliminando o `BlackboardAgent` como intermediário entre os especialistas e o *blackboard*. Com isso, o *blackboard* passou a residir em uma base de dados PostgreSQL, com acesso livre aos especialistas para leitura e escrita.

Foram realizados novos experimentos, agora em um ambiente distribuído composto por quatro máquinas e utilizando uma conexão sem fio em rede no padrão 802.11g. O tamanho total de arquivos utilizado nesse experimento foi de 500 mil arquivos. Foram realizados testes com um tamanho fixo de bloco de 2.000 arquivos, variando o número de agentes e de máquinas. O tamanho de bloco pequeno tinha por objetivo igualar o tempo total de execução dos agentes nas máquinas e observar as perdas resultantes dos esforços de coordenação do gerente operacional.

A Tabela 5.4 apresenta os resultados obtidos. A coluna `Ag./Comp.` indica o número de agentes por máquina. Apenas o tempo médio T_{med} e o desvio padrão são mostrados. Os resultados foram encorajadores com relação aos ganhos de distribuição. A redução no tempo de execução T_{med} de uma para quatro máquinas é muito significativa, de mais de 3,6 vezes (de 354,67 para 96,67 segundos). Observou-se, no entanto, que os ganhos de se colocar vários agentes na mesma máquina não são proporcionalmente significativos. Assim, caso um recurso seja bem aproveitado por um agente especialista, não há ganho evidente em colocar mais agentes no mesmo recurso.

A Figura 5.4 demonstra mais claramente que os ganhos de distribuição seguem o esperado, mas que o aumento no número de agentes em uma mesma plataforma não apresenta ganhos significativos de desempenho. Nesse experimento, cerca de 10% do

Tabela 5.4: Tempo de execução com tamanho de bloco fixo em ambiente distribuído

Computadores	**Agentes**	**Ag./Comp.**	T_{med}	**desvio**
1	1	1	354,67	3,05
1	2	2	348,33	2,00
1	4	4	345,33	4,16
1	8	8	342,00	0,58
2	2	1	191,67	1,53
2	4	2	186,33	4,00
2	8	4	184,67	0,58
2	16	8	182,00	2,09
4	4	1	98,67	0,58
4	8	2	98,00	0,00
4	16	4	96,67	0,58

desempenho é perdido com o aumento do número de máquinas, com relação ao ganho esperado. Dobrando-se a quantidade de máquinas, portanto, não se obtém um tempo de execução 50% menor que o original, e assim por diante. Portanto, há um limite no número de recursos e agentes que um gerente operacional pode coordenar. A decisão de adotar uma organização hierárquica no SMA decorre da possibilidade de dividir os custos de coordenação entre vários gerentes, permitindo que um gerente em um nível mais alto coordene muito mais atividades do que ele seria capaz sozinho.

Figura 5.4: Resultados da execução distribuída

Após esses experimento, determinou-se que os agentes especialistas receberiam um recurso dedicado para desenvolver suas tarefas. Para reduzir os custos de coordenação, os gerentes operacionais não mais precisariam controlar a distribuição dos blocos de tarefas em um nível grande de detalhe (como determinar um tamanho ótimo de bloco). Eles apenas deveriam decidir quantos agentes empregar e dividir o total de arquivos entre estes agentes. Isso porque mesmo que um agente termine sua tarefa muito antes de outros, em um contexto maior, o recurso será rapidamente reutilizado pelo gerente, seja

Tabela 5.5: Cálculos de prioridade de um conjunto de casos

Caso	Prioridade	RCP	Recursos
Caso 1	NORMAL (3)	3/10 = 0,3	1,2
Caso 2	NORMAL (3)	3/10 = 0,3	1,2
Caso 3	HIGH (4)	4/10 = 0,4	1,6

pelo próprio agente especialista, por outro tipo de agente ou até mesmo cedido para outro caso.

Quatro novos agentes especialistas foram concebidos e para testá-los foram utilizados dados de uma investigação real, que continha 110 mil arquivos em dois discos discos rígidos. Foram introduzidos os cálculos de prioridade apresentados na Seção 4.4.2. Para realizá-los, foram consideradas casos e evidências adicionais fictícios.

Os quatro agentes especialistas utilizados foram: (i) `HashSetAgent`, (ii) `FilePathAgent`, (iii) `FileSignatureAgent` e (iv) `TimelineAgent`. O código a seguir apresenta uma regra de produção com a sintaxe Jess do `HashSetAgent` para identificação de um arquivo conhecido.

```
(defrule known-file-filter
  "Known file filter"
  ?f <- (file)
  ?kf <- (known-file { hash == f.hash }
  =>
  (assert (blackboard-fact
    (id ?f.id)
    (agent ?f.agent)
    (decision ?kf.decision)
    (description ?kf.description)))
  (printout t "Known file with status "
  ?kf.decision " inserted in the blackboard (" ?kf.description ")" crlf))
```

Para examinar os arquivos, quatro recursos foram registrados no *blackboard*. Assim, o gerente estratégico determina a existência de três casos pendentes e quatro recursos disponíveis. As prioridades dos casos e das evidências nesse teste foram atribuídas arbitrariamente. A Tabela 5.5 apresenta o uso das Equações 4.2, 4.3 e 4.4. O caso três recebe 2 recursos, enquanto os demais casos recebem apenas um. As partes fracionárias foram entregues ao caso com maior prioridade.

Acompanhando, por exemplo, o trabalho do gerente tático do caso 3, este recebe a atribuição de examinar caso 3 e também a informação de que possui dois recursos à disposição. O gerente busca então na base de dados os dados das evidências relacionadas ao caso 3. Ele encontra quatro evidências, para as quais realiza os cálculos de prioridade, presentes na Tabela 5.6.

Nesse caso, como os recursos são bastante limitados, nenhuma evidência teria direito a sequer um dos recursos. Com a política de distribuição dos valores fracionários empregada nesse teste, apenas a Evidência 4 seria examinada, utilizando os dois recursos disponíveis. Esse resultado sugeriu a necessidade de definir no nível estratégico o número de casos e evidências que devem ser examinadas ao mesmo tempo. Embora direcionar todos os

Tabela 5.6: Cálculos de prioridade de um caso

Evidência	Prioridade	REP	Recursos
Evidência 1	LOW (2)	2/14 = 0,143	0,286
Evidência 2	NORMAL (3)	3/14 = 0,214	0,428
Evidência 3	HIGH (4)	4/14 = 0,286	0,572
Evidência 4	VERY HIGH (5)	5/14 = 0,357	0,714

recursos do sistema para a evidência mais importante do caso mais importante seja uma política coerente, nem sempre ela é viável, tendo em vista que um caso muito grande em quantidade de evidência e de grande importância impediria a resolução de outros casos durante o longo tempo em que for examinado.

O gerente operacional então recebe uma mensagem do gerente tático apontando qual evidência ele deve examinar e a lista dos recursos disponíveis. O gerente operacional cria os agentes desejados nos recursos disponíveis. Nesse caso, o `HashSetAgent` e o `FilePathAgent` são criados primeiro. Após o término do trabalho desses agentes, os outros dois agentes são criados.

O gerente operacional envia comandos para os especialistas examinarem blocos de 5.000 arquivos. É importante observar que alguns agentes como o `TimelineAgent` não trabalham com a noção de blocos, pois examinam o sistema como um todo. Os blocos nesse caso prejudicam suas recomendações. O gerente deve estar ciente dessa peculiaridade, para passar apenas um bloco com o tamanho total da evidência para o especialista. Durante a execução, os especialistas inserem suas recomendações e observações no *blackboard*, conforme definido na Seção 4.3.

O código a seguir apresenta a definição no Jess dos fatos inseridos no *blackboard*.

```
(deftemplate blackboard-fact
"Estrutura que armazena a recomendação dos agentes"
  (slot id (type INTEGER))
  (slot agent)
  (slot recommendation)
  (slot description)
  (slot time-taken))
```

O código a seguir apresenta uma instância do `blackboard-fact` apresentado anteriormente ao arquivo `mmsystem.dll`, que possui o código identificador `6473` na base de arquivos da evidência. Note que a recomendação do arquivo nesse caso é `ignore`, uma vez que o arquivo pertence ao `Microsoft Windows XP`, como pode ser visto na descrição.

```
  (assert (blackboard-fact
    (id 6473)
    (agent hashsetagent-01)
    (recommendation "Ignore")
    (description "Microsoft Windows XP file")
    (time-taken 320)))
```

Nesse experimento, os agentes especialistas eram empregados por inteiro, ou seja, todas as suas regras de produção eram executadas sequencialmente. O gerente operacional,

portanto, só determinava quais especialistas desejava utilizar e em qual ordem, atribuindo a eles os recursos necessários. Notou-se que algumas regras não apresentavam resultados tão bons quanto outras e que para executar as melhores regras de um agente, às vezes seria necessário aguardar a execução de várias regras inferiores de outro agente ocupando um recurso. Por isso, posteriormente, o planejamento das ações dos agentes especialistas passou a levar em consideração as regras de produção individualmente, inclusive na avaliação do desempenho do agente.

A Tabela 5.7 apresenta uma amostra do *blackboard* após a execução dos agentes. Os atributos dos arquivos foram omitidos para simplificar a exibição. O sinal entre parênteses na recomendação `Inform` apresenta um viés para incluir (+) ou ignorar (-) o arquivo com base na informação do agente. É um estado intermediário entre as recomendações de incluir e ignorar.

Tabela 5.7: Amostra dos resultados do *blackboard*

Arquivo	**Recomendação**	**Conflito**
1	FileSignatureAgent - Inform(+) (possible digital camera image) TimeLineAgent - Inform(+) (recently created)	NO
2	HashSetAgent - Ignore (zero-size file) FilePathAgent - Inform(+) (Windows spool file)	YES
3	FileSignatureAgent - Inform(+) (possible Yahoo! Login information) FilePathAgent - Inform(-) (cookie file)	YES
4	HashSetAgent - Alert (suspected child porn) FileSignatureAgent - Alert (bad extension: expecting a video file) FilePathAgent - Inform(+) (file in user folder)	NO

Com esse experimento, observou-se a existência real de conflitos nas recomendações dos agentes. Nesse primeiro momento, a resolução de conflitos era realizada de forma bastante simples. Sempre que houvesse conflito, qualquer recomendação com viés de inclusão deveria prevalecer. Com isso, nenhum arquivo que pudesse ser importante era ignorado, mas o volume de arquivos selecionados certamente aumentaria.

Para melhorar a resolução de conflitos, decidiu-se incluir uma fase de revisão dos resultados do *blackboard* em que o especialista observa as recomendações do SMA e faz as correções necessárias. Essas correções são então utilizadas para avaliar a precisão dos agentes, mais especificamente de suas regras de produção. Com esse mecanismo de revisão os gerentes podem determinar quais agentes acertam mais e em quais situações, podendo resolver conflitos com maior precisão e também empregar os melhores agentes ou blocos de regras prioritárias.

5.4 Estudo de Caso 1

Além dos três experimentos listados foram realizados dois estudos de caso com quatro agentes especialistas implementados (`HashSetAgent`, `FilePathAgent`, `FileSignatureAgent` e `TimelineAgent`). O objetivo foi avaliar os resultados obtidos pelo conjunto de agentes, segundo os seguintes critérios:

- abrangência do exame - quantidade de arquivos que receberam pelo menos uma recomendação de algum agente especialista;
- potencial de redução do volume de arquivos a examinar - quantidade de arquivos que tiveram `ignorar` como recomendação;
- existência de conflitos - quantidade de arquivos com duas recomendações divergentes e
- correlação - quantidade de arquivos que sugerem relação entre duas ou mais evidências.

Na avaliação do desempenho individual dos agentes, a abrangência foi considerada em relação a quais arquivos o agente tem condições de examinar, mesmo que ele não consegui emitir uma recomendação sobre alguns dos arquivos.

No primeiro estudo, um caso contendo sete discos rígidos e um total de 450 mil arquivos foi examinado, conforme apresentado em Hoelz et al. (2008a). A investigação visava descobrir indícios de contrabando e esclarecer, se possível, o modo de operação da quadrilha.

Uma possibilidade interessante explorada nesse estudo de caso foi a realização de algumas pré-análises rápidas para sugerir a prioridade de uma evidência com base no seu conteúdo. Descobriu-se, por exemplo, que uma das evidências apresentava 83% de todas as mensagens de correio eletrônico presentes no caso. Essa evidência recebeu então a prioridade mais alta. Outra evidência, no entanto, apresentava apenas dez mil arquivos, ou menos de 3% do total de arquivos do caso, sendo que aproximadamente 10% desses arquivos eram diretórios ou arquivos de tamanho vazios (tamanho zero). Ela recebeu então a prioridade mais baixa. A Tabela 5.8 apresenta a atribuição de prioridades das evidências e a divisão dos quatro recursos disponíveis para o exame do caso.

Tabela 5.8: Distribuição de recursos no estudo de caso 1

EVIDÊNCIA	PRIORIDADE	REP	RECURSOS
Evidência 1	VERY LOW (1)	1/20 = 0,05	0,3
Evidência 2	LOW (2)	2/20 = 0,10	0,6
Evidência 3	LOW (2)	2/20 = 0,10	0,6
Evidência 4	NORMAL (3)	3/20 = 0,15	0,9
Evidência 5	NORMAL (3)	3/20 = 0,15	0,9
Evidência 6	HIGH (4)	4/20 = 0,20	1,2
Evidência 7	VERY HIGH (5)	5/20 = 0,25	1,5
TOTAL	20	20	6

Mais uma vez, a escassez de recursos leva a uma situação em que não há recursos suficientes para nenhum dos gerentes operacionais. Utilizando a ideia de subconjunto

de trabalho apresentado no Capítulo 4, o exame das evidências é limitado então a três evidências simultâneas. Assim, o novo cálculo de distribuição dos recursos, apresentado na Tabela 5.9, é realizado sobre as três evidências de maior prioridade.

Tabela 5.9: Distribuição de recursos em um subconjunto de trabalho

EVIDÊNCIA	PRIORIDADE	REP	RECURSOS
Evidência 5	NORMAL (3)	$3/12 = 0{,}25$	$1{,}5 \rightarrow 1$
Evidência 6	HIGH (4)	$4/12 = 0{,}33$	$2 \rightarrow 2$
Evidência 7	VERY HIGH (5)	$5/12 = 0{,}42$	$2{,}5 \rightarrow 3$
TOTAL	12	12	6

Os demais cálculos e o funcionamento do sistema foram omitidos, pois são semelhantes aos Experimentos 1 e 2 apresentados nas Seções 5.1, 5.2 e 5.3. Após a execução dos quatro agentes especialistas, os resultados obtidos foram avaliados segundo os quatro critérios definidos: abrangência, redução, conflitos e correlação. A correlação nesse caso foi realizada pelo próprio gerente tático, sem o uso de agentes especialistas dedicados a essa tarefa. Os resultados obtidos pelos agentes especialistas são apresentado a seguir:

1. o `HashSetAgent` encontrou 140 mil arquivos duplicados, 30 mil arquivos de tamanho zero e outros 6% dos arquivos que pertenciam ao conjunto de *hashes* do sistema operacional Microsoft Windows 98. Considerando a sobreposição de algumas das recomendações, o índice de redução sugerido foi de 42%. Em termos de abrangência, o agente analisou 69% dos arquivos do caso em que o cálculo de *hash* era aplicável. O agente não emitiu nenhuma recomendação de alerta;

2. o `FilePathAgent` sugeriu ignorar 42 mil arquivos de imagem pequenos (com menos de um *cluster*, 4096 bytes, de tamanho lógico) localizados no diretório de arquivos temporário do navegador de Internet. Ele também sugeriu ignorar 1.761 arquivos de *cookie*. Outras sugestões incluíram ignorar aproximadamente 89 mil arquivos relacionados à plataforma Java e outros 3.528 arquivos de documentação de programas. A redução total sugerida pelo agente foi de 31%. O agente sugeriu a inclusão de 5875 arquivos localizados em diretórios de atividade do usuário, como a pasta "Meus documentos". A abrangência do agente foi de 64%.

3. o `FileSignatureAgent` sugeriu a inclusão de 17 fotografias digitais, 155 *cookies* e outros 1709 arquivos temporários do navegador de Internet relacionados a serviços de *webmail*. Arquivos cuja extensão não correspondia ao cabeçalho (primeiros bytes do arquivo), em um total de 153, receberam a recomendação de alerta. O agente sugeriu ainda ignorar 113 mil arquivos baseado nos tipos (arquivos executáveis como DLL, EXE, VXD e arquivos de *bytecode* Java), que representavam 25% do total. A abrangência foi de cerca de 45%;

4. o `TimelineAgent` detectou a criação de 23 mil arquivos relacionados à instalação de ambientes Java em duas das evidências no mesmo dia, o que resultou em um recomendação de ignorar tais arquivos. Ele também sugeriu o exame de 8.800 arquivos utilizados nos últimos 30 dias, o que representa cerca de 2% dos arquivos. A abrangência do exame foi de 74% dos arquivos.

As Tabelas 5.10 e 5.11 apresentam os resultados dos agentes. O fator de redução (número de arquivos que os agentes sugeriram ignorar) obtido foi de 62%. A cobertura total da análise foi de cerca de 80%, o que significa que 80% dos arquivos foi examinado por pelo menos um agente especialista.

O tempo necessário para a realização desse exame por um examinador foi de aproximadamente 25 horas. O tempo não leva em consideração a preparação do caso com a ferramenta FTK. Foi considerado apenas o tempo gasto pelo perito para a análise e seleção do conteúdo considerado relevante. O SMA levou apenas quatro horas para realizar a análise dos sete discos rígidos em um processador com quatro núcleos e 3 GB de RAM. O tamanho total dos arquivos do caso era de 113 GB. A redução no tempo de execução, portanto, foi da ordem de 84%.

Neste estudo de caso, a maioria das decisões dos agentes não apresentou conflitos, sendo que o conjunto de conflitos foi menor que 1% dos arquivos. Os conflitos foram resolvidos pelo princípio de sempre sugerir a inclusão do arquivo divergente. Em termos de correlação, um aplicativo específico de controle comercial foi encontrado em três evidências. O número de arquivos relacionados à atividades de usuário encontrados em múltiplas evidências foi de apenas 51 arquivos, consistindo principalmente de documentos e planilhas.

5.5 Estudo de Caso 2

O segundo estudo de caso, apresentado em Hoelz et al. (2009), consistia em um caso com 10 discos rígidos e quatro mídias removíveis do tipo *pen drive* em uma investigação de fraude. O número total de arquivo era 353.466 com um total de 75,5 GB, incluindo arquivos recuperados mas excluindo fragmentos de espaço livre. A seguir são descritos alguns dos resultados mais significativos obtidos pelos agentes especialistas:

1. o `HashSetAgent` encontrou 246.941 arquivos duplicados (com 44.362 valores distintos de *hash*), 3.025 arquivos de tamanho zero e mais 15.553 arquivos que foram ignorados utilizando as bibliotecas de arquivos conhecidos. Cada um desses resultados vêm de uma regra diferente utilizada pelo agente. Considerando a sobreposição entre algumas das recomendações, a redução final sugerida pelo agente foi de 69,8%. Em termos de abrangência, o agente examinou 81,3% dos arquivos do caso em que o cálculo de *hashes* era aplicável. Devido à natureza do caso, apenas bibliotecas de *hash* de aplicativos comumente instalados, como sistemas operacionais e pacotes de escritório, foram utilizadas;

2. o `FilePathAgent` sugeriu a remoção de 5.811 arquivos de imagem pequenos (com tamanho lógico menor que a média das imagens), localizados nos arquivos temporários do navegador de Internet. Sugeriu também que 2.134 arquivo de *cookie* fossem ignorados. Outras sugestões incluíam ignorar 9.871 arquivos relacionados à plataforma Java e outros 6.186 arquivos de documentação de programas. A redução total sugerida foi de 6,8% dos arquivos. O `FilePathAgent` identificou a presença do aplicativo OpenOffice e sugeriu a inclusão de 100 documentos no formato ODT. Ele também sugeriu a inclusão de 2.095 arquivos do pacote Microsoft Office localizados em pastas do usuário do sistema e aletou sobre a presença de sistemas conhecidos

(específicos desse caso) que poderiam ter sido utilizados na fraude. A abrangência do agente nesse caso foi de 9,8%;

3. o `FileSignatureAgent` sugeriu ignorar 200 fotografias digitais, incluir 275 arquivos de *cookie* e 279 arquivos temporários do navegador de Internet relacionados à serviços de *webmail* e *home banking*. Outros 200 arquivos possuíam extensão que não correspondia ao cabeçalho do arquivo e receberam a recomendação de aleta. O agente sugeriu ainda ignorar 67.621 arquivos baseado em seus tipos (arquivos executáveis como DLL, EXE, VXD e arquivos de *bytecode* Java, que representavam aproximadamente 19% dos casos. A abrangência do agente foi de aproximadamente 26%;

4. o `TimelineAgent` detectou diversos pontos de instalação de *software* e atualização do sistema. Ele sugeriu ignorar arquivos executáveis e documentos criados nesses eventos. Com isso, aproximadamente 63.000 arquivos foram ignorados. O agente alertou sobre a modificação de 376 arquivo em finais de semana. Esse comportamento não era esperado, pois a maioria absoluta das modificações observadas nos arquivos foi feita em dias úteis. Outros 39 documentos receberam a recomendação de inclusão, pois foram modificados nas duas semanas anteriores à apreensão da evidências, período que era de interesse para a investigação. O agente obteve um fator de redução de 17,8% e 48% de abrangência.

As Tabelas 5.10 e 5.11 apresentam os resultados dos agentes. Com relação ao fator de redução, o percentual final obtido foi de 73%. A abrangência total da análise foi de aproximadamente 85%. O conjunto de conflito foi menor que 1% dos arquivos. Um exemplo significativo de conflito ocorreu entre o `FilePathAgent` e o `FileSignatureAgent`. Enquanto o primeiro sugeriu ignorar todos os arquivos de *cookie*, o segundo sugeriu a inclusão de *cookies* relacionados a serviços de *webmail* e *home banking*.

Em termos de correlação, foram pesquisados arquivos executáveis, documentos e mensagens de correio eletrônico presentes em duas ou mais evidências. Os resultados indicaram a presença de sete sistemas específicos instalados em todas as evidências. Foram identificados 52 documentos em uma das mídias removíveis que também estavam em um dos discos rígidos e outros 428 documentos que estavam em mais de uma evidência. Não foram encontradas mensagens de correio eletrônico comuns à mais de um evidência.

O exame dessas evidências demorou cerca de duas horas utilizando um processador com quatro núcleos e 3 GB de memória RAM. O caso foi previamente processado utilizando a ferramenta forense FTK. O tempo gasto pela ferramenta não foi contabilizado nesse tempo total. O tempo necessário para realizar o mesmo exame por dois peritos foi de 24 horas úteis, ou o correspondente à três dias de dedicação integral ao caso. Ressaltando mais uma vez que o tempo de pré-processamento do caso utilizando a ferramenta FTK não foi contabilizado. Só foi considerado o tempo gasto pelos peritos para a análise e seleção do conteúdo considerado relevante.

Para melhorar a abrangência dos exames, é necessário introduzir novas regras nos agentes existentes e também criar novos agentes, permitindo examinar o máximo de arquivos possível, mesmo que por apenas um agente especialista. Em termos de abrangência, observou-se um resultado interessante. Em algumas ferramentas forenses, arquivos como *logs* de aplicativos de troca de mensagens e documentos do conjunto de aplicativos de escritório OpenOffice são muitas vezes classificados em categorias genéricas de arquivos,

sendo facilmente ignorados equivocadamente em meio a centenas de milhares de arquivos. Nesse experimento, esses tipos de arquivos foram examinados e apresentados para o examinador por pelo menos um agente.

5.6 Comparação dos resultados

A Tabela 5.10 apresenta uma comparação das características e resultados dos dois estudos de caso realizados. A diferença nos resultados deve-se não só à diferença do tipo de investigação, mas também à natureza das evidências. Enquanto o primeiro caso apresentava apenas discos rígidos, alguns deles com sistema operacionais mais antigos instaladas, como Microsoft Windows 98, no segundo caso havia discos rígidos e *pen drives*. Essas diferenças destacaram a necessidade de uma abordagem que levasse em conta as características dos casos e de suas evidências. A partir desse ponto, iniciou-se o estudo da aplicação do RBC.

Tabela 5.10: Comparação dos resultados dos estudos de caso

	Estudo 1	**Estudo 2**
Crime	Contrabando	Fraude em licitação
Número de evidências	7	14
Tipo das evidências	Discos rígidos	Discos rígidos e *pen drives*
Total de arquivos	450.000	353.466
Tamanho	113 GB	75,5 GB
Redução	62%	73%
Abrangência	80%	85%
Tempo gasto pelo examinador	25 horas	24 horas
Tempo gasto pelo MADIK	4 horas	2 horas

Avaliando os resultados, notou-se que os obtidos pelo MADIK foram similares, mas não tão completos quando a análise dos especialistas humanos. Observou-se que o examinador humano gasta muito tempo em atividades repetitivas como ignorar arquivos ou inspecionar fragmentos de páginas da Internet armazenadas em *cache*, no intuito de garantir que seu exame é realmente abrangente. O uso do MADIK pode ser muito útil nessas atividades, embora o fator de redução, entre 62% e 73%, ainda possa ser melhorado. A comparação dos resultados também sugere que enquanto para o examinador humano é mais custoso o exame em termos do número de arquivos, para o SMA, o volume dos dados, relacionado ao tamanho do caso em gigabytes, tem maior impacto.

Com relação ao desempenho de cada tipo de agente especialista, a Tabela 5.11 apresenta a comparação dos resultados obtidos. As discrepâncias no desempenho dos especialistas também indicou a necessidade de aumentar o controle da execução desses. Desde então, a tarefa é passada ao agente com uma regra de produção explicitamente associada, ou seja, o agente executa apenas uma regra de produção por vez sobre o conjunto dos arquivos. Isso permite também aplicar o RBC, aplicando as melhores regras nas melhores situações.

O conjunto de conflitos, embora tenha se mostrado bastante reduzido, ele deve aumentar com a inclusão de novas regras de produção e novos agentes especialistas no SMA.

Tabela 5.11: Comparação dos especialistas nos estudos de caso

Agente	Caso	Abrangência	Redução
`HashSetAgent`	1	69%	42%
	2	81,3%	69,8%
`FilePathAgent`	1	64%	31%
	2	9,8%	6,8%
`FileSignatureAgent`	1	45%	25%
	2	26%	19%
`TimelineAgent`	1	74%	9%
	2	48%	17,8%

Uma observação sobre os critérios de avaliação dos agentes especialistas deve ser feita. A importância do agente para o sistema não deve ser medida ou comparada individualmente. Isso porque o resultado da coletividade é o fator mais importante no sistema MADIK, tanto em termos de redução, quanto de abrangência. A utilização desses critérios teve como objetivo comparar a atuação do agente em tipos de casos diferentes. As Figuras 5.5 e 5.6 ilustra os resultados obtidos nos dois estudos de caso.

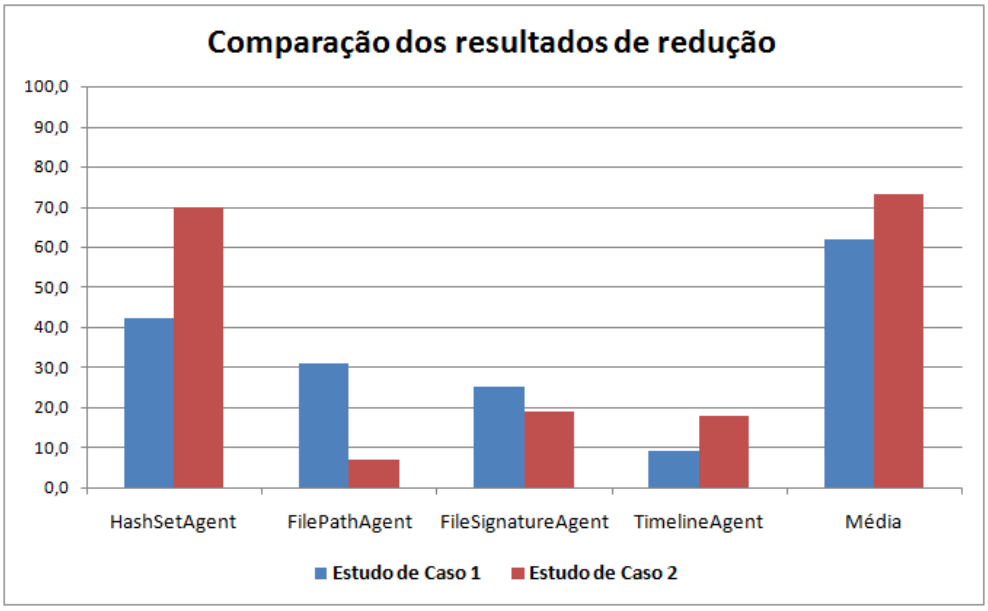

Figura 5.5: Comparação dos resultados de redução dos dois estudos de caso

O valor de abrangência alcançado em ambos os casos ainda é considerado baixo. Em casos com 400 mil arquivos, uma abrangência média de 82% resultaria em um conjunto de 81 mil arquivos sem qualquer recomendação. Espera-se que com a avaliação sistemática desse conjunto não coberto pelos especialistas, possa-se descobrir novas regras de produção para atingir níveis mais altos de abrangência com maior especialização dos agentes.

O percentual de redução, por outro lado, apresentou-se surpreendentemente alto. De fato, era de se esperar que em investigações de crimes por computador incidentais, nos quais o computador não é utilizado diretamente na prática do crime e age mais como uma "testemunha" dos acontecimentos, um grande volume de arquivos de sistema operacional e

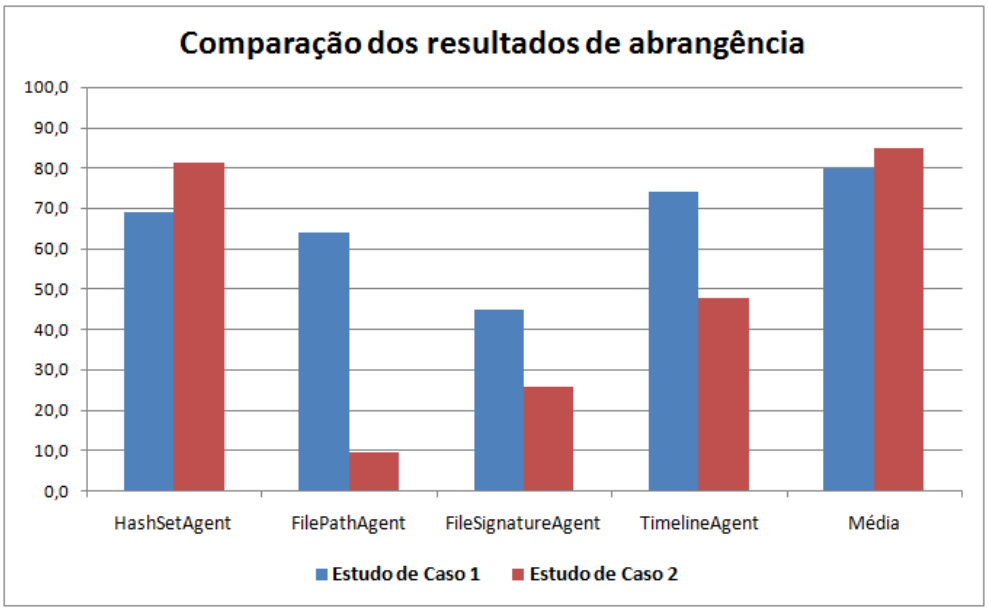

Figura 5.6: Comparação dos resultados de abrangência dos dois estudos de caso

de aplicativos instalados pudessem ser ignorados, deixando apenas arquivos relacionados às atividades do usuário como criação de documentos, planilhas, troca de mensagens instantâneas e de correio eletrônico, navegação na Internet em serviços de *webmail* e serviços bancários, dentre outros. Futuramente, deve-se avaliar também se esse desempenho se mantém em casos de crimes auxiliados por computador ou centrados em computador.

Capítulo 6

Conclusões

Esta dissertação apresentou uma proposta de aplicação de uma abordagem multiagente para o exame pericial de sistemas computacionais. O sistema proposto como solução foi denominado *Multi-Agent Digital Investigation toolKit* (MADIK). Ele apresenta uma arquitetura baseada em uma organização hierárquica de quatro níveis, com a utilização de um *blackboard* para compartilhamento de informações e de RBC como mecanismo de aprendizagem a ser desenvolvido.

O objetivo maior deste trabalho foi conceber um SMA que permitisse ganhos significativos de desempenho e eficácia nos exames periciais, com melhor aproveitamento de recursos computacionais e humanos e que fornecesse mecanismos de reutilização de conhecimento. A inteligência e autonomia dos agentes que compõem o sistema buscam incorporar o conhecimento dos especialistas para dar aos peritos recomendações sobre o conteúdo das evidências, permitindo que o exame pelo especialista humano dê maior ênfase aos arquivos com maior possibilidade de serem relevantes para a investigação.

O sistema foi testado utilizando dados de investigações reais com quatro agentes especializados. Os resultados mostraram que a aplicação de um SMA no exame pericial de computadores é uma abordagem interessante para reduzir o volume de arquivos a serem examinados pelo perito, assim como reduzir o tempo total necessário para realizar o exame. As reduções no volume dos arquivos a serem examinados, sugeridas pelos agentes especialistas, foram de 62% e 73% nos dois estudos de caso realizados. As reduções no tempo total de exame quando comparados com examinadores humanos foram de 84% e 91%.

O sistema MADIK não é proposto como um substituto para as ferramentas forenses comumente utilizadas, mas como um complemento. Portanto, um trabalho futuro deve ser a integração do sistema com essas ferramentas. Tal integração deve aumentar as possibilidades de análise e automação de tarefas rotineiras.

A abordagem de distribuição de trabalho na organização hierárquica também apresentou resultados promissores, que indicam que o sistema pode sustentar um número grande de agentes antes que os custos de comunicação e coordenação interrompam os ganhos de distribuição. Essa natureza distribuída da proposta permitirá que novas análises que atualmente, devido ao alto custo computacional, são inviáveis utilizando apenas uma estação de trabalho sejam realizadas de maneira distribuída em tempo hábil. Uma abordagem distribuída como a desta proposta também é de suma importância para conseguir lidar com o volume de dados encontrados nas investigações atuais.

A autonomia da organização de agentes também permite que os exames sejam realizados ininterruptamente, utilizando o recurso ocioso dos computares disponíveis no laboratório forense. Com isso, os peritos utilizam melhor o seu tempo em exames de maior complexidade, enquanto os agentes especialistas analisam os casos mais comuns. Nos estudos de caso realizados, os agentes especialistas trabalhando cooperativamente alcançaram índices de abrangência do exame dos arquivos de 80% e 85%. Com a melhoria dessa abrangência e da especialização dos agentes, reduz-se também de forma indireta o volume de dados a serem examinados. Uma vez que o perito pode confiar mais nas recomendações dos agentes, ele precisa examinar apenas os arquivos que não foram analisados por nenhum agente especialista, reduzindo assim o tempo de revisão dos resultados.

O MADIK também pode ser empregado na pré-análise das evidências, fornecendo rapidamente elementos que ajudem o perito a priorizar de maneira mais precisa o material a ser examinado. Assim, a evidência com maior potencial como prova é examinada primeiro e os seus resultados podem ser utilizados prontamente pela equipe de investigação.

As características de correlação, embora ainda sejam limitadas, demonstram as possibilidades providas pelo sistema para a descoberta de evidências importantes que podem não ser notadas quando as evidências são examinadas separadamente ou quando um caso apresenta um grande volume de dados. Nesses casos é comum que diversos peritos examinem separadamente as evidências e devido à ausência de recursos colaborativos nas ferramentas forense atuais, não compartilhem suas descobertas, o que pode resultar na perda de evidências.

A combinação da redução no volume das evidências a serem examinadas pelos peritos e a redução no tempo total de execução obtidas pelo SMA demonstram o potencial da proposta e os ganhos de produtividade que ela pode oferecer aos peritos em Informática Forense, resultando também em resultados mais rápidos e de maior qualidade para os demais envolvidos na persecução penal, desde a investigação até o processo judicial. Os resultados obtidos mostraram que MADIK é um resposta viável e promissora para o cenário atual de demanda e complexidade crescentes.

6.1 Trabalhos futuros

A aplicação do MADIK na realização de exames periciais de sistemas computacionais e os resultados obtidos sugerem diversas oportunidades de trabalhos futuros para ampliar a aplicação do sistema e melhorar seus resultados.

A integração com as ferramentas periciais existentes como *Forensic ToolKit* (FTK) e *Guidance EnCase* é uma possibilidade interessante. Conforme citado no trabalho, o MADIK não pretende recriar soluções para a aquisição, recuperação e extração de dados já implementadas nessas ferramentas, mas funcionar como uma camada superior de análise das evidências disponibilizadas por estes aplicativos.

A criação de novos agentes é uma possibilidade contínua. Conforme apresentado na Seção 2.3, há muitas propostas que podem ser implementadas em termos de novos agentes ou como extensão dos atuais. Como exemplos, podem ser citadas as propostas de Kornblum (2006) e Roussev et al. (2007) que utilizam novas técnicas para o cálculo de *hashes* que poderiam ser integradas ao `HashSetAgent`. Outro exemplo é a abordagem sugerida em Vel (2004), que permitiria o aprendizado automático de classificação de arquivos, o que seria bastante útil ao `FileSignatureAgent`. Além disso, é possível aproveitar di-

versas ferramentas forenses de finalidade específica já existentes, utilizando um agente especialista como *wrapper*, semelhante ao proposto por Alinka et al. (2006).

Com a aplicação do MADIK em novos casos e a disponibilidade mais abundante de resultados, poder-se-á realizar uma avaliação mais precisa da qualidade dos agentes e dos planos concebidos pelo sistema. Pretende-se estudar a aplicação de técnicas de descoberta de conhecimento e mineração de dados sobre os dados do *blackboard* para identificar novas regras de correlação e associação e pontos de melhoria nas regras de produção dos agentes tanto em termos de abrangência quanto de confiabilidade. Nesse sentido, pode-se estudar a utilização da metodologia *Cross-Industry Standard Process for Evidence Mining* (CRISP-EM), proposta por Venter et al. (2007) para mineração de evidências digitais em investigações criminais. A metodologia CRISP-EM é baseado no *Cross-Industry Standard Process for Data Mining* (CRISP-DM) utilizado para mineração de dados em geral. A Figura 6.1 ilustra as fases do processo CRISP-EM.

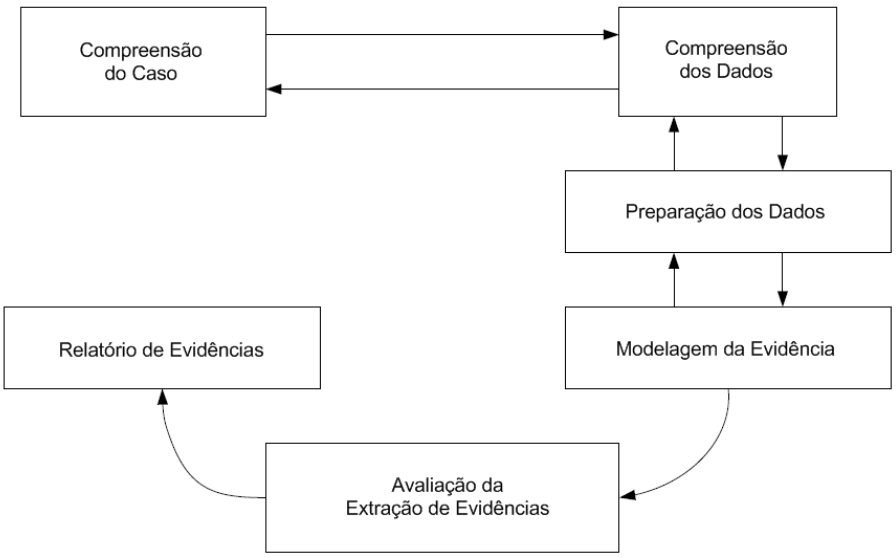

Figura 6.1: Fases principais do processo CRISP-EM, adaptada de Venter et al. (2007)

O uso de ontologias também deve ser ampliado em estudos paralelos, levando em consideração não só o domínio pericial, mas também o domínio jurídico e legal, permitindo que mecanismos de raciocínio e argumentação sobre o *blackboard* possam ser implementados. Isso permitirá que o MADIK seja aplicado também na fase do laudo pericial, auxiliando o perito na apresentação de suas descobertas para profissionais de outros domínios como promotores e juízes.

A implementação e teste do uso de RBC, proposta na Seçaõ 3.6 é a proposta mais imediata de trabalho futuro. Conforme apresentado na Seção 4.1, a descrição de um caso envolve vários elementos, sendo o principal o tipo de crime. Porém, mais do que a descrição do caso, também é importante a descrição das evidências que o compõe. Isso porque um crime de fraude, por exemplo, pode apresentar um perfil de evidências diferente em casos de pequeno e grande porte. Em casos de grande porte, pode-se encontrar evidências como servidores de e-mail, sistemas de contabilidade e controle de processos e até *mainframes*, que exigem uma priorização no exame diferente das evidências mais

comumente encontradas, como discos rígidos e *pen drives*, e que não são encontrados em investigações de menor porte.

Além disso, no nível operacional é importante determinar quais exames não melhores em cada tipo de evidência e caso, o que permitiria a utilização do planejamento baseado em casos nas atividades do MADIK. Para isso, pode-se utilizar como ponto de partida, a matriz de tarefas e objetivos apresentada por Beebe and Clark (2005) e ilustrada anteriormente pela Figura 2.7, na página 19. Nela definem-se as tarefas que devem ser realizadas para atingir os objetivos periciais propostos. Em um abordagem utilizando RBC, essa definição deverá ser mais detalhada, levando em consideração a natureza do caso e da evidência.

Com o uso de RBC, novos casos podem ser adaptados dos casos básicos definidos inicialmente ou de casos anteriores. A fase de revisão no ciclo do RBC é similar à revisão dos resultados do *blackboard* proposta neste trabalho. Com os resultados do mecanismo de revisão atual, é possível utilizar a confiabilidade dos agentes para reavaliar os planos utilizados com a alteração na ordem de execução das regras de produção do agentes especialistas. Com o uso de RBC, isso pode ser feito com maior precisão para cada tipo de caso e evidência.

Para isso, estudos devem ser realizados especialmente nos cálculos de similaridade empregados na fase de recuperação de casos do ciclo do RBC. No caso do MADIK, esta recuperação deve levar em consideração a organização hierárquica do SMA, o que exige que a similaridade seja calculada em dois níveis diferentes: (i) tático, levando em consideração a natureza dos casos e (ii) operacional, levando em consideração a natureza das evidências.

O MADIK foi proposto prevendo o armazenamento do máximo de informações dos exames realizados, incluindo as características dos casos e evidência e os planos utilizados pelos gerentes. Com a recuperação de casos similares empregada no RBC, pode-se recuperar planos de exames anteriores, que indicam quais agentes especialistas e suas respectivas regras de produção devem ser empregados e em qual ordem.

Portanto, são muitas as possibilidades de trabalhos futuros a partir da proposta e dos resultados obtidos nesse trabalho. Isso demonstra o potencial da abordagem multiagente aplicada à Informática Forense, que serve como base para a aplicação de outras propostas de IA como mineração de dados e RBC, visando auxiliar o especialista humano na realização dos exames periciais de sistemas computacional com a redução no tempo total do exame e no volume de dados a ser analisado, a possibilidade de correlação e descoberta de evidências em casos de grande porte, a obtenção e reutilização de conhecimento adquirido nos exames periciais e a melhor utilização dos recursos computacionais dos laboratórios forenses.

Referências

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59.

Alinka, W., Bhoedjanga, R., Bonczb, P., and de Vries, A. (2006). XIRAF – XML-based indexing and querying for digital forensics. *Digital Investigation*, 3S:S50–S58.

Beebe, N. L. and Clark, J. G. (2005). A hierarchical, objectives-based framework for the digital investigations process. *Digital Investigation*, 2(2):147–167.

Beebe, N. L. and Clark, J. G. (2007). Digital forensic text string searching: Improving information retrieval effectiveness by thematically clustering search results. *Digital Investigation*, 4S:S49–S54.

Bellifemine, F. L., Caire, G., and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE*. Wiley Series in Agent Technology, Sussex, England. ISBN 978-0-470-05747-6.

Bogen, C. and Dampier, D. A. (2005). Preparing for Large-Scale Investigations with Case Domain Modeling. In *Proceedings of the 2005 Digital Forensic Research Workshop (DFRWS)*.

Boyd, C. and Forster, P. (2004). Time and date issues in forensic computing - a case study. *Digital Investigation*, 1:18–23.

Brinson, A., Robinson, A., and Rogers, M. (2006). A cyber forensics ontology: Creating a new approach to studying cyber forensics. *Digital Investigation*, 3(3S):37–43.

Bruschi, D. and Monga, M. (2004). How to Reuse Knowledge About Forensic Investigations.

Buchholz, F. and Falk, C. (2005). Design and Implementation of Zeitline: a Forensic Timeline Editor. In *Proceedings of the 2005 Digital Forensic Research Workshop (DFRWS)*.

Buchholz, F. and Tjaden, B. (2007). A brief study of time. *Digital Investigation*, 4S:S31–S42.

Buteau, B. L. (1990). A generic framework for distributed, cooperating blackboard systems. In *Proceedings of the 1990 ACM Annual Conference on Cooperation*.

Case, A., Cristina, A., Marziale, L., Richard, G. G., and Roussev, V. (2008). FACE: Automated digital evidence discovery and correlation. *Digital Investigation*, 5(Supplement 1):S65–S75.

Ciardhuáin, S. . (2004). An Extended Model of Cybercrime Investigations. *International Journal of Digital Evidence*, 3(1).

Corchado, E., Corchado, J. M., and Abraham, A., editors (2007). *Innovations in Hybrid Intelligent Systems*. Springer.

Corkill, D. D. (2003). Collaborating software: Blackboard and multiagent systems & the future. In *Proceedings of the International Lisp Conference*.

Cossentino, M. and Potts, C. (2002). PASSI: a Process for Specifying and Implementing Multi-Agent Systems Using UML.

Costa, B. R. (2008). Proposta, Implementação e Testes de um Agente de Monitoração para o Sistema MADIK na Plataforma JADE. Trabalho de Graduação, Universidade de Brasília.

Cox, J. S. and Durfee, E. H. (2003). Discovering and exploiting synergy between hierarchical planning agents. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 281–288, New York, NY, USA. ACM.

DeLoach, S. A., Oyenan, W. H., and Matson, E. T. (2008). A capabilities-based model for adaptive organizations. *Autonomous Agents and Multi-Agent Systems*, 16:13–56.

Dileo, J., Jacobs, T., and Deloach, S. (2002). Integrating Ontologies into Multiagent Systems Engineering. In *Proceedings of 4th International Bi-Conference Workshop on Agent Oriented Information Systems (AOIS 2002)*, pages 15–16.

Durfee, E. H. and Lesser, V. R. (1991). Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation. *IEEE Transactions on Systems, Man, and Cybernetics, Special Issue on Distributed Sensor Networks*, SMC-21(5):1167–1183.

Foundation for Intelligent Physical Agents (FIPA) (2002). FIPA Abstract Architecture Specification (SC00001L).

Fox, M. and Long, D. (2003). PDDL 2.1 : An Extension to PDDL for Expressing Temporal Planning Domains. *Journal of Artificial Intelligence Research*, 20:61–124.

Ghallab, M., Nau, D., and Traverso, P. (2004). *Automated Planning: Theory And Practice*. Morgan Kaufmann Publishers.

Gruber, T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In Guarino, N. and Poli, R., editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands. Kluwer Academic Publishers.

Hannoun, M., Boissier, O., Sichman, J. S., and Sayettat, C. (2000). MOISE: An Organizational Model for Multi-agent Systems. In *IBERAMIA-SBIA*, pages 156–165.

Harrill, D. C. and Mislan, R. P. (2007). A Small Scale Digital Device Forensics ontology. *Small Scale Digital Device Forensics Journal*, 1.

Henderson-Sellers, B. and Giorgini, P. (2005). *Agent-Oriented Methodologies*. Idea Group Publishing.

Hill, E. F. (2003). *Jess in Action: Java Rule-Based Systems*. Manning Publications Co., Greenwich, CT, USA.

Hoelz, B. W. P. (2007). Um sistema multiagente para exames periciais em sistemas computacionais. In *Proceedings of the II International Conference of Forensic Computer Science*, Guarujá, Brazil.

Hoelz, B. W. P., Ralha, C. G., and Geeverghese, R. (2008a). A Colaborative Multi-Agent Approach to Computer Forensics. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*.

Hoelz, B. W. P., Ralha, C. G., and Geeverghese, R. (2009). Artificial Intelligence Applied to Computer Forensics. In *Proceedings of the 24th Annual ACM Symposium on Applied Computing (to appear)*.

Hoelz, B. W. P., Ralha, C. G., Geeverghese, R., and Júnior, H. C. (2008b). MADIK: A Colaborative Multi-Agent System Approach To Computer Forensics. In *Proceedings of the 16th International Conference on Cooperative Information Systems*.

Horling, B. and Lesser, V. (2005). A survey of multi-agent organizational paradigms. *The Knowledge Engineering Review*, 19:4:281Ũ316.

Hosmer, C. (1998). Time Lining Computer Evidence. WetStone Technologies, Inc. Whitepaper.

Huebner, E., Bem, D., and Bem, O. (2003). Computer Forensics: Past, Present And Future. *Information Security Technical Report*, 8(2):32–36.

Ieong, R. S. (2006). FORZA Ű Digital forensics investigation framework that incorporate legal issues. *Digital Investigation*, 3(3S):29–36.

Jagannathan, V., Dodhiawala, R., and Baum, L., editors (1989). *Blackboard Architectures and Applications*. Academic Press, Orlando, FL, USA.

Kenneally, E. E. and Brown, C. L. (2005). Risk sensitive digital evidence collection. *Digital Investigation*, 2(2):101–119.

Khan, M., Chatwin, C., and Young, R. (2007). A framework for post-event timeline reconstruction using neural networks. *Digital Investigation*, 4:146–157.

Kolodner, J. (1993). *Case-based reasoning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Kolp, M., Giorgini, P., and Mylopoulos, J. (2006). Multi-agent architectures as organizational structures. *Autonomous Agents and Multi-Agent Systems*, 13:3–25.

Kornblum, J. (2006). Identifying almost identical files using context triggered piecewise hashing. *Digital Investigation*, 3S:S91–S97.

Lee, J., Un, S., and Hong, D. (2008). High-speed search using Tarari content processor in digital forensics. *Digital Investigation*, 5:S91–S95.

Marling, C., Sqalli, M., Rissland, E., Muñoz-Avila, H., and Aha, D. (2002). Case-based reasoning integrations. *AI Magazine*, 23:1:69–86.

Martin, C. and Barber, K. S. (2006). Adaptive decision-making frameworks for dynamic multi-agent organizational change. *Autonomous Agents and Multi-Agent Systems*, 13:391–428.

McDermott, D. (1998). PDDL - the planning domain definition language. Technical Report Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control.

Mead, S. (2006). Unique file identification in the National Software Reference Library. *Digital Investigation*, 3:138–150.

Nau, D. (2007). Current trends in automated planning. *AI Magazine*, 28(4):43–58.

Noy, N. F. and McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory.

Palmer, G. (2001). A Road Map for Digital Forensic Research. Technical Report DTR - T001-01 FINAL, DFRWS. Report from the First Digital Forensic Research Workshop (DFRWS).

Pinson, S. and Moraïtis, P. (1996). An intelligent distributed system for strategic decision making. *Group Decision and Negotiation*, 6:77–108.

Reith, M., Carr, C., and Gunsch, G. (2002). An Examination of Digital Forensic Models. *International Journal of Digital Evidence*, 1(3).

Reust, J., editor (2006). *Digital Forensic Research Workshop 2005 Report.*

Roussev, V., III, G. G. R., and Marziale, L. (2007). Multi-resolution similarity hashing. *Digital Investigation*, 4S:S105–S113.

Roussev, V. and Richard III, G. G. (2004). Breaking the Performance Wall: The Case for Distributed Digital Forensics. In *Digital Forensic Research Workshop - DFRWS.*

Ruibin, G. and Gaertner, M. (2005). Case-Relevance Information Investigation: Binding Computer Intelligence to the Current Computer Forensic Framework. *International Journal of Digital Evidence*, 4(1).

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence - A Modern Approach*. Person Education.

Sommer, P. (2004). The challenges of large computer evidence cases. *Digital Investigation*, 1:16–17.

Staab, S. and Maedche, A. (2000). Ontology engineering beyond the modeling of concepts and relations. In *Proceedings of the ECAI'2000 Workshop on Application of Ontologies and Problem-Solving Methods, IOS Press, Amsterdam*.

Stevens, M. W. (2004). Unification of relative time frames for digital forensics. *Digital Investigation*, 1:225–239.

Stumme, G., Ehrig, M., Handschuh, S., Hotho, A., Maedche, A., Motik, B., Oberle, D., Schmitz, C., Staab, S., Stojanovic, L., Stojanovic, N., Studer, R., Sure, Y., Volz, R., and Zacharias, V. (2003). The Karlsruhe View on Ontologies. Technical report, University of Karlsruhe, Institute AIFB.

Sycara, K. P. (1998). Multiagent systems. *AI Magazine*, 19:79–92.

Turner, P. (2005). Digital provenance - interpretation, verification and corroboration. *Digital Investigation*, 2:45–49.

Turner, P. (2006). Selective and intelligent imaging using digital evidence bags. *Digital Investigation*, 3S:59–64.

van Liere, R., Harkes, J., and de Leeuw, W. (1998). A distributed blackboard architecture for interactive data visualization. In *Ninth IEEE Visualization 1998 (VIS '98)*.

Vel, O. D. (2004). File classification using byte sub-stream kernels. *Digital Investigation*, 1:150–157.

Veloso, M. M. (1994). *Planning and Learning by Analogical Reasoning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Venter, J., de Waal, A., and Willers, C. (2007). Specializing crisp-dm for evidence mining. In *Book Series-IFIP International Federation for Information Processing*, volume 242, chapter Specializing CRISP-DM for Evidence Mining. Springer, 1 edition.

Viroli, M., Holvoet, T., Ricci, A., Schelfthout, K., and Zambonelli, F. (2007). Infrastructures for the environment of multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14:49–60.

Vázquez-Salceda, J., Dignum, V., and Dignum, F. (2005). Organizing multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 11:307–360.

Weiss, G., editor (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. The MIT Press.

Wooldridge, M. (2002). *An Introduction to Multiagent Systems*. John Wiley & Sons.

Zimmerman, T. and Kambhampati, S. (2003). Learning-assisted automated planning: looking back, taking stock, going forward. *AI Magazine*, 24(2):73–96.

Zoethout, K., Jager, W., and Molleman, E. (2008). Task dynamics in self-organising task groups: expertise, motivational, and performance differences of specialists and generalists. *Autonomous Agents and Multi-Agent Systems*, 16(1):75–94.

Glossário

blackboard um espaço compartilhado onde os agentes podem ler e escrever dados, como em um quadro-negro. vi, vii, 49–51, 70, 73–75, 77, 78, 81, 83, 86, 87, 92, 96, 98, 101, 102, 104–106, 114, 116, 117

defacement prática que consiste na substituição do conteúdo original de um sítio *web*, geralmente da página inicial, por um conteúdo totalmente diverso pelo invasor. 6

***hash* criptográfico** é uma função unidirecional que transforma uma entrada com um número variável de bytes em uma saída de tamanho fixo pré-determinado. 18, 31

busca e apreensão ação com fulcro em autorização judicial que tem por finalidade procurar pessoa ou coisa que se deseja encontrar, para apresentá-la à autoridade que a determinou. 16

cadeia de custódia é um processo que visa manter e documentar o histórico de custódia das evidências, desde a coleta até o fim do processo, com o intuito de garantir a idoneidade dessas evidências e a rastreabilidade em caso de questionamentos.. 12, 16, 20, 28

esteganografia técnica que consiste em ocultar a existência de uma mensagem dentro de outra, como, por exemplo, ocultar dados em um arquivo de imagem aparentemente não modificado. 10, 18

evidência qualquer vestígio que, após exame, apresente relação constatada com o fato investigado. 5–13, 16, 17, 19–24, 26, 32–34

evidência digital qualquer evidência originária de equipamento eletrônico ou computacional. vi, 1, 8, 9, 12, 14, 20–23, 27, 29, 34

EXIF EXIF é uma especificação dos formatos de imagem utilizados em câmeras digitais. 32

materialidade a existência de elementos materiais que comprovem efetivamente a ocorrência do fato investigado. 17

perdimento na pena de perdimento de bens, os bens adquiridos ilegalmente ou utilizados em atividade ilegal tem seu uso cautelar ou definitivo revertido para a União. 19

vestígio qualquer elemento encontrado em uma cena de crime que possa ter relação com o fato investigado. 10–12, 16, 22, 23, 34

XQuery linguagem de consulta projetada para extração e manipulação de dados em documentos XML. 29