GUSTAVO HENRIQUE MATOS BEZERRA MOTTA

*UM MODELO DE AUTORIZAÇÃO CONTEXTUAL PARA O
CONTROLE DE ACESSO AO PRONTUÁRIO ELETRÔNICO DO PACIENTE EM
AMBIENTES ABERTOS E DISTRIBUÍDOS*

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção do
Título de Doutor em Engenharia.

São Paulo

2003

GUSTAVO HENRIQUE MATOS BEZERRA MOTTA

*UM MODELO DE AUTORIZAÇÃO CONTEXTUAL PARA O
CONTROLE DE ACESSO AO PRONTUÁRIO ELETRÔNICO DO PACIENTE EM
AMBIENTES ABERTOS E DISTRIBUÍDOS*

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção do
Título de Doutor em Engenharia.

Área de concentração:
Sistemas Eletrônicos.

Orientador:
Prof. Dr. Sérgio Shiguemi Furuie

São Paulo

2003

Ficha Catalográfica

Motta, Gustavo Henrique Matos Bezerra

Um modelo de autorização contextual para o controle de acesso ao prontuário eletrônico do paciente em ambientes abertos e distribuídos.
/ Gustavo Henrique Matos Bezerra Motta – São Paulo, 2003.
xvii, 212 p.

Tese (Doutorado – Programa de Pós-graduação em Engenharia Elétrica. Área de Concentração: Sistemas Eletrônicos) – Escola Politécnica da Universidade de São Paulo.

1. Prontuário eletrônico do paciente. 2. Autorização contextual.
3. Controle de acesso. 4. Segurança. I. Título.

“Um exemplo frisante para ilustrar este inconveniente está no fato de ter sido o habitante das florestas tropicais sul-americanas quem inventou a **maca...**”

Karl von den Steinen, 1894

A MARIA EUGÊNIA, exemplo de perseverança e dedicação na busca dos objetivos, que com seu amor, apoiou e incentivou meu progresso em mais uma jornada.

Agradecimentos

Ao professor Sérgio Shiguemi Furuie, pela confiança depositada em mim desde o nosso primeiro encontro, que, com orientação precisa e valiosa, contribuiu decisivamente para a realização deste trabalho e para o meu progresso profissional.

Aos meus pais, Glória e Luiz, pela educação que me proporcionaram.

Ao engenheiro Cândido Pinto de Melo (*in memoriam*), venerável exemplo de vida, que com sua liderança nos legou, no InCor-HC.FMUSP, um ambiente de P&D ímpar na área de informática em saúde.

Ao professor Marco Antônio Gutierrez, pelas idéias compartilhadas no início deste projeto, em particular, o incentivo para utilização do serviço de diretórios LDAP.

Aos diretores do Serviço de Informática do InCor, Marco Antônio Gutierrez, Umberto Tachinardi, Sérgio Shiguemi Furuie e Fábio Antero Pires, pelo apoio à execução desta pesquisa e pela determinação em utilizar seus resultados.

Ao professor Luis Carlos Carvalho, que me orientou nos primeiros passos da vida acadêmica e da pesquisa, por me apontar sempre os melhores caminhos.

A Fabiane Nardon, pela confiança e disposição em utilizar os resultados práticos desta pesquisa desde o seu início.

Aos amigos da Unidade de Pesquisa e Desenvolvimento do InCor, Marina Rebelo, Ramon Moreno, Nivaldo Bertozzo, Cecil Robilotta, Marcelo Santos, Ângela Vilhena, Luiz Kobayashi, Keyla Urtiga, Paulo Pilon, Júlio Figueiredo e Alice Bacic, que proporcionaram um ambiente descontraído, propício à criatividade e à troca de idéias.

Aos amigos das Unidades de Tecnologia e de Sistemas, que direta ou indiretamente, me auxiliaram a realizar este trabalho, em especial a Giuliana Orlandi, Marcos Yamaguti e Mary Kenchian, pela prestimosidade nas inúmeras instalações e configurações do servidor LDAP e a Cristina Papel, pela ajuda na configuração da política de acesso ao PEP.

Aos docentes da Pós-graduação em Engenharia Elétrica da Escola Politécnica da USP, pela contribuição à minha formação como pesquisador.

Aos colegas do Departamento de Informática da UFPB, cujo apoio ao assumirem as minhas funções didáticas, viabilizaram a realização deste trabalho.

A Carla e Leandro Rosa, amigos fraternos, que proporcionaram um início de jornada paulistana mais agradável.

A Malone e Eduardo Araújo, pela amizade e pelos bons momentos compartilhados, que tornaram mais amena a realização deste árduo trabalho.

À CAPES, pelo auxílio financeiro e à FAPESP, pelo suporte em infraestrutura de pesquisa.

Às secretárias do Serviço de Informática, Sílvia Marina e Lúcia, pela atenção e disposição em atender às minhas solicitações.

Às secretárias da Pós-Graduação da Escola Politécnica da USP, pela atenção, eficiência e educação dispensadas.

Sumário

<i>Resumo</i>	<i>xvi</i>
<i>Summary/Abstract</i>	<i>xvii</i>
<i>1 Introdução</i>	<i>1</i>
1.1 Aspectos ético-legais do acesso ao PEP	2
1.2 Aspectos computacionais do acesso ao PEP.....	5
1.3 Objetivo	7
1.4 Justificativa.....	8
1.5 Metodologia	10
1.6 Organização do trabalho	12
<i>2 Modelos de controle de acesso</i>	<i>13</i>
2.1 Conceitos e terminologia	13
2.1.1 O papel da autenticação.....	14
2.1.2 O papel da auditoria	15
2.1.3 O papel da política de acesso	16
2.2 Considerações sobre o controle de acesso discricionário	17
2.3 Considerações sobre o controle de acesso compulsório.....	19
2.4 O controle de acesso baseado em papéis.....	20
2.4.1 O padrão para CABP proposto pelo NIST.....	24
2.4.1.1 O modelo de referência	24
2.4.1.2 A especificação funcional.....	36
2.5 Trabalhos relacionados	38
2.5.1 Modelo E-RBAC.....	41
2.5.2 Modelo C-TMAC	42
2.5.3 Modelo CS-RBAC	43
2.5.4 Modelo TCM.....	44
2.5.5 Modelo DRIVE-RBAC	45

2.5.6	Modelo RBPEAC	46
2.5.7	Modelo XORBAC	47
2.5.8	Modelo OASIS.....	48
2.5.9	Modelo GTRBAC.....	49
2.6	Considerações finais.....	51
3	<i>Modelo de autorização contextual</i>	52
3.1	Contextos.....	53
3.2	Regras de autorização	55
3.2.1	Valores	55
3.2.2	Operadores	57
3.2.2.1	Aritméticos.....	57
3.2.2.2	Relacionais	57
3.2.2.3	Booleanos.....	58
3.2.3	Parametrização	58
3.3	A especificação formal do MACA	60
3.3.1	MACA básico	61
3.3.2	MACA hierárquico	64
3.3.3	MACA restrito.....	69
3.3.3.1	Separação de responsabilidade estática	73
3.3.4	Especificação do algoritmo de decisão de acesso.....	77
3.4	Aplicação do MACA a um PEP exemplo.....	80
3.4.1	Serviços do PEP exemplo	81
3.4.2	Requisitos da política de acesso	81
3.4.3	Especificação da política de acesso no MACA.....	83
3.4.3.1	Representação dos serviços do PEP exemplo	84
3.4.3.2	Hierarquia de papéis	87
3.4.3.3	Autorizações contextuais	88
3.4.3.4	Separação de responsabilidades	91
3.5	O MACA e as políticas administrativas no CABP.....	92
3.5.1	Descentralização e autonomia.....	93
3.5.2	Execução de uma política administrativa usando o MACA	95
3.6	Considerações finais.....	99

4	Arquitetura e implementação	100
4.1	Arquitetura de software	100
4.1.1	Prontuário eletrônico de pacientes	101
4.1.2	Base de informações de gerência de segurança	102
4.1.3	Servidor de segurança	104
4.1.3.1	O serviço de autorização do MACA e o controle de acesso no SSC	104
4.1.3.2	O serviço de autorização do MACA e o RAD – <i>Facility</i>	107
4.2	Implementação	109
4.2.1	Representação do MACA no LDAP	109
4.2.1.1	Introdução ao LDAP	109
4.2.1.2	Representação do MACA: aspectos de projeto	112
4.2.2	Servidor de segurança	117
4.2.2.1	Autenticação: objetos e interações	118
4.2.2.2	Autorização de acesso: objetos e interações	120
4.2.3	Ferramentas administrativas	128
4.2.3.1	O MACA AD <i>Web</i>	128
4.2.3.2	O MACA AD	131
4.2.3.3	O MACA CS Monitor	133
4.3	Resultados	135
4.3.1	Avaliação de desempenho	135
4.3.2	Aplicação do MACA no PEP do InCor-HC.FMUSP	142
4.3.3	Aplicação do MACA no DATASUS/MS	148
4.4	Considerações finais	150
5	Discussão	151
5.1	O MACA e o controle de acesso ao PEP	151
5.1.1	A contribuição das autorizações contextuais do MACA	151
5.1.1.1	Políticas de acesso para o PEP	152
5.1.1.2	Políticas administrativas	154
5.1.2	A contribuição da separação de responsabilidades independente da ativação automática de papéis	155
5.2	O MACA e os trabalhos relacionados	157
5.2.1	CABP hierárquico	157
5.2.2	CABP restrito	158

5.2.3	Autorizações	159
5.2.4	Regras contextuais	162
5.2.5	Implementação e aplicação ao PEP	164
5.3	Limitações do MACA e extensões futuras	165
6	Conclusão	168
Anexo A Linguagem de expressões lógicas: sintaxe formal		170
Anexo B Modelo do formulário de solicitação de acesso a sistemas informatizados		176
Anexo C Glossário de definições matemáticas da notação Z		178
Referências		181
Produção bibliográfica e técnica		192

Lista de figuras e gráficos

<i>Figura 1</i> – Esquema do controle de acesso e sua relação com outros serviços de segurança (Sandhu & Samarati, 1994).	15
<i>Figura 2</i> – Esquema do modelo de referência para CABP do NIST (Ferraiolo <i>et al.</i> , 2001).	26
<i>Figura 3</i> – Exemplos de hierarquias de papéis: (a) geral; (b) árvore; e (c) árvore invertida.	31
<i>Figura 4</i> – Hierarquia de papéis em estrutura de árvore invertida.	65
<i>Figura 5</i> – Aplicação do MACA para um PEP exemplo. (a) hierarquia parcial de papéis dos usuários de um hospital em estrutura de árvore invertida. (b) Objetos do PEP representados parcialmente numa estrutura de árvore invertida, com a lista das respectivas operações associadas. (c) Autorizações contextuais associadas a cada papel.	85
<i>Figura 6</i> – Exemplo de uma política administrativa usando o MACA. (a) hierarquia dos papéis administrativos. (b) Representação das funções administrativas do MACA. (c) Autorizações contextuais associadas a cada papel.	93
<i>Figura 7</i> – Organograma parcial da estrutura hierárquica de um hospital. Existem quatro tipos de unidades organizacionais: 1) a própria organização; 2) institutos; 3) divisões; e 4) serviços. Todo serviço está subordinado a uma divisão, toda divisão está subordinada a um instituto e todo instituto está subordinado a organização, que é única.	94
<i>Figura 8</i> – Esquema da arquitetura de software para integração do MACA.	101
<i>Figura 9</i> – Caso de uso para autenticação do usuário em CORBA (Object Management Group, 2001c).	105
<i>Figura 10</i> – Modelo de controle de acesso do SSC (Object Management Group, 2001c).	106
<i>Figura 11</i> – Interações entre cliente, objeto servidor (alvo) e o RAD Facility (Object Management Group, 2000b).	108
<i>Figura 12</i> – Serviço de diretórios distribuídos LDAP.	110
<i>Figura 13</i> – Exemplo parcial da árvore de informações de diretório do InCor – HC-FMUSP.	111
<i>Figura 14</i> – Diagrama de seqüência com as interações entre os objetos do servidor de segurança, para prover autenticação e autorização de acesso.	119
<i>Figura 15</i> – Página para consultar contas de usuários no MACA AD Web.	129

<i>Figura 16</i> – Página para gerência de contas de usuários MACA AD <i>Web</i>	129
<i>Figura 17</i> – Página do MACA AD <i>Web</i> para vinculação/desvinculação de papéis de usuários.	130
<i>Figura 18</i> – Janela do MACA AD para gerência de recursos (objetos e operações).	132
<i>Figura 19</i> – Janela do MACA AD para gerência de papéis.....	132
<i>Figura 20</i> – Janela do MACA AD para gerência de autorizações.	133
<i>Figura 21</i> – Página do MACA CS Monitor com estatísticas do sistema.....	134
<i>Figura 22</i> – Página do MACA CS Monitor com a lista parcial de usuários com sessões abertas.	134
<i>Figura 23</i> – Hierarquia dos papéis não administrativos atualmente definidos no MACA para o InCor-HC.FMUSP.....	144
<i>Gráfico 1</i> – Carga média de solicitações por minuto atendidas pelo servidor de segurança em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).	140
<i>Gráfico 2</i> – Tempos médios de resposta (ms) a solicitações de autenticação em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).	141
<i>Gráfico 3</i> – Tempos médios de resposta (ms) a solicitações de autorização em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).	141

Lista de tabelas

<i>Tabela 1</i> – Sumário das características identificadas nos modelos para CABP revisados.....	40
<i>Tabela 2</i> – Esquemas LDAP para representar entidades do MACA.....	115
<i>Tabela 3</i> – Experimento 1: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com <i>cache</i> do usuário ativado e IIOP sobre TLS.	139
<i>Tabela 4</i> – Experimento 2: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com <i>cache</i> do usuário ativado e IIOP sem TLS.	139
<i>Tabela 5</i> – Experimento 3: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com <i>cache</i> do usuário desativado e IIOP sem TLS.....	140

Lista de abreviaturas e siglas

AID: árvore de informações de diretório;
API: *application programming interface*;
BIGS: base de informações para gerência de segurança;
CABP: controle de acesso baseado em papéis;
CAC: controle de acesso compulsório;
CAD: controle de acesso discricionário;
CIAS: *clinical image access service*;
CNES: cadastro nacional de estabelecimento de saúde;
COAS: *clinical observation access service*;
CORBA: *common object request broker architecture*;
CS-RBAC: *context-sensitive role-based access control*;
C-TMAC: *contextual team-based access control*;
DAP: *directory access protocol*;
DATASUS/MS: Departamento de Informática do Sistema Único de Saúde/Ministério da Saúde;
DN: *distinguished name*;
DNS: *domain name server*;
DRIVE: *drug in virtual enterprise*;
E-RBAC: *environment role-based access control*;
EUA: Estados Unidos da América;
GPL: *general public license*.
GTRBAC: *generalized temporal role-based access control*;
HC-FMUSP: Hospital das Clínicas da Faculdade de Medicina da USP;
HIPAA: *health insurance portability and accountability Act of 1996*;
HISA: *healthcare information systems architecture*;
HIV: *human immunodeficiency virus*;
IBM: *International Business Machine*;
IDL: *interface definition language*;
IETF: *Internet Engineering Task Force*;
IIOP: *internet inter-ORB protocol*;
InCor: Instituto do Coração;
IP: *internetwork protocol*;
ISO: *International Organization for Standardization*;

ITU-T: International Telecommunication Union - Telecommunication Standardization Sector;

JNDI: java naming directory interface;

JSP: java server pages;

LDAP: lightweight directory access protocol;

MACA: Modelo de Autorização Contextual para o Controle de Acesso baseado em papéis;

MS: Ministério da Saúde;

NIST: National Institute of Standards and Technology;

OASIS: open architecture for securely interworking services;

OID: object identifier;

OMG: Object Management Group;

ORB: object request broker;

OSI: open systems interconnection;

PACS: picture archiving and communication system;

PEP: prontuário eletrônico do paciente;

PIDS: person identification service;

PL/SQL: procedural language/structured query language;

RAD – Facility: serviço de decisão para acesso a recursos;

RBPEAC: role-based policy-enforced access control;

RDN: relative distinguished name;

SASL: simple authentication and security layer;

SGBD: sistema gerenciador de banco de dados;

SIH: sistema de informação hospitalar;

SQL: structured query language;

SR: separação de responsabilidades;

SRD: separação de responsabilidades dinâmica;

SRE: separação de responsabilidades estática;

SSC: serviço de segurança da arquitetura CORBA;

TCM: Tees confidentiality model;

TCP/IP: transport control protocol/internet network protocol;

TLS: transport layer security;

TRBAC: temporal role-based access control;

UML: unified modeling language;

XML: extensible markup language;

XORBAC: extended object role-based access control;

Resumo

MOTTA, G. H. M. B. *Um modelo de autorização contextual para o controle de acesso ao prontuário eletrônico do paciente em ambientes abertos e distribuídos*. 2003. 212 p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2003.

Os recentes avanços nas tecnologias de comunicação e computação viabilizaram o pronto acesso às informações do prontuário eletrônico do paciente (PEP). O potencial de difusão de informações clínicas resultante suscita preocupações acerca da privacidade do paciente e da confidencialidade de seus dados. As normas presentes na legislação dispõem que o conteúdo do prontuário deve ser sigiloso, não cabendo o acesso a ele sem a prévia autorização do paciente, salvo quando necessário para beneficiá-lo. Este trabalho propõe o MACA, um modelo de autorização contextual para o controle de acesso baseado em papéis (CABP) que contempla requisitos de limitação de acesso ao PEP em ambientes abertos e distribuídos. O CABP regula o acesso dos usuários ao PEP com base nas funções (papéis) que eles exercem numa organização. Uma autorização contextual usa informações ambientais disponíveis durante o acesso para decidir se um usuário tem o direito e a necessidade de acessar um recurso do PEP. Isso confere ao MACA flexibilidade e poder expressivo para estabelecer políticas de acesso ao PEP e políticas administrativas para o CABP que se adaptam à diversidade ambiental e cultural das organizações de saúde. O MACA ainda permite que os componentes do PEP utilizem o CABP de forma transparente para o usuário final, tornando-o mais fácil de usar quando comparado a outros modelos de CABP. A arquitetura onde a implementação do MACA foi integrada adota o serviço de diretórios LDAP (*Lightweight Directory Access Protocol*), a linguagem de programação Java e os padrões CORBA *Security Service* e *Resource Access Decision Facility*. Com esses padrões abertos e distribuídos, os componentes heterogêneos do PEP podem solicitar serviços de autenticação de usuário e de autorização de acesso de modo unificado e coerente a partir de múltiplas plataformas. A implementação do MACA ainda tem a vantagem de ser um software livre, de basear-se em componentes de software sem custos de licenciamento e de apresentar bom desempenho para as demandas de acesso estimadas. Por fim, a utilização rotineira do MACA no controle de acesso ao PEP do InCor-HC.FMUSP, por cerca de 2000 usuários, evidenciam a exequibilidade do modelo, da sua implementação e da sua aplicação prática em casos reais.

Palavras-chave: prontuário eletrônico do paciente (PEP), autorização contextual, controle de acesso baseado em papéis (CABP), segurança, arquiteturas abertas e distribuídas, contexto.

Summary/Abstract

MOTTA, G. H. M. B. *A contextual authorization model for access control of electronic patient record in open distributed environments*. 2003. 212 p. Thesis (Doctoral) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2003.

The recent advances in computing and communication technologies allowed ready access to the electronic patient record (EPR) information. High availability of clinical information raises concerns about patients privacy and data confidentiality of their data. The legal regulation mandates the confidentiality of EPR contents. Everyone has to be authorized by the patients to access their EPR, except when this access is necessary to provide care on their behalf. This work proposes MACA, a contextual authorization model for the role-based access control (RBAC) that considers the access restrictions requirements for the EPR in open and distributed environments. RBAC regulates user's access to EPR based on organizational functions (roles). Contextual authorizations use environmental information available at access time, like user/patient relationship, in order to decide whether a user is allowed to access an EPR resource. This gives flexibility and expressive power to MACA, allowing one to establish access policies for the EPR and administrative policies for the RBAC that considers the environmental and cultural diversity of healthcare organizations. MACA also allows EPR components to use RBAC transparently, making it more user friendly when compared with other RBAC models. The implementation of MACA architecture uses the LDAP (Lightweight Directory Access Protocol) directory server, the Java programming language and the standards CORBA Security Service and Resource Access Decision Facility. Thus, heterogeneous EPR components can request user authentication and access authorization services in a unified and coherent way across multiple platforms. MACA implementation complies with free software policy. It is based on software components without licensing costs and it offers good performance for the estimated access demand. Finally, the daily use of MACA to control the access of about 2000 users to the EPR at InCor-HC.FMUSP shows the feasibility of the model, of its implementation and the effectiveness of its practical application on real cases.

Key-words: electronic patient record (EPR), contextual authorization, role-based access control (RBAC), security, open and distributed architectures, contexts.

Introdução

“Para mostrar a validade de uma regra (moral), pode ser suficiente uma boa razão; mas esta boa razão habitualmente não basta para fazer que a regra seja observada.”

Norberto Bobbio, 1998

O prontuário eletrônico do paciente (PEP) dispõe, em formato digital, informações que tradicionalmente são mantidas em registros físicos. O PEP registra informações, sinais e imagens, geradas a partir de fatos, acontecimentos e situações sobre a saúde do paciente e a assistência a ele prestada, de caráter legal, sigiloso e científico (Conselho Federal de Medicina, 2002a). Também inclui conteúdos administrativos e financeiros relativos a procedimentos ou tratamentos realizados (Rodrigues *et al.*, 2001). Enfim, mantém os documentos sobre o estado de saúde e os cuidados recebidos por um indivíduo ao longo da vida (Marin *et al.*, 2003).

Os recentes avanços nas tecnologias de comunicação e computação viabilizam o pronto acesso às informações do PEP por meio de sistemas distribuídos, independente do local e do momento em que o paciente é atendido. A disponibilidade das informações é vantajosa porque:

- Provedores de serviços de saúde podem conhecer melhor o paciente. Informações atualizadas permitem aumentar a qualidade do atendimento ao paciente e reduzir custos, minimizando a realização de exames e procedimentos redundantes ou desnecessários (National Academy of Sciences, 1997);
- Pesquisadores e o Serviço Público de Saúde podem realizar estudos epidemiológicos mais abrangentes, mais precisos, com benefícios para

toda a sociedade (Marin *et al.*, 2003; National Academy of Sciences, 1997);

- Gestores públicos e privados têm melhores subsídios para a tomada de decisão na definição de padrões para o atendimento à saúde (Rindfleisch, 1997). Essa disponibilidade facilita a participação e o controle social nas decisões sobre as políticas públicas de saúde, prioridades, alocação de recursos e utilização de novas tecnologias, permitindo que a sociedade seja informada da qualidade e da eficácia do seu sistema de saúde (Sacardo & Fortes, 2000).

O potencial de difusão de informações clínicas oferecido pelo PEP suscita preocupações acerca da privacidade do paciente e da confidencialidade de seus dados, mesmo com os benefícios que possa trazer. O dilema entre liberar ou não o acesso às informações presentes no prontuário é uma questão amplamente discutida pelos estudiosos da ética médica. As normas presentes nos códigos de ética profissional, em leis e em legislação complementar definem uma política geral de acesso para o PEP. Elas expressam as obrigações, as permissões e as proibições a que os entes relacionados ao prontuário estão sujeitos, bem como as sanções previstas para quem viola alguma regra.

As seções 1.1 e 1.2, a seguir, discutem os aspectos ético-legais e computacionais do acesso ao prontuário eletrônico do paciente, respectivamente.

1.1 Aspectos ético-legais do acesso ao PEP

As informações presentes no prontuário são de propriedade do paciente. As instituições e os profissionais da área de saúde são fiéis depositários do prontuário (Francisconi & Goldim, 1998). O paciente fornece informações pessoais em confidência durante o atendimento ou elas são obtidas a partir de exames e procedimentos realizados com finalidade diagnóstica ou terapêutica (Francisconi & Goldim, 1998). Em suma, o prontuário mantém informações íntimas que só interessam ao próprio paciente e que foram confidenciais por uma necessidade.

O segredo do conteúdo do PEP é um direito de todo cidadão, com guarida no art. 5º, inciso X, da Constituição Federal de 1988 (Brasil, 2001), que garante a inviolabilidade da intimidade, da vida privada, da imagem e da honra das pessoas. Oliveira (2001) aponta que o segredo não tem caráter puramente privado; corresponde a um patrimônio de todos, pois interessa à sociedade que o indivíduo possa confiar sua vida privada a alguém sem que ela venha à público. O dever para preservação do se-

greto é previsto no Código Penal Brasileiro ¹ e na maioria dos códigos de ética profissional.

O Código de Ética Médica impõe, no art. 11, o segredo como um princípio fundamental para o exercício da medicina:

“Art. 11 – O médico deve manter sigilo quanto às informações confidenciais que tiver conhecimento no desempenho de suas funções. O mesmo se aplica ao trabalho em empresas, exceto nos casos em que seu silêncio prejudique ou ponha em risco a saúde do trabalhador ou da comunidade.” (Conselho Federal de Medicina, 1996, p. 16).

O Capítulo IX do código determina as obrigações que o médico tem para com o segredo profissional. Por exemplo, é dever do médico orientar seus auxiliares e zelar para que respeitem o segredo profissional a que estão obrigados por lei. O código também proíbe o médico de facilitar o acesso ao prontuário para pessoas não obrigadas ao segredo profissional.

O silêncio exigido aos médicos tem a finalidade de preservar a privacidade do indivíduo. A revelação da vida privada do paciente pode acarretar prejuízos aos seus interesses morais e econômicos (França, 2002). Em princípio, só o consentimento do paciente autoriza a revelação do conteúdo do prontuário (Oliveira, 2001). O direito do paciente de exercer a sua privacidade, pois, consiste na autonomia para controlar as informações sobre seu estado de saúde, decidindo quais informações quer guardar para si e quais deseja comunicar (Sacardo & Fortes, 2000). O direito à privacidade só é pleno se o paciente for capaz de determinar quem pode acessar (ou acessou) seu prontuário, quando e com que finalidade. Ao médico, cabe o dever para com a confidencialidade, isto é, responsabilizar-se para que as informações dadas em confidência não sejam reveladas sem a autorização prévia do paciente (Sacardo & Fortes, 2000).

Esse dever, entretanto, não é um preceito absoluto. O próprio Código de Ética Médica, que no art. 102 proíbe o médico de *“revelar fato que tenha conhecimento em virtude do exercício de sua profissão”*, faz a ressalva de que tal revelação pode ser feita *“por justa causa, dever legal ou autorização expressa do paciente”* (Conselho Federal de Medicina, 1996, p. 27). Segundo França (2002), *“o que se proíbe é a revelação ilegal que tenha como motivação a má-fé, a levandade ou o baixo interesse”*. Um grande número de pessoas precisa acessar informações confidenciais em hospi-

¹ Violação do segredo profissional: art. 154. Revelar alguém, sem justa causa, segredo de que tem ciência em razão de função, ministério, ofício ou profissão, e cuja revelação possa produzir dano a outrem. Pena – detenção, de 3 (três) meses a 1 (um) ano, ou multa.

tais e clínicas diante dos notáveis progressos verificados nas áreas médica e da informática em saúde. Logo, o compromisso com a confidencialidade não se limita ao médico. Estende-se aos demais profissionais dos campos de saúde e administrativo, incluindo-se as secretárias e as recepcionistas que tenham contato com informações confidenciais (Francisconi & Goldim, 1998).

A troca de informações entre os membros da equipe de saúde com o auxílio de prontuários eletrônicos é um imperativo do trabalho de assistência à saúde atual. Entretanto, Sacardo & Fortes (2000) observam que a troca deve se limitar àquelas informações que cada profissional necessita para realizar suas atividades em benefício do cuidado ao paciente. Para Francisconi & Goldim (1998), ninguém tem o direito de usar as informações do prontuário livremente, salvo o próprio paciente, ou o representante legal, quando o paciente é menor de idade ou incapaz. O acesso só deve ser autorizado em função da necessidade profissional dos médicos e outros profissionais de saúde, assim como dos funcionários administrativos (Francisconi & Goldim, 1998). Opinião semelhante tem França (2002), para quem as instituições de saúde devem estabelecer um critério para o uso e a revelação do conteúdo do prontuário, para que se limitem ao essencial e ao justo fim invocado, omitindo-se ao máximo os detalhes pessoais.

Sacardo & Fortes (2000) recomendam a criação de barreiras para que pessoas, mesmo pertencentes ao quadro de funcionários de instituições de saúde, não possam acessar informações confidenciais sem que haja clara necessidade em benefício do paciente ou exigências administrativas regulamentadas. Francisconi & Goldim (1998) também recomendam o estabelecimento de “*medidas para evitar que pessoas sem qualquer envolvimento com o paciente, ou que não necessitem saber detalhes imprescindíveis à sua atividade profissional, venham a ter informações sobre o mesmo*”. Os autores afirmam que as instituições têm o dever de manter um sistema seguro de proteção ao conteúdo do PEP, com o contínuo aprimoramento das normas e das rotinas de controle de acesso e de identificação de usuários. Visão semelhante para restrição de acesso ao conteúdo do PEP também é defendida por entidades de regulação nacionais e internacionais, conforme consta em resoluções do Conselho Federal de Medicina (Conselho Federal de Medicina, 2002b; Conselho Federal de Medicina, 1996) e em documentos publicados pela Organização Pan-Americana de Saúde (Marin *et al.*, 2003; Rodrigues *et al.*, 2001); pelo Instituto Canadense para Informação de Saúde (Canadian Institute for Health Information, 1999);

pela Academia Nacional de Ciências e do Departamento de Serviços Humanos e de Saúde dos Estados Unidos da América (National Academy of Sciences, 1997; United States Department of Health and Human Services, 2003); e pelo Departamento de Saúde da Grã-Bretanha (National Health Service, 2002).

Em face do exposto, pode-se concluir o seguinte:

- O conteúdo do prontuário pertence ao paciente;
- As instituições e os profissionais de saúde são fiéis depositários do conteúdo fornecido, direta ou indiretamente, em confiança;
- As instituições e os profissionais são obrigados a não revelar as informações dadas em confiança sem autorização prévia do paciente, salvo quando as informações forem usadas em função da necessidade de cuidado ao paciente, por justa causa ou por dever legal;
- As instituições de saúde são obrigadas a implementar e aprimorar continuamente normas e rotinas de controle de acesso e de identificação de usuários, como parte de um sistema seguro de proteção ao conteúdo do PEP.

1.2 Aspectos computacionais do acesso ao PEP

Com o crescente uso do prontuário em sistemas de informação hospitalar (SIH), a responsabilidade em manter a privacidade do paciente não depende mais, exclusivamente, dos médicos e demais profissionais de saúde e administrativos. A responsabilidade também é compartilhada pelos sistemas, tornando-se um requisito básico para o uso ético e legal da tecnologia da informação na assistência à saúde (Al-Salqan, 1998).

O controle de acesso é importante para manter o sigilo do conteúdo do prontuário eletrônico do paciente. Objetiva limitar as ações realizadas por usuários legítimos para evitar a quebra da privacidade devido a acessos desnecessários ou não autorizados. A violação da privacidade pode acarretar danos ao paciente, como a perda ou a dificuldade de encontrar emprego, não conseguir um seguro de vida ou de saúde, ou até mesmo impor barreiras para sua aceitação na sociedade (Al-Salqan, 1998). Isso porque as informações do PEP podem indicar uso de drogas, doenças sexualmente transmissíveis, comportamento sexual, problemas psiquiátricos, predisposição genética para doenças, enfermidades graves, etc. Observou-se na seção 1.1 que há uma forte preocupação para que tais informações sejam usadas estritamente

para prover cuidados a saúde do paciente, devendo permanecer sigilosas em qualquer outra situação não prevista.

Entretanto, duas questões principais dificultam a concepção e a aplicação de modelos de autorização e controle de acesso para o prontuário eletrônico do paciente em ambientes de sistemas de processamento distribuído e aberto. Encontrar soluções exequíveis para essas questões é essencial para o uso em larga escala do PEP em grandes instituições de saúde (Beznosov *et al.*, 1999; Kaihara, 1998; National Academy of Sciences, 1997; Smith & Eloff, 1999).

A primeira questão refere-se a como lidar com o dilema entre liberar ou não o acesso ao PEP. Por um lado, o controle de acesso não deve, em nenhuma circunstância, prejudicar o atendimento ao paciente por negar acesso legítimo às informações e aos serviços requisitados pela equipe médica. Por outro lado, fora desse contexto, as informações do prontuário devem permanecer com acesso restrito, exceto para atender à vontade do paciente ou a determinações legais. Projetar modelos de autorização e controle de acesso capazes de implementar políticas de acesso conciliando dois lados antagônicos não é uma tarefa trivial.

Por exemplo, é indesejável impor uma política excessivamente proibitiva, impedindo um médico de acessar o prontuário de um paciente gravemente doente num pronto-socorro. A recomendação em tais casos é considerar a circunstância da emergência uma exceção para liberação do acesso (National Academy of Sciences, 1997; Smith & Eloff, 1999). Tampouco é desejável estabelecer uma política demasiadamente permissiva, facilitando a violação da privacidade do paciente.

É preciso dispor de um modelo adequado para uma definição precisa de políticas de acesso para o PEP, onde uma autorização é concedida ou negada no momento exato, de acordo com a necessidade e o direito do usuário, levando em conta as circunstâncias e os contextos existentes durante o acesso. O modelo deve ser conceitualmente simples, bem fundamentado, neutro quanto à política de acesso e capaz de resolver satisfatoriamente os conflitos inerentes às autorizações para acesso ao prontuário eletrônico do paciente. Ainda deve permitir uma implementação eficiente e a administração viável das políticas de acesso ao PEP, que demandam um controle detalhado para um grande número de usuários e recursos.

A segunda questão refere-se a como administrar tais políticas e a como impor o controle de acesso para o prontuário eletrônico do paciente. PEPs realísticos, efetivamente usados em instituições de saúde, são sistemas distribuídos, compostos por aplicações diversas, acessando bancos de dados distintos em plataforma hetero-

gêneas (Smith & Eloff, 1999). Em geral, os componentes do PEP são sistemas legados e têm seus próprios mecanismos para controle de acesso implementados de forma estanque.

Um requisito básico para atender à essa questão é adotar uma arquitetura de software aberta e distribuída, capaz de suportar, numa instituição, a administração da política de autorização e o controle de acesso de modo unificado e coerente, a partir de diferentes sistemas, em plataformas e linguagens de programação heterogêneas. O objetivo é conseguir interoperabilidade e portabilidade nos serviços de segurança, particularmente os de autorização e controle de acesso. A administração unificada da política de autorização de acesso para informações em saúde é recomendada por Beznosov (2000). Nela, apenas um grupo de usuários privilegiados pode estabelecer autorizações. O autor também recomenda isolar a lógica de autorização das aplicações, atendendo um princípio fundamental da Engenharia de Software, o *separation of concerns*. Assim, mudanças na lógica de autorização não implicam em alterações no código das aplicações. A administração unificada permite efetuar as mudanças independentemente das aplicações que solicitam as autorizações de acesso, mas o seu efeito é imediato.

1.3 Objetivo

O objetivo principal deste trabalho é propor um modelo de autorização contextual (MACA) para o control de acesso baseado em papéis (CABP), contemplando as necessidades discutidas nas seções anteriores, a fim de limitar o acesso ao prontuário eletrônico do paciente em ambientes abertos e distribuídos.

Como objetivos secundários, têm-se:

- Extensão, com a inclusão de autorizações contextuais, do modelo de referência proposto pelo *National Institute of Standards and Technology* (NIST) para o controle de acesso baseado em papéis (Ferraiolo *et al.*, 2001);
- Especificação de uma arquitetura de software baseada em padrões de processamento aberto e distribuído para integração do MACA;
- Implementação do MACA e da arquitetura para demonstrar que são exeqüíveis;
- Aplicação prática da implementação do MACA no controle de acesso ao PEP do Instituto do Coração do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo (InCor – HC-FMUSP).

1.4 Justificativa

O controle de acesso baseado em papéis tem características adequadas para atender os requisitos de controle de acesso para o PEP. Suporta o princípio de segurança denominado “princípio do privilégio mínimo”, também conhecido como “princípio da necessidade de saber/fazer”. Segundo o princípio, um usuário só deve possuir as autorizações indispensáveis, mínimas, para realizar as tarefas ou acessar os dados que ele *necessita* para exercer suas atribuições funcionais (Sandhu *et al.*, 1996). Essa característica é essencial no controle de acesso ao PEP, onde a necessidade é um fator determinante para conceder uma autorização, conforme discutido na seção 1.1.

A adoção do controle de acesso baseado em papéis do NIST a fim de regular o acesso a informações sobre a saúde de pacientes também é compatível com as exigências para segurança da informação baseadas na lei norte americana HIPAA (*Health Insurance Portability and Accountability Act of 1996*) (United States Department of Health and Human Services, 2003). O modelo de referência para CABP do NIST é fruto do significativo desenvolvimento dos aspectos teóricos e de implementação prática do CABP nos últimos dez anos (Ferraiolo *et al.*, 2001). A semântica do arcabouço teórico-conceitual do modelo está especificada na notação formal Z (Spivey, 1998). Por ser um modelo de referência, é um instrumento conceitual abstrato usado como base para modelos mais concretos, voltado para aplicações específicas. A especificação formal provê um conjunto padronizado e preciso de componentes funcionais e um vocabulário de termos que permitem a extensão consistente e bem fundamentada do modelo de referência.

A principal contribuição deste trabalho é a extensão do modelo de referência para CABP do NIST com a inclusão de autorizações contextuais para aplicação ao problema do controle de acesso ao prontuário eletrônico do paciente. Uma autorização contextual concede ou nega o acesso com base na avaliação de uma expressão lógica denominada de *regra de autorização*, durante uma solicitação de acesso. A regra é definida em termos de informações ambientais disponíveis em contextos, que podem ser preexistentes, denotando dados sobre o usuário corrente, a data, a hora e o local do acesso, a segurança da conexão do método de autenticação, dentre outros. A implementação do modelo também permite a programação de novos contextos, incorporando informações disponíveis nos sistemas que compõem o PEP, independente de plataforma ou linguagem de programação. O uso de contextos influencian-

do a decisão de liberar acesso ao PEP é recomendado pela National Academy of Sciences (1997), mas não é contemplado no modelo de referência para CABP do NIST.

As facilidades das autorizações contextuais conferem mais flexibilidade e poder expressivo para a definição de políticas de acesso para o PEP. Informações contextuais disponíveis no momento do acesso, como o relacionamento entre o usuário e o paciente, podem influenciar na decisão de autorizar a realização de uma tarefa. Por exemplo, uma política de acesso num hospital pode determinar que medicamentos só podem ser prescritos para pacientes internados ou em atendimento no pronto-socorro. Assim, uma autorização contextual é “ciente” do contexto (pacientes internados, local do acesso) associado com a operação a realizar (prescrever medicamento), possibilitando a especificação de políticas de acesso mais precisas e flexíveis, com granularidade fina, onde o acesso é concedido ou negado no momento exato, de acordo com o direito e a necessidade do usuário. Ademais, a capacidade do MACA de suportar contextos com alto nível de abstração possibilita a execução de políticas de acesso para o PEP que consideram os diferentes ambientes e as idiossincrasias da cultura de cada organização de saúde.

Ressalte-se que este trabalho considera um aspecto particular da segurança do PEP, o controle de acesso. Outros aspectos igualmente importantes para segurança global do PEP, como a confidencialidade de comunicação, a integridade, a auditoria, a não repudição, a disponibilidade, a segurança física e as políticas de segurança, são tratados de forma acessória, quando necessário.

A integração da implementação do MACA numa arquitetura baseada em padrões de processamento aberto e distribuído é necessária porque os atuais enfoques para integração dos componentes do PEP também se baseiam em padrões, conforme análise realizada por Grimson *et al.* (2000). Destacam-se os seguintes enfoques para essa integração: os serviços de *middleware*² e os serviços orientados à troca de mensagens.

Mostrar que é viável implementar eficientemente o MACA numa arquitetura aberta e distribuída é essencial para seu uso efetivo pelos sistemas que compõem o prontuário eletrônico do paciente em organizações de saúde reais. Este trabalho está particularmente interessado em descrever a implementação numa arquitetura baseada em serviços de *middleware* no âmbito do InCor – HC-FMUSP, onde há um forte acoplamento entre os segmentos que compõe o PEP. Segundo Grimson *et al.* (2000), tal enfoque é o preferido para integração intra-organizacional, ao passo que a

² Serviço de propósito geral que se situa entre plataformas e aplicações (Bernstein, 1997).

solução baseada em troca de mensagens deve ser a escolhida para integração interorganizacional, devido ao fraco acoplamento existente entre as instituições.

A aplicação prática da implementação do MACA no InCor – HC-FMUSP visa incrementar a segurança de acesso ao PEP na instituição e também viabilizar a sua disponibilidade futura na Internet. A experiência em um ambiente real de uma instituição de grande porte, com significativa heterogeneidade das aplicações que compõem o PEP, resulta em importantes lições, que contribuirão para aprofundar o conhecimento do problema do controle de acesso ao prontuário eletrônico.

1.5 Metodologia

O modelo de autorização proposto neste trabalho foi especificado na notação formal Z, atualmente em processo de padronização na *International Organization for Standardization* (ISO, 1999), a fim de manter coerência semântica com o modelo de referência para CABP do NIST, também definido em Z. A notação possui sintaxe e semântica formalmente definidas e os modelos especificados em Z usam lógica de primeira ordem e teoria dos conjuntos.

A arquitetura de software para implementação do MACA baseia-se nos seguintes padrões abertos: o protocolo de acesso à diretórios LDAP (*Lightweight Directory Access Protocol*) (Yeong *et al.*, 1995), especificado pela *Internet Engineering Task Force* (IETF); o serviço de segurança da arquitetura CORBA (SSC) (Object Management Group, 2001c); e o serviço de decisão para acesso a recursos (RAD – *Facility*) (Object Management Group, 2000b), ambos especificados pelo *Object Management Group* (OMG).

O LDAP especifica um serviço de diretórios capaz de armazenar informações numa estrutura hierárquica e distribuída, mas com transparência de localização para os usuários. É uma versão simplificada do serviço de diretórios distribuído X.500 (ITU-T, 1992), especificado para servir como base de informações para gerência de segurança (BIGS) em sistemas distribuídos (Muftic, 1993). As versões mais recentes de produtos de rede, bancos de dados e sistemas operacionais interoperam com o LDAP, possibilitando a unificação de dados para autenticação de usuário (*e. g.*, login e senha) nesses produtos e nas aplicações que compõem o PEP. Ademais, o LDAP possui implementações em múltiplas plataformas, otimizadas para consultas intensivas e suporte transparente para distribuição e replicação de dados.

O serviço de segurança da arquitetura CORBA foi escolhido porque o CORBA *Healthcare* (antigo CORBAmed) (Object Management Group, 2000a; Object Management Group, 2001a; Object Management Group, 2001b; Object Management

Group, 2001d) é a infra-estrutura de *middleware* recomendada no InCor – HC-FMUSP para integração das informações do PEP. O SSC especifica protocolos e interfaces padrões que provêem para clientes os seguintes serviços de segurança: autenticação; autorização e controle de acesso; auditoria; integridade e confidencialidade na comunicação entre clientes e servidores; e não repudição. Simplifica a implementação de segurança em aplicações distribuídas porque oculta a heterogeneidade entre plataformas, servindo como base para interoperabilidade entre serviços de segurança (Hartman *et al.*, 2001). O RAD – *Facility* supre deficiências do serviço de autorização do SSC em atender os requisitos de segurança de acesso para informações em saúde (Beznosov, 2000; Beznosov *et al.*, 1999). O *framework*³ do RAD – *Facility* permite à aplicação um controle de acesso detalhado, mas de um modo em que a lógica de autorização é separada da aplicação, com transparência em relação ao modelo de decisão efetivamente empregado.

A implementação do MACA foi realizada na linguagem de programação Java (Arnold *et al.*, 2000) porque, quando integrada à arquitetura CORBA, forma uma base robusta para o desenvolvimento de sistemas que suportam eficientemente a computação cliente-servidor (Lewandowski, 1998). Java oferece robustez com a gerência automática de memória e o suporte nativo à concorrência, além da portabilidade e da gerência de erros. CORBA complementa Java com a transparência de rede e com a tecnologia para distribuição e integração de objetos.

O *desenvolvimento evolucionário* (Sommerville, 1996, p. 11) foi o processo de software adotado para implementação do MACA e da arquitetura. Nesse processo, as atividades de especificação, projeto, implementação e validação ocorrem simultaneamente. Com base numa especificação preliminar, um protótipo inicial é projetado e implementado, para depois ser validado. O objetivo da validação é observar as deficiências do software em atender requisitos previamente estabelecidos, de modo que a especificação preliminar seja refinada e o processo se repita. Ao final de vários ciclos, espera-se obter um produto que atenda aos requisitos. O desenvolvimento evolucionário é recomendado para projetos onde há questões em aberto, de difícil solução, pois busca minimizar os riscos inerentes envolvidos (Philips, 1998).

Neste trabalho, os requisitos gerais para o problema do controle de acesso ao PEP são estabelecidos nas seções 1.1 e 1.2 e a especificação preliminar para solução do problema é o modelo de referência para CABP do NIST (Ferraiolo *et al.*, 2001).

³ Suporte para especificar uma arquitetura considerando a integração, a portabilidade, a interoperabilidade e a distribuição (Putman, 2001).

1.6 Organização do trabalho

- *Capítulo 2:* revisa os principais tipos de modelos para controle de acesso e comenta sua adequação para aplicação ao PEP. Introduz os conceitos e a terminologia da área de controle de acesso, relacionando-a com a segurança computacional. Apresenta sucintamente os tipos de modelos tradicionais e depois descreve, na notação formal Z, o modelo para controle de acesso baseado em papéis do NIST. Finalmente, revisa os modelos mais relevantes para o CABP levantados da literatura, com a identificação de suas características essenciais e das eventuais aplicações ao PEP;
- *Capítulo 3:* propõe e especifica, na notação Z, o modelo de autorização contextual (MACA) que estende o modelo de controle de acesso baseado em papéis do NIST. Ainda apresenta o conceito de contexto no MACA e descreve a linguagem de expressões lógicas usada para expressar regras de autorização. Finaliza ilustrando, com exemplos, a utilidade do MACA em atender políticas de acesso para o PEP e políticas para administração do CABP;
- *Capítulo 4:* define a arquitetura de software idealizada para integração do MACA, descrevendo seus aspectos de implementação e de desempenho. Especifica os componentes de software e de comunicação da arquitetura, com respectivos padrões, protocolos e APIs, discutindo características relevantes da integração. Expõe detalhes da representação, no serviço de diretórios LDAP, dos dados persistentes do MACA e do desenvolvimento em Java das suas funcionalidades, incluindo a implementação de contextos. Ainda mostra os resultados dos experimentos para avaliação de desempenho do MACA e apresenta e discute os resultados da sua aplicação prática no InCor – HC-FMUSP;
- *Capítulo 5:* discute as contribuições, as limitações e relação do MACA com os modelos revisados no Capítulo 2. Procura defender a validade do modelo em atender o problema do controle de acesso ao PEP e o compara com os modelos revisados, de acordo com as características essenciais identificadas. Por fim, faz considerações acerca das limitações do modelo e de possíveis extensões visando o seu aperfeiçoamento;
- *Capítulo 6:* traz as conclusões do trabalho, indicando os objetivos alcançados e as principais contribuições para área de conhecimento em questão.

2

Modelos de controle de acesso

“O conhecimento é, pois, uma aventura incerta que comporta em si mesma, permanentemente, o risco de ilusão e de erro.”

Edgar Morin, 1999

Este capítulo revisa os principais tipos de modelos para controle de acesso e comenta sua adequação para aplicação ao PEP. A seção 2.1 introduz os conceitos e a terminologia da área de controle de acesso, relacionando-a com a segurança computacional. As seções 2.2 e 2.3 consideram a aplicação no PEP dos modelos de controle de acesso discricionário e compulsório. A seção 2.4 introduz, na notação formal Z, o modelo de referência para controle de acesso baseado em papéis do NIST. Uma revisão sucinta dos modelos mais relevantes para o CABP levantados na literatura, com a identificação de suas características essenciais e de suas eventuais aplicações ao PEP, é apresentada na seção 2.5. A seção 2.6 traz as considerações finais do capítulo.

2.1 Conceitos e terminologia

A segurança em um sistema de informação visa protegê-lo contra ameaças à confidencialidade e à integridade das informações sob sua responsabilidade, bem como contra ameaças à disponibilidade de recursos e de informações (Brinkley & Schell, 1995; Joshi *et al.*, 2001). A violação da confidencialidade ocorre com a revelação não autorizada da informação, enquanto a violação da integridade ocorre pela modificação não autorizada dos dados. Já a violação da disponibilidade ocorre com a retenção não autorizada de informações ou recursos computacionais (Brinkley & Schell, 1995).

A prevenção contra as ameaças à confidencialidade e à integridade em sistemas de informação pode ser alcançada com a aplicação de métodos de controle de

acesso e com técnicas de criptografia e de segurança de redes (Brinkley & Schell, 1995; Joshi *et al.*, 2001). O controle de acesso é empregado em favor das propriedades de confidencialidade e de integridade das informações mantidas em sistemas de informação (Brinkley & Schell, 1995); enquanto a criptografia e a segurança de redes buscam assegurar tais propriedades nas informações transmitidas através de redes de comunicação (Stallings, 1999).

A meta do controle de acesso é limitar as ações que um usuário legítimo de um sistema de computação pode realizar (Sandhu & Samarati, 1994), com base nas autorizações aplicáveis ao mesmo no momento do acesso. Autorizações estabelecem os direitos de acesso⁴ que um sujeito⁵ pode exercer em um determinado objeto (recurso)⁶ computacional. O controle de acesso procura restringir o uso não autorizado dos objetos para evitar violações de segurança. A restrição do que pode ou não ser acessado é imposta diretamente às ações do usuário ou de programas que atuam em benefício deste.

Em geral, o controle de acesso exige a autenticação prévia do usuário que pretende usar os objetos protegidos (*Figura 1*). A autenticação verifica a legitimidade da identidade do usuário para o sistema de segurança, tipicamente por meio de uma identificação pessoal (login) e senha. O controle de acesso é imposto por um monitor de referência (*Figura 1*) que intermedeia todas as tentativas de um sujeito de executar operações de objetos protegidos. O monitor consulta um banco de dados de autorização para decidir se o sujeito tem permissão para executar a operação pretendida (Sandhu & Samarati, 1994). A auditoria registra cronologicamente dados sobre a atividade no sistema e os analisa para descobrir violações de segurança e diagnosticar suas causas (Sandhu & Samarati, 1996). O administrador de segurança é um usuário privilegiado, responsável pela configuração e gerência das autorizações, de acordo com política de acesso adotada. Em alguns modelos (*e.g.*, discricionário), os usuários também podem alterar segmentos do banco de autorizações relativos aos objetos que lhes pertencem.

2.1.1 O papel da autenticação

A correta identificação do usuário pelo serviço de autenticação é pré-condição para um controle de acesso eficaz. Caso contrário, um usuário pode, mediante uma fraude, assumir a identidade de outrem, com privilégios mais poderosos, e ter acesso a informações ou realizar ações à que não tem direito. A utilização de um login

⁴ Os termos direito de acesso, privilégios e permissões são usados neste texto indistintamente.

⁵ Um sujeito pode ser um usuário humano ou algum agente autônomo que atua em auxílio deste.

⁶ Recursos ou objetos computacionais oferecem os meios para revelação ou alteração da informação.

Figura 1 – Esquema do controle de acesso e sua relação com outros serviços de segurança (Sandhu & Samarati, 1994).

individualizado, com senha estática, aliada a uma política de gerência de senhas (*e.g.*, prazo de expiração, tamanho mínimo, uso de senhas não triviais), é o mínimo recomendado para identificação de usuários que acessam informações do PEP (Conselho Federal de Medicina, 2002b; United States Department of Health and Human Services, 2003). Entretanto, em face da fragilidade desse esquema, particularmente em ambientes distribuídos, recomenda-se a adoção de métodos de autenticação mais robustos, como aqueles baseados na identificação biométrica (Jain *et al.*, 2000), em certificados digitais (ITU-T, 1993) e cartões inteligentes (Shelfer & Procaccino, 2002) ou em senhas dinâmicas (Lamport, 1981). Reconhece-se a importância da autenticação para a segurança do PEP no presente trabalho, mas o foco de atenção é no controle de acesso.

2.1.2 O papel da auditoria

A ciência, pelos usuários, da realização contínua de análises dos registros de auditoria, como parte de uma política institucional, atua como forte inibidor para o uso indevido das informações presentes no PEP (National Academy of Sciences, 1997). A auditoria também é importante nas investigações após incidentes e pode ser realizada por órgãos internos a uma instituição ou por entidades externas independentes.

Entretanto, ela é inócua se não houver a aplicação rigorosa das sanções previstas em legislação para os casos comprovados de violações da confidencialidade e da integridade do conteúdo PEP. A identificação e a comprovação dos casos requerem auditorias efetivas, a autenticação correta dos usuários, a garantia da autenticidade e da integridade dos registros de auditoria e o uso de ferramentas de software que os analisem em busca de abusos e outras violações. Ademais, a existência de um serviço de auditoria é essencial para informar o paciente quem acessou o seu prontuário eletrônico e com que propósito. Nessa situação, o paciente também atua como auditor do seu PEP, podendo detectar falhas de segurança, fraudes ou acesso abusivo (Anderson, 1996).

2.1.3 O papel da política de acesso

Segundo Sandhu & Samarati (1994), uma política de acesso define diretrizes de alto nível que determinam como o acesso é controlado e como as decisões de autorização de acesso são estabelecidas. Em geral, tais políticas são expressas em linguagem natural, o que dá margem à ambigüidades e incompletudes. Brinkley & Schell (1995) definem o conceito de “política técnica” como sendo a interpretação da “política de acesso de alto nível” em termos dos relacionamentos formais entre as entidades ativas (sujeitos), as entidades passivas (objetos) e os modos de acesso (operações) permitidos. Os relacionamentos especificam autorizações de acesso que são precisa e completamente definidas, passíveis de interpretação não ambígua.

Os modelos de controle de acesso, portanto, definem uma linguagem para expressar políticas técnicas que representam uma política de acesso de alto nível. No que diz respeito ao controle de acesso, um sistema será seguro na medida em que satisfaça à sua política de acesso.

Em geral, não há modelos melhores do que outros, mas sim modelos mais ou menos adequados ao tipo de política que uma instituição deseja para o acesso aos recursos computacionais. O PEP requer, dos modelos de controle de acesso, a capacidade de expressar políticas de acesso *ad hoc*, ou seja, políticas formuladas com o único objetivo de atender as necessidades de controle de acesso de uma instituição, levando em conta o seu ambiente e a sua cultura. Por exemplo, uma instituição define uma política de acesso *ad hoc* para o PEP quando determina que *enfermeiras somente podem ver registros de pacientes internados em sua enfermária ou que lá estiveram nos últimos 30 dias* ou então que *médicos que são membros de uma equipe somente podem ver registros de pacientes sob cuidados de algum membro da equi-*

pe. Segundo Denley & Weston Smith (1999), políticas *ad hoc* são particularmente necessárias em estabelecimentos de atendimento secundário, onde o paciente recebe cuidados de equipe multiprofissional e multidisciplinar, inclusive em emergências.

Diferentes modelos de controle de acesso, capazes de suportar políticas distintas, podem coexistir de forma combinada para proporcionar a proteção desejada. Dentre os modelos existentes, destacam-se os tradicionais controle de acesso discricionário (CAD) e controle de acesso compulsório (CAC). Recentemente, o controle de acesso baseado em papéis vem recebendo atenção como uma alternativa para os modelos tradicionais (Joshi *et al.*, 2001).

2.2 Considerações sobre o controle de acesso discricionário

A posse pelos usuários, dos objetos protegidos, caracteriza o controle de acesso discricionário. No CAD, o acesso a um objeto é controlado com base na identidade do usuário e nas autorizações que especificam seus privilégios, *e. g.*, ler, escrever, executar. As autorizações são definidas em listas de controle de acesso localizadas no objeto protegido (Sandhu & Samarati, 1994). Elas podem conceder privilégios para um usuário específico ou para um grupo de usuários. Quando um sujeito tenta efetuar uma operação num objeto, o monitor de referência verifica na lista de controle de acesso do objeto se existe alguma autorização que concede o privilégio. Existindo a autorização, o acesso é concedido, caso contrário, ele é negado.

O proprietário do objeto é livre para conceder ou revogar privilégios de acesso a outros usuários, o que torna o CAD bastante flexível e amplamente utilizado (Joshi *et al.*, 2001). Entretanto, possibilita que um usuário autorizado a acessar um objeto faça uma cópia do mesmo e o repasse para outros usuários sem a permissão explícita do proprietário (Sandhu & Samarati, 1994). O problema é que no CAD, em geral, não é decidível determinar se existe um estado alcançável no sistema no qual um usuário específico possui um privilégio que anteriormente não possuía (Brinkley & Schell, 1995; Joshi *et al.*, 2001). Extensões recentes propostas para o CAD visam resolver este problema, mas ainda estão em estágio de desenvolvimento teórico, com pouco ou nenhum resultado experimental (Joshi *et al.*, 2001).

Ademais, por sua natureza assimétrica e descentralizada, o CAD dificulta a administração unificada da política de acesso. Por exemplo, para determinar todos os privilégios de um usuário é preciso examinar as listas de controle de acesso de cada objeto num sistema. Outra dificuldade ocorre quando se pretende remover privilégios parcial ou totalmente. Além de ser necessário visitar todas as listas, o pro-

prietário do objeto deve conceder o privilégio de modificação da lista para quem administra a política de controle de acesso.

A administração da política se agrava quando se considera que os objetos protegidos são sistemas distribuídos e heterogêneos, o caso do prontuário eletrônico do paciente. Acrescente-se a numerosa quantidade de usuários em grandes hospitais, com diferentes privilégios, e a administração unificada da política de acesso ao PEP torna-se difícil na prática. Isso decorre do grande número de associações diretas entre usuários e autorizações para gerenciar nas listas de controle de acesso, da ordem de $\sum_i^{n_{fo}} (|U_i| \cdot |A_i|)$, onde U_i é o conjunto de indivíduos com uma função organizacional i , A_i é o conjunto de autorizações necessárias para exercer a função e n_{fo} é o número de funções (Ferraiolo *et al.*, 1999). Como consequência, têm-se um elevado custo de gerência da política de acesso e o risco de manter-se privilégios residuais indevidos para usuários. Embora a utilização de grupos de usuários minimize esses problemas, cabe ressaltar que grupos são usualmente empregados para categorizar usuários, mas não autorizações de funções organizacionais específicas (Sandhu, 1996). É sabido que os usuários tendem a mudar de funções numa organização numa frequência maior que as mudanças de autorizações necessárias à tais funções (Ferraiolo, 1993). Logo, a troca de função de um usuário implica na atribuição e na revogação de autorizações, mais sujeitas a erros, requerendo mais capacidade técnica e de conhecimento organizacional, e que não se pode resolver exclusivamente pelo remanejamento dos grupos do usuário (Sandhu *et al.*, 1996).

A despeito do arrazoado anterior, quando se considera que o paciente é o proprietário do prontuário, o CAD passa a ser uma alternativa a considerar. Um exemplo é a política discricionária para sistemas de informações clínicas proposta por Anderson (1996), baseada em listas de controle de acesso associadas a informações identificadas do paciente. Nela, o paciente é o proprietário da informação, mas é o seu médico o responsável pela alteração da lista para incluir novos usuários com diferentes privilégios, de acordo com a determinação do paciente. A política mostra-se útil quando o paciente tem um “médico de família” num serviço de cuidado primário à saúde efetivo. Nesses casos, as listas de controle de acesso são relativamente fáceis de administrar porque a quantidade de profissionais que precisam acessar as informações protegidas é pequena. Denley & Weston Smith (1999), porém, observam que a administração de políticas baseadas em listas de controle de acesso não são sustentáveis quando usadas em sistemas de informações clínicas de hospitais que realizam cuidado secundário à saúde. Os autores reconhecem a necessidade do con-

trole de acesso basear-se nas funções desempenhadas por profissionais com competências e responsabilidades distintas, levando-se em consideração políticas de acesso *ad hoc*.

Isso desfavorece a aplicação do CAD para o controle de acesso ao PEP, mesmo quando se considera que o paciente é o proprietário do seu prontuário. A questão é como utilizar o CAD num âmbito institucional, conciliando as necessidades do paciente com a administração viável da política de acesso.

2.3 Considerações sobre o controle de acesso compulsório

O controle de acesso compulsório confina o fluxo da informação numa única direção, numa hierarquia de acesso, para assegurar a confidencialidade e a integridade da informação e baseia-se na classificação de sujeitos e objetos (Joshi *et al.*, 2001). O nível da classe de um objeto reflete a importância da informação mantida no objeto, isto é, o potencial de dano a ser causado pela revelação ou alteração não autorizada da informação (Sandhu & Samarati, 1994).

As classes do CAC formam um conjunto parcialmente ordenado. Nos ambientes militares e governamentais, elas historicamente são: *ultra-secreto* > *secreto* > *confidencial* > *liberado* (Sandhu, 1993; Sandhu & Samarati, 1994). Diz-se que cada classe domina a si mesma e todas as outras abaixo dela (Sandhu, 1993; Sandhu & Samarati, 1994). Os acessos de sujeitos a objetos somente são liberados pelo monitor de referência quando relacionamentos específicos são satisfeitos entre as classes associadas a ambos, tipicamente, os seguintes: 1) a classe do sujeito deve dominar a classe do objeto que ele pretende ler; 2) a classe do sujeito deve ser dominada pela classe que ele pretende usar para gravar (Sandhu, 1993; Sandhu & Samarati, 1994). Por exemplo, um sujeito da classe *secreto* pode ler qualquer objeto de uma classe igual ou inferior a sua, mas quando grava informações lidas ou criadas, essas só podem ser de uma classe igual ou superior a dele. Informações sensíveis gravadas por um sujeito da classe *secreto* não podem ser repassadas para usuários de uma classe de acesso menos privilegiada, mesmo quando se fazem cópias da informação. As classes e os relacionamentos vistos anteriormente provêm confidencialidade. Outras classes e relacionamentos podem ser estabelecidos para oferecer integridade e para minimizar problemas de conflitos de interesses (Sandhu, 1993).

O CAC também não impõe dificuldades para administração da política de acesso. As classes dos sujeitos são atribuídas por administradores de segurança, enquanto as classes dos objetos são determinadas compulsoriamente pelo sistema, de

acordo com as classes dos sujeitos que os criam. Os administradores são usuários privilegiados que centralizam a gerência da política de acesso, sendo os únicos com poder de modificar arbitrariamente a classe de um objeto.

Enfim, o CAC provê mecanismos robustos que asseguram com alto grau de confiança a implementação de políticas de controle do fluxo da informação, incapazes de se realizar com o controle de acesso discricionário. Foi adrede desenvolvido para impor as severas políticas de segurança para informações sensíveis das áreas militares e governamentais. Em virtude disso, não possui a flexibilidade necessária para suportar políticas de acesso *ad hoc*, essenciais para um adequado controle de acesso às informações do PEP. Atender casos excepcionais ou que considerem o contexto na liberação do acesso a informações clínicas pode violar os relacionamentos entre as classes do CAC, com a conseqüente violação do modelo. A preservação da integridade do modelo, por outro lado, não pode causar prejuízo ao paciente, seja por negar o acesso em casos de precisão durante um atendimento, ou por concedê-lo em situações onde não havia necessidade. Ademais, a implementação dos mecanismos que satisfazem o CAC é uma tarefa difícil, particularmente em ambientes distribuídos (Brinkley & Schell, 1995; Joshi *et al.*, 2001).

2.4 O controle de acesso baseado em papéis

As características do controle de acesso baseado em papéis favorecem a execução de políticas de acesso para o PEP, particularmente em organizações de saúde de grande porte, em níveis secundário e terciário. O CABP regula o acesso dos usuários aos objetos protegidos com base nos papéis que eles exercem numa organização (Sandhu & Samarati, 1994). Os papéis denotam funções que descrevem a autoridade e a responsabilidade concedidas aos usuários (Sandhu *et al.*, 1996). No CABP, autorizações para acessar recursos não são associadas diretamente a usuários, mas para papéis, de acordo com as atribuições pertinentes.

É um enfoque para controle de acesso adequado às organizações de saúde, que se destacam por possuírem processos de negócio complexos, operados por numerosos profissionais com autoridades e responsabilidades distintas, mediante os diversos sistemas de informação que compõem o PEP, como admissão/internação, diagnóstico, prescrição médica, programação cirúrgica, registro de procedimentos, laudos, agenda e contas de paciente. Com o CABP, se um usuário é médico e exerce a função de diretor clínico, ele tem os papéis *Médico* e *Diretor Clínico* associados. Já auxiliares administrativos com a função de registrar a admissão/internação de paci-

entes têm os papéis *Auxiliar de Registro de Saúde* ou *Analista de Registro de Saúde* associados. Consequentemente, os direitos deles de acesso ao PEP são os definidos para os papéis, de acordo com a necessidade de saber/fazer inerente a autoridade e a responsabilidade de cada função exercida.

O controle de acesso baseado em papéis vem recebendo crescente atenção como um enfoque generalizado para o controle de acesso porque incorpora importantes e reconhecidas vantagens (Joshi *et al.*, 2001):

- É politicamente neutro (Osborn *et al.*, 2000), ou seja, oferece meios para expressar diferentes políticas de acesso, em vez de embutir uma política particular, como no CAC. Pode ser configurado para suportar, tanto as políticas *ad hoc*, necessárias para regular o acesso ao PEP (v. seção 2.1.3), quanto as políticas tradicionais do controle de acesso discricionário e compulsório (Osborn *et al.*, 2000);
- Suporta o princípio do privilégio mínimo, pelo qual um usuário só tem os privilégios necessários para desempenhar suas funções, evitando-se, assim, que ele possa efetuar ações desnecessárias ou potencialmente danosas e que são autorizadas apenas como um efeito colateral (Ferraiolo *et al.*, 1999). A capacidade de atender esse princípio é condição básica para satisfazer as recomendações de liberação do acesso às informações clínicas de pacientes discutidas na seção 1.1;
- Permite a implementação do conceito da “separação de responsabilidades” (SR), cujo objetivo é reduzir as chances de fraude ou dano acidental decorrentes da demasiada concentração de poder numa única pessoa. A SR distribui para vários usuários a responsabilidade e a autoridade para realizar uma tarefa suscetível de fraudes ou mau uso, de modo que um indivíduo não seja poderoso o suficiente para efetuarla sem um conluio (Ferraiolo *et al.*, 2001). A idéia é adotar a SR para minimizar a possibilidade de um usuário realizar ações em que há conflito de interesses.

O CABP ainda facilita a administração da política de acesso por interpor os papéis entre usuários e autorizações. Assim, quando um usuário muda de função, basta alterar os papéis atribuídos para ele. Por outro lado, como papéis são usados para categorizar autorizações (Sandhu, 1996) de funções organizacionais específicas, quando as atribuições e responsabilidades das funções mudam, basta alterar as autorizações associadas aos respectivos papéis. O desacoplamento das associações dire-

tas entre usuários e autorizações reduz a complexidade e o custo administrativo do CABP, quando comparado com o CAD. Enquanto no CAD o número de associações para gerenciar é da ordem de $\sum_i^{n_{fo}} (|U_i| \cdot |A_i|)$, no CABP ele é reduzido para $\sum_i^{n_{fo}} (|U_i| + |A_i|)$, onde U_i é o conjunto de indivíduos com uma função organizacional i , A_i é o conjunto de autorizações necessárias para exercer esta função e n_{fo} é o número de funções (Ferraiolo *et al.*, 1999).

Isso viabiliza a administração de uma política de acesso detalhada, com granularidade fina, para um grande número de usuários e recursos, abrangendo coerentemente os múltiplos sistemas que compõem o PEP. Ferraiolo *et al.* (1999) afirmam que um maior detalhamento do controle de acesso fortalece a aplicação do princípio do privilégio mínimo. Por exemplo, em vez adotar uma política abrangente, liberando o acesso a todos os prontuários para usuários com o papel *Médico*, poderia-se aplicar uma política pormenorizada, onde a liberação só seria permitida para médicos com vínculo assistencial com o paciente, tal como, ser membro da equipe que o assiste. Conseqüentemente, a quantidade de PEPs passíveis de acesso sem necessidade ficaria reduzida. Observe-se, ainda, que políticas detalhadas exigem um esforço de auditoria e de monitoração menor do que o esforço necessário quando se executa políticas de acesso mais abrangentes (Ferraiolo *et al.*, 1999).

Ademais, o CABP permite colocar a administração da política de acesso na perspectiva de um modelo organizacional (Kern *et al.*, 2003; Oh & Park, 2001; Oh & Sandhu, 2002; Schaad *et al.*, 2001) e não em função de plataformas tecnológicas específicas. Em geral, os termos usados para designar os papéis e os recursos protegidos pertencem ao vocabulário da cultura organizacional. Assim, a tarefa de atribuir ou remover papéis de um usuário para conceder-lhe os privilégios das novas funções que passe a exercer pode ser realizada por um profissional da área de recursos humanos com um mínimo de conhecimento em tecnologias da informação. Os procedimentos para remoção de privilégios ou bloqueio de contas para acesso ao PEP, quando o vínculo de um usuário com um estabelecimento de saúde se encerra, podem ser feitos de forma confiável e com pouco esforço, mesmo quando há alta rotatividade de pessoal. Isso é particularmente útil em hospitais escola, onde o número de usuários do PEP com vínculo temporário não é desprezível.

Por outro lado, o uso de termos habituais para designar os recursos e os papéis facilita a definição de autorizações, exigindo dos formuladores da política de acesso maior conhecimento organizacional e menor capacidade técnica. Favorece,

ainda, para a contribuição efetiva dos gestores das organizações de saúde na formulação e na administração das políticas de acesso ao PEP, e isso é imperativo, segundo Huston (2001). Tal facilidade existe porque os termos familiares aproximam a linguagem usada para expressar as políticas técnicas da linguagem usada para expressar as políticas de acesso de alto nível (v. seção 2.1.3). Ferraiolo *et al.* (1999) observam que o fator limitante para impor efetivamente políticas de acesso não é tanto a capacidade do modelo, mas sim a impraticabilidade de impor as políticas através da administração direta de mecanismos de controle de acesso com baixo nível de abstração, como os encontrados no CAD ou no CAC.

No CABP, os recursos protegidos são representados de forma abstrata, independente de localização e forma concretas. Em adição, as autorizações não são armazenadas junto aos objetos a que se referem, viabilizando a implementação de funções administrativas para revisão da política de acesso. Com tais funções, administradores podem saber todos os usuários associados a um papel, todos os papéis associados a um usuário, bem como, todas as autorizações de um usuário particular. Dado um objeto protegido, também é possível saber todos os usuários que podem acessá-lo.

A administração da política no CABP pode ser realizada de forma unificada por usuários privilegiados, isto é, usuários que possuem papéis administrativos associados. Utiliza-se o próprio CABP para controlar o acesso às funções administrativas do CABP, havendo vários modelos propostos para esse fim (Kern *et al.*, 2003; Oh & Sandhu, 2002; Sandhu & Munawer, 1999; Sandhu, *et al.*, 1999). Fundamenta-se na descentralização da autoridade e da responsabilidade administrativas por múltiplos papéis para viabilizar a gerência e evitar a existência do “superusuário” através da separação de responsabilidades. Por exemplo, as tarefas de criar contas de usuários, de atribuir papéis, de gerenciar papéis e autorizações seriam realizadas por usuários com papéis mutuamente exclusivos, i. e., sem usuários em comum. Assim, quem cria contas não pode atribuir papéis e vice-versa. Por outro lado, quem gerencia papéis e autorizações ficaria proibido de criar contas ou atribuir papéis.

Cabe ressaltar que o controle de acesso baseado em papéis não é uma panacéia. Oferece os meios para atender importantes princípios de segurança e implementar uma variedade de políticas de acesso requeridas pelo PEP, mas não é capaz de impô-los *per se*. Exige um planejamento adequado. Por exemplo, assegurar a aderência ao princípio do privilégio mínimo é, em grande parte, um desafio administrativo que requer a identificação das funções organizacionais, a especificação correta

dos privilégios necessários para executar cada função e a limitação do poder de cada usuário estritamente de acordo com sua função (Ferraiolo *et al.*, 1999). Em tese, administradores poderiam configurar um modelo CABP para violar o princípio do privilégio mínimo, chegando até mesmo a permitir a existência de superusuários pela não utilização do conceito da separação de responsabilidades. Acresça-se que o CABP não é capaz de implementar políticas que imponham obrigações para usuários, mas, tão somente, permissões e proibições. Já determinar o grau de abstração usado para representar os objetos protegidos depende da cooperação entre os formuladores da política de acesso e os responsáveis pelo desenvolvimento desses objetos.

O modelo de referência proposto pelo NIST, apresentado na subseção seguinte, visa normatizar o escopo, os conceitos e a terminologia do CABP a partir de um consenso de pesquisadores da área (Ferraiolo *et al.*, 2001). É um instrumento conceitual abstrato, aberto a extensões, usado como base para o desenvolvimento de modelos mais concretos, que podem disponibilizar apenas as características básicas do CABP, ou, então, características mais avançadas, como a separação de responsabilidades. A forma como tais características são implementadas também pode variar entre diferentes implementações do CABP.

2.4.1 O padrão para CABP proposto pelo NIST

O padrão proposto pelo NIST para o CABP compreende um modelo de referência e uma especificação funcional (Ferraiolo *et al.*, 2001). O modelo de referência estabelece rigorosamente os conjuntos de entidades, os relacionamentos e as restrições que constituem o CABP. Visa definir um vocabulário comum de termos para especificar coerentemente os requisitos e determinar o escopo das características do CABP incluídas no padrão. A especificação funcional define os requisitos dos procedimentos administrativos para criação e manutenção dos conjuntos de entidades e dos relacionamentos do CABP; as funções de revisão para consultas administrativas; e as funções de sistema para criar e gerenciar sessões de usuários e efetuar decisões de autorização de acesso.

2.4.1.1 O modelo de referência

Esta seção apresenta, na notação formal Z (Spivey, 1998), a especificação do modelo de referência para o CABP, estendido no MACA (capítulo 3) pela introdução de autorizações contextuais. O modelo de referência está organizado nos três componentes relacionados na lista abaixo e especificados nas subseções a seguir. Cada um dos componentes é definido por um conjunto básico de conjuntos de entidades; um con-

junto de relações envolvendo os conjuntos de entidades; um conjunto de restrições imposto aos relacionamentos; e um conjunto de funções mapeando membros de um conjunto de entidades para membros de outros conjuntos de entidades.

- *CABP Básico*: incorpora os aspectos que caracterizam um modelo de controle de acesso como sendo um modelo para CABP. O essencial é que usuários estejam associados a papéis e que esses estejam associados às autorizações. Requer que as relações usuário-papel e papel-autorização sejam do tipo muitos-para-muitos. Introduce ainda o conceito de sessão, que permite, ao usuário que a iniciou, a ativação/desativação simultâneas de mais de um papel;
- *CABP Hierárquico*: adiciona ao CABP Básico requisitos para suportar hierarquias de papéis. Matematicamente, uma hierarquia é uma ordem parcial que define uma relação de responsabilidades entre papéis. Em geral, papéis com maior responsabilidade (mais específicos) adquirem as autorizações de papéis com menor responsabilidade (mais genéricos). Por outro lado, papéis com menor responsabilidade incorporam os usuários dos papéis com maior responsabilidade. O CABP Hierárquico admite duas formas possíveis de hierarquia: uma genérica, capaz de suportar uma ordem parcial arbitrária; e outra restrita, podendo suportar apenas estruturas como árvores ou árvores invertidas;
- *CABP Restrito*: adiciona ao CABP Básico ou Hierárquico relações de separação de responsabilidades para impor restrições que ajudem a evitar que um usuário seja autorizado a executar operações associadas a papéis com conflitos de interesses. No CABP Restrito, a separação de responsabilidades pode ser estática ou dinâmica. A SR estática (SRE) atua na relação usuário-papel e na hierarquia de papéis, enquanto a SR dinâmica (SRD) atua durante a ativação/desativação de papéis numa sessão de usuário.

CABP Básico

Os conjuntos de entidades e as relações integrantes do CABP Básico (Ferraiolo *et al.*, 2001) estão ilustradas na *Figura 2*. O modelo possui cinco conjuntos de entidades principais: *usuários*, *papéis*, *obs* (conjunto de objetos), *ops* (conjunto de operações) e *sessões*. As entidades são representadas no modelo formal pelo “tipo básico” seguinte:

Figura 2 – Esquema do modelo de referência para CABP do NIST (Ferraiolo *et al.*, 2001).

[NOME].

Na notação Z, um tipo básico define um conjunto de objetos usados numa especificação para os quais não se assume uma estrutura particular (Potter *et al.*, 1991). Tais objetos são tratados como atômicos na especificação. O tipo básico *NOME* representa os identificadores possíveis das entidades principais do CABP Básico (Ferraiolo *et al.*, 2001). Ou seja, representam-se as entidades no modelo formal por meio de seus nomes. O esquema Z a seguir define os conjuntos das entidades principais do modelo:

<i>Conjuntos de Entidades</i>	
<i>usuários</i> :	$\mathbb{P} \text{ NOME}$
<i>papeis</i> :	$\mathbb{P} \text{ NOME}$
<i>obs</i> :	$\mathbb{P} \text{ NOME}$
<i>ops</i> :	$\mathbb{P} \text{ NOME}$
<i>sessoes</i> :	$\mathbb{P} \text{ NOME}$
$\forall c_1, c_2: \{usuários, papeis, obs, ops, sessoes\} \bullet c_1 \cap c_2 = \emptyset$	

Um esquema em Z, em geral, está dividido em duas partes: uma superior, onde são declaradas variáveis, e outra inferior, onde se estabelecem os relacionamentos entre os valores das variáveis (Spivey, 1998). O Anexo C reproduz algumas das definições matemáticas usadas nos esquemas apresentados nos capítulos 2 e 3 com o objetivo de facilitar o entendimento. Porém, a referência completa à notação é encontrada em Spivey (1998). O esquema *Conjuntos de Entidades* descreve o espaço do estado dos

conjuntos de elementos do modelo do CABP Básico e suas variáveis representam as observações possíveis que se podem fazer do estado:

- *usuarios* é o conjunto dos nomes que representam os usuários que têm o acesso controlado através do CABP. Um usuário é um ser humano ou algum agente autônomo que atua em benefício deste;
- *papeis* é o conjunto dos nomes que representam papéis. Um papel corresponde a uma função ou cargo numa organização, com atribuições bem definidas, pertinentes à autoridade e à responsabilidade conferidas aos detentores do papel;
- *obs* é o conjunto dos nomes que representam objetos protegidos. Um objeto é uma entidade que contém ou processa informações através da execução de operações associadas. Os arquivos em um sistema operacional são exemplos de objetos protegidos;
- *ops* é o conjunto dos nomes que representam operações. Uma operação é uma imagem executável de um programa e, quando invocada, executa alguma função sobre um objeto para um usuário. Por exemplo, em objetos que são tabelas de um sistema de gerência banco de dados, operações típicas são inserir, atualizar, excluir e consultar;
- *sessoes* é o conjunto de nomes que representam sessões. Uma sessão corresponde a um momento particular quando um usuário se conecta a um sistema para realizar alguma atividade que tem o acesso controlado através do CABP.

Representações e interpretações mais concretas das entidades ficam em aberto para extensões ou implementações do modelo de referência. Os relacionamentos estabelecidos no predicado da parte inferior dos esquemas de estado em Z são chamados de “invariante do sistema” (Spivey, 1998). É invariante porque define propriedades que devem ser verdadeiras em todo estado possível do sistema. No esquema *ConjuntosdeEntidades*, o invariante afirma que os conjuntos de entidades de usuários, papéis, objetos, operações e sessões são disjuntos, isto é, não podem compartilhar os mesmos nomes.

As relações e as funções de mapeamento entre os membros dos conjuntos de entidades são especificados no esquema a seguir, que complementa o modelo do CABP Básico:

CABP_Basico*ConjuntosdeEntidades* $up: usuarios \leftrightarrow papeis$ $usuarios_associados: papeis \rightarrow P\ usuarios$ $autorizacoes: ops \leftrightarrow obs$ $pa: papeis \leftrightarrow autorizacoes$ $autorizacoes_associadas: papeis \rightarrow P\ autorizacoes$ $op: autorizacoes \rightarrow ops$ $ob: autorizacoes \rightarrow obs$ $sessoes_do_usuario: usuarios \rightarrow P\ sessoes$ $usuario_da_sessao: sessoes \rightarrow usuarios$ $papeis_da_sessao: sessoes \rightarrow P\ papeis$ $autorizacoes_da_sessao: sessoes \rightarrow P\ autorizacoes$ $\forall p: \text{dom } usuarios_associados \bullet usuarios_associados(p) = \{u \in usuarios \mid u \mapsto p \in up\}$ $\forall p: \text{dom } autorizacoes_associadas \bullet$
 $autorizacoes_associadas(p) = \{a \in autorizacoes \mid p \mapsto a \in pa\}$ $\forall a: \text{dom } op \bullet op(a) = first(a)$ $\forall a: \text{dom } ob \bullet ob(a) = second(a)$ $\forall s: \text{dom } usuario_da_sessao \bullet s \in sessoes_do_usuario(usuario_da_sessao(s))$ $\forall s: \text{dom } papeis_da_sessao \bullet$
 $papeis_da_sessao(s) \subseteq \{p \in papeis \mid usuario_da_sessao(s) \mapsto p \in up\}$ $\forall s: \text{dom } autorizacoes_da_sessao \bullet$
 $autorizacoes_da_sessao(s) \subseteq \{a \in autorizacoes_associadas(p) \mid p \in papeis_da_sessao(s)\}$

A primeira declaração do esquema acima inclui o esquema *ConjuntosdeEntidades* no esquema *CABP_Basico*. Essa declaração tem o efeito de incluir as declarações de *ConjuntosdeEntidades* nas declarações do esquema *CABP_Basico* e anexar o predicado do invariante de *ConjuntosdeEntidades* ao predicado do invariante *CABP_Basico*. Os predicados dos invariantes são anexados como uma conjunção. As relações e as funções de mapeamento introduzidas no esquema *CABP_Basico* são descritas a seguir:

- A relação *up* (Figura 2) estabelece as associações entre usuários e papéis como um relacionamento muitos-para-muitos, isto é, um usuário pode ter mais de um papel atribuído e um papel pode ter mais de um usuário vinculado;
- A função *usuarios_associados* mapeia um papel *p* para o conjunto dos usuários associados a esse papel na relação *up*, conforme estabelecido no invariante do esquema;
- A relação *autorizacoes* (Figura 2) define autorizações como um relacionamento muitos-para-muitos entre operações e objetos. Semantica-

mente, uma autorização é um consentimento para executar uma operação relacionada a um ou mais objetos de um sistema. O padrão proposto para CABP do NIST inclui somente autorizações positivas, isto é, autorizações que concedem o acesso. Entretanto, o modelo deixa em aberto a possibilidade de incorporar também autorizações negativas em suas extensões e implementações. Uma autorização negativa proíbe explicitamente a execução de uma operação relacionada a um ou mais objetos de um sistema;

- A relação *pa* (Figura 2) estabelece as associações entre papéis e autorizações como um relacionamento muitos-para-muitos, isto é, um papel pode ter mais de uma autorização associada e uma autorização pode estar vinculada a mais de um papel;
- A função *autorizacoes_associadas* mapeia um papel *p* para o conjunto das autorizações vinculadas a esse papel na relação *pa*, conforme estabelecido no invariante do esquema;
- A função auxiliar *op* retorna a operação de uma dada autorização, conforme estabelecido no invariante do esquema;
- A função auxiliar *ob* retorna o objeto de uma dada autorização, conforme estabelecido no invariante do esquema;
- A função *sessoes_do_usuario* mapeia um usuário *u* para o conjunto de suas respectivas sessões. Ou seja, fornece o conjunto de sessões correntes iniciadas por um usuário particular;
- A função *usuario_da_sessao* mapeia uma sessão *s* para o usuário que a iniciou. O invariante estabelece que o conjunto de sessões de um usuário somente contém as sessões por ele iniciadas. Acresça-se que uma sessão está associada a um único usuário;
- A função *papeis_da_sessao* mapeia uma sessão *s* para o conjunto de papéis ativados pelo usuário que iniciou a sessão. O invariante estabelece que o conjunto de papéis da sessão é um subconjunto do conjunto de papéis atribuídos para o usuário que iniciou a sessão;
- A função *autorizacoes_da_sessao* mapeia uma sessão *s* para o conjunto autorizações disponíveis para o usuário que iniciou a sessão. O invariante estabelece que o conjunto de autorizações de uma sessão é um subconjunto do conjunto de autorizações associadas aos papéis ativados na sessão por um usuário.

O CABP Básico evidencia a atuação dos papéis como intermediário entre usuários e autorizações. De fato, os papéis classificam os relacionamentos muitos-para-muitos existentes entre usuários e autorizações de acordo com as funções exercidas. São uma abstração para lidar com tais relacionamentos, o que facilita a administração de políticas de acesso em implementações de modelos do CABP Básico.

O invariante do esquema permite concluir que as autorizações disponíveis para um usuário são as autorizações associadas aos papéis ativos existentes em todas as sessões correntes iniciadas pelo usuário. O modelo proposto pelo NIST não prescreve um método específico para ativação de papéis durante uma sessão, entretanto, requer a capacidade de ativação simultânea de múltiplos papéis.

CABP Hierárquico

Em muitas organizações, uma considerável quantidade de autorizações é compartilhada pela maioria dos usuários a fim de que possam executar funções comuns a diferentes papéis. A utilização exclusiva do CABP Básico resultaria na atribuição das mesmas autorizações para cada um dos papéis, tornando a administração da política de acesso pouco eficiente e mais sujeita a erros. Por exemplo, a revogação de uma autorização implicaria em desassociá-la explicitamente de cada papel onde está vinculada, o que aumentaria o risco de se manterem privilégios residuais indevidos para usuários.

A disposição de papéis numa hierarquia, possibilitada pelo CABP Hierárquico, reduz essa desvantagem do CABP Básico porque classifica as relações muitos-para-muitos entre usuários e autorizações existentes em diferentes papéis. A classificação ocorre segundo uma relação de responsabilidade existente entre os papéis. Considere-se o papel *Diretor Clínico* ilustrado na hierarquia da *Figura 3* (a). Ele incorpora todas as autorizações dos papéis de *menor* responsabilidade em suas linhas hierárquicas: *Diretor*, *Administrador*, *Médico*, *Profissional de Saúde* e *Usuário*. Assim, sempre que uma autorização é vinculada a um desses papéis, ela, implicitamente, fica também vinculada ao papel *Diretor Clínico*. Do modo oposto, o papel *Usuário* incorpora todos os usuários associados aos papéis de *maior* responsabilidade em suas linhas hierárquicas: *Analista de Registro de Saúde*, *Auxiliar de Registro de Saúde*, *Administrador*, *Diretor Clínico*, *Diretor*, *Médico*, *Profissional de Saúde*, *Paramédico* e *Auxiliar de Enfermagem*. Assim, sempre que um usuário é associado a um desses papéis, ele, implicitamente, fica também vinculado ao papel *Usuário*. Note-se que, a hierarquia faz uma classificação de segunda ordem porque classifica as relações do

Figura 3 – Exemplos de hierarquias de papéis: (a) geral; (b) árvore; e (c) árvore invertida.

tipo usuário-autorização, já classificadas primariamente pelos papéis. Estruturar papéis em hierarquias também facilita a representação das linhas de autoridade e responsabilidade de uma organização.

O esquema a seguir especifica o CABP Hierárquico como uma extensão do CABP Básico:

<i>CABP_Hierarquico</i>
<i>CABP_Basico</i>
$hp: \text{papeis} \leftrightarrow \text{papeis}$
$\text{usuarios_associados_hp}: \text{papeis} \rightarrow \mathbb{P} \text{ usuarios}$
$\text{autorizacoes_associadas_hp}: \text{papeis} \rightarrow \mathbb{P} \text{ autorizacoes}$
$\forall p_1, p_2: hp \bullet p_1 \succeq p_2$
$\forall p: \text{dom usuarios_associados_hp} \bullet$ $\text{usuarios_associados_hp}(p) = \{u \in \text{usuarios} \mid \exists p_i: \text{papeis} \mid p_i \succeq p \in hp \bullet u \mapsto p_i \in up\}$
$\forall p: \text{dom autorizacoes_associadas_hp} \bullet$ $\text{autorizacoes_associadas_hp}(p) = \{a \in \text{autorizacoes} \mid \exists p_i: \text{papeis} \mid p_i \succeq p \in hp \bullet p_i \mapsto a \in pa\}$
$\forall p_1 \succeq p_2: hp \bullet \text{autorizacoes_associadas_hp}(p_2) \subseteq \text{autorizacoes_associadas_hp}(p_1) \wedge$ $\text{usuarios_associados_hp}(p_2) \subseteq \text{usuarios_associados_hp}(p_1)$

A hierarquia de papéis (*Figura 2*) é definida pela relação hp entre papéis. O invariante estabelece, com o operador relacional $\succeq$, que hp é uma ordem parcial entre papéis (v. definição no Anexo C, p. 179). Uma ordem parcial é uma relação reflexiva, transitiva e antissimétrica. As duas funções introduzidas trazem os usuários e as autorizações associadas aos papéis, levando-se em conta a hierarquia hp .

- A função $\text{usuarios_associados_hp}$ mapeia um papel p para o conjunto que inclui os usuários associados ao papel na relação up e também os usuários dos papéis com maior responsabilidade que a dele na hierarquia hp ;
- A função $\text{autorizacoes_associadas_hp}$ mapeia um papel p para o conjunto que inclui as autorizações associadas ao papel na relação pa e também as autorizações vinculadas aos papéis com menor responsabilidade que a dele na hierarquia hp .

Por fim, o invariante estabelece a propriedade que classifica as relações usuário-autorização. Para ilustrar, considere-se o relacionamento $\text{Diretor Clínico} \succeq \text{Usuário}$ da *Figura 3* (a): as autorizações associadas ao papel de menor responsabilidade (*Usuário*) são também autorizações do papel de maior responsabilidade (*Diretor Clínico*); e os usuários associados ao papel de maior responsabilidade (*Diretor Clínico*) são também usuários do papel de menor responsabilidade (*Usuário*).

Uma hierarquia geral, como a ilustrada na *Figura 3* (a), se caracteriza por permitir que um papel seja composto diretamente a partir de mais de um papel de menor responsabilidade, como no caso do papel *Diretor Clínico*, composto a partir de *Diretor* e *Médico*. Por outro lado, hierarquias gerais também permitem que um papel de menor responsabilidade seja usado diretamente na composição de mais de um papel de maior responsabilidade, como o caso do papel *Administrador*, utilizado na

composição *Auxiliar de Registro de Saúde* e *Diretor*. Hierarquias de papéis também podem ser vistas como uma relação de herança porque papéis de maior responsabilidade “herdam” as autorizações (o poder) dos papéis de menor responsabilidade. Assim, um papel de menor responsabilidade é chamado de “ancestral” do papel de maior responsabilidade, o qual é chamado de “descendente” daquele. Por exemplo, na *Figura 3 (a)*, *Médico* é descendente de *Usuário* e ancestral de *Diretor Clínico*.

A proposta do NIST admite hierarquias restritas, estruturadas em árvores ou árvores invertidas, como alternativa a utilização de hierarquias gerais. Numa estrutura em árvore (*Figura 3 (b)*), um papel pode ter mais de um ancestral direto, mas, no máximo, só pode ter um descendente imediato. Nas hierarquias estruturadas em árvores invertidas (*Figura 3 (c)*), os papéis podem ter vários descendentes diretos, mas no máximo um ancestral imediato. O esquema a seguir restringe o modelo de CABP Hierárquico a uma estrutura de árvore invertida:

<i>CABP_ArvoreInvertida</i>
<i>CABP_Hierarquico</i>
<i>ascendentes</i> : $papeis \rightarrow \mathbb{P} papeis$
<i>ascendentes_imediatos</i> : $papeis \rightarrow \mathbb{P} papeis$
$\forall p: \text{dom } ascendentes \bullet ascendentes(p) = \{p_i \in papeis \mid p \succeq p_i \in hp\}$
$\forall p: \text{dom } ascendentes_imediatos \bullet$ $ascendentes_imediatos(p) = \{p_i \in ascendentes(p) \mid$ $\exists p_2: papeis \mid p_i \neq p_2 \wedge p_2 \neq p \bullet p \succeq p_2 \in hp \wedge p_2 \succeq p_i \in hp\}$
$\forall p, p_r, p_2: papeis \bullet$ $p_i \in ascendentes_imediatos(p) \wedge p_2 \in ascendentes_imediatos(p) \Rightarrow p_i = p_2$

A função *ascendentes* retorna todos os ancestrais de um dado papel, enquanto a função *ascendentes_imediatos* retorna apenas os ancestrais diretos. O invariante estabelece que um papel p_i é ascendente direto de um outro papel p caso não exista um papel intermediário e distinto deles na hierarquia que seja ancestral de p e descendente de p_i . Por fim, o invariante estabelece a condição restritiva que estrutura a hierarquia geral numa árvore invertida: um papel qualquer não pode ter mais de um ascendente imediato. A especificação da hierarquia de papéis em estrutura de árvore pode ser encontrada na proposta do NIST para CABP (Ferraiolo *et al.*, 2001).

CABP Restrito

Restrições são um poderoso mecanismo para delinear a política de acesso numa organização (Sandhu *et al.*, 1996). O CABP Restrito introduz restrições por meio de relações de separação de responsabilidades (SR), que particionam de modo compulsó-

rio a responsabilidade e a autoridade para realizar uma ação por vários papéis mutuamente exclusivos. A SR é um princípio de segurança aceito para impor políticas de redução de conflitos de interesses por sua larga aplicação em organizações comerciais, industriais e de governo (Brewer & Nash, 1989; Clark & Wilson, 1987). Seu propósito é assegurar que eventuais fraudes ou danos acidentais somente ocorram como resultado de conluíus, mas não decorrentes da demasiada concentração de poder numa única pessoa. A idéia é adotar a SR para minimizar a possibilidade de um usuário realizar ações em que há conflito de interesses. Por exemplo, pode-se configurar uma política de SR para proibir que um mesmo indivíduo atue como médico assistente e médico auditor, de modo a evitar que ele possa auditar os próprios atos.

O conflito de interesses ocorre no CABP quando um usuário detém autorizações para executar ações associadas a papéis conflitantes. A separação de responsabilidades limita as autorizações disponíveis aos usuários e pode ser estática ou dinâmica.

Separação de Responsabilidade Estática

A separação de responsabilidade estática impõe restrições na atribuição de papéis para usuários, conforme especificado no esquema a seguir:

<i>CABP_SeparacaodeResponsabilidadeEstatica</i>
<i>CABP_Hierarquico</i>
$sre : \mathbb{P} \text{ papéis} \leftrightarrow \{n \in \mathbb{N} \mid n \geq 2\}$
$\forall (pc, n) : sre \bullet$ $\quad \forall t : \mathbb{P} \text{ papéis} \mid t \subseteq pc \bullet$ $\quad \#t \geq n \Rightarrow \bigcap \{usuarios_associados_hp(p) \mid p \in t\} = \emptyset$

A especificação acima estabelece a SRE para o modelo de CABP Hierárquico. A proposta do NIST também admite a configuração da SRE com o modelo de CABP Básico ou então com o CABP Hierárquico com estrutura de árvore ou árvore invertida. A SRE é definida pela relação *sre*, que compreende uma coleção de pares formados por um conjunto de papéis conflitantes *pc* e um número natural *n* maior que um. O invariante estabelece que nenhum usuário pode associar-se a *n* ou mais papéis do conjunto *pc* dos pares da relação *sre* (v. definição do operador de intersecção generalizada de conjuntos $\cap$ na p. 179 no Anexo C). Para ilustrar, suponha o par $\langle \{Médico Assistente, Médico Auditor\}, 2 \rangle$. De acordo com o invariante, um usuário associado ao papel *Médico Assistente* está proibido de vincular-se ao papel *Médico Auditor* e vice-versa. Ou seja, ambos os papéis estão proibidos de possuir usuários em comum. No-

te-se que, a utilização no invariante da função *usuarios_associados_hp* considera a hierarquia, assegurando a SRE para os membros dos eventuais papéis descendentes de *Médico Assistente e Médico Auditor*. A SRE ainda permite a configuração de combinações de papéis conflitantes que um usuário pode associar-se mediante valores de n superiores a dois. Por exemplo, uma organização pode estabelecer que um usuário não pode associar-se a mais do que três de um total de cinco papéis que executam funções mutuamente conflitantes, definido um valor de n igual a quatro.

Uma consequência da SRE envolvendo pelo menos dois papéis com conjuntos de autorizações distintas é a impossibilidade da existência de superusuários. Supondo que um superusuário é aquele que tem atribuído para si todos os papéis disponíveis, então ele deveria ser vinculado também aos dois papéis conflitantes na relação *sre*, mas, de acordo com o invariante do esquema *CABP_SeparacaodeResponsabilidadeEstatica*, tais papéis não podem ter usuários em comum, logo o superusuário não pode existir. A outra possibilidade é o superusuário estar associado a um papel raiz com os poderes de todos os outros papéis. O papel raiz é aquele que herda, direta ou indiretamente, as autorizações dos demais papéis. Ou seja, ele é descendente também dos dois papéis conflitantes na relação *sre* e, de acordo com o invariante do esquema *CABP_SeparacaodeResponsabilidadeEstatica*, tal papel somente pode existir se o conjunto de usuários associados a ele for vazio. Logo, não pode haver um superusuário com os poderes deste papel.

A separação de responsabilidade estática limita as autorizações disponíveis para um usuário pela definição de restrições nas relações *up* e *hp* (Figura 2). Gligor *et al.* (1998) mostram que a SRE também pode ser definida impondo-se restrições na relação *pa*, de modo que dois ou mais papéis (conflitantes) fiquem proibidos de compartilhar autorizações que provoquem conflitos de interesses potenciais. Restrições estáticas adicionais incluem a limitação do número máximo de papéis de um usuário ou da quantidade máxima de usuários associados a um papel.

Separação de Responsabilidade Dinâmica

Há situações em que os conflitos de interesses potenciais ocorrem somente quando um usuário ativa simultaneamente dois ou mais papéis específicos. Os conflitos não emergem quando um mesmo usuário os ativa alternadamente. A separação de responsabilidade dinâmica, portanto, limita as autorizações disponíveis para um usuário impondo restrições nos papéis que ele pode ativar simultaneamente numa mesma sessão, conforme especificado no esquema a seguir:

CABP_SeparacaodeResponsabilidadeDinâmica

RBAC_Hierarquico
 $srd: \mathbb{P} \text{ papeis} \leftrightarrow \{n \in \mathbb{N} \mid n \geq 2\}$

 $\forall (pc, n) : srd \bullet \#pc \geq n$
 $\forall s: \text{sessoes}, (pc, n) : srd, scp: \mathbb{P} \text{ papeis} \bullet$
 $scp \subseteq pc \wedge scp \subseteq \text{papeis_da_sessao}(s) \Rightarrow \#scp < n$

A SRD é definida pela relação srd , que compreende uma coleção de pares formados por um conjunto de papéis conflitantes pc e um número natural n maior que um. O invariante estabelece que usuários são proibidos de ativar, simultaneamente numa sessão, n ou mais papéis do conjunto pc dos pares da relação srd . A SRD pode ser estendida para proibir a ativação simultânea de papéis conflitantes em todas as sessões correntes do usuário, e não apenas no escopo de uma sessão. Gligor *et al.* (1998) mostram outros tipos de SRD, como aquela em que a autorização para um usuário executar uma operação vai depender do histórico de operações que ele já executou.

A SRD fortalece o princípio do menor privilégio porque restringe o espaço de autorizações efetivamente disponíveis para o usuário num dado instante de tempo. A ativação de um papel pelo usuário implica na desativação dos papéis com relação de SRD, de modo que a disponibilidade de autorizações sensíveis não persiste além do tempo necessário para a tarefa pretendida. Note-se que, a separação estática de responsabilidades também é capaz de restringir os papéis que um usuário pode ativar simultaneamente, mas de forma menos flexível.

Cabe ressaltar que a separação de responsabilidades, estática ou dinâmica, é inerente à implementação de políticas de acesso orientadas à aplicação, que deve ser projetada considerando-se a partição das tarefas que levem a conflitos de interesses ou erros (Gligor *et al.*, 1998). Tipicamente, a SR requer a inclusão de funções independentes e redundantes nas aplicações. Os resultados das execuções das funções devem coincidir para efetivar uma tarefa. Por exemplo, duas pessoas distintas devem aprovar a execução de uma tarefa crítica, como alterar o salário de um funcionário. Outra modalidade de SR exige, nas aplicações, o particionamento das tarefas operacionais e de verificação, de modo que, a classe dos usuários que emitem ordens médicas seja diferente da classe dos usuários que as auditam, por exemplo.

2.4.1.2 A especificação funcional

O modelo de referência anterior descreve o espaço do estado dos componentes do CABP, isto é, os possíveis estados abstratos de implementações do modelo. A especificação funcional define operações que atuam sobre tais estados para administração e revisão dos componentes do modelo e também para gerenciamento de sessões. Es-

pecifica rigorosamente a semântica das operações para preservar as propriedades do modelo estabelecidas nos invariantes dos esquemas de estado. Por exemplo, a operação que associa um usuário a um papel, na presença da separação de responsabilidade estática, deve assegurar que:

1. O usuário é membro do conjunto de usuários cadastrados;
2. O papel é membro do conjunto de papéis cadastrados;
3. O usuário não está associado ao papel;
4. Após a associação, os papéis a que o usuário está vinculado não violam as restrições estabelecidas pela relação de SRE.

A especificação formal completa dessas operações em esquemas Z são encontradas no apêndice A do documento que propõe o padrão para CABP do NIST (Ferraiolo *et al.*, 2001). As operações especificadas para cada um dos componentes do modelo de referência estão agrupadas nas três categorias a seguir:

- *Funções administrativas* para criação e manutenção dos conjuntos de elementos e respectivas relações, usadas na construção dos vários componentes do modelo do CABP;
- *Funções de suporte de sistema* necessárias às implementações do CABP para suportar a funcionalidade do modelo durante a interação do usuário com um sistema de informação. Por exemplo, as funções de gerência de sessões e de autorização de acesso aos objetos protegidos;
- *Funções de revisão* para verificar o efeito das ações realizadas pelas funções administrativas. Por exemplo, funções que fornecem as autorizações disponíveis para um usuário, ou os usuários autorizados a executar uma operação sobre um objeto.

O esquema Z, apresentado a seguir, ilustra a especificação de uma das funções de suporte de sistema. No caso, o algoritmo da função que decide se um usuário está autorizado a executar uma operação sobre um objeto por meio de sua sessão.

<i>AcessoAutorizado</i>
$\exists CABP_Basico$
<i>sessad</i> , <i>operacad</i> , <i>objeto</i> : NOME
<i>resultado?</i> : BOOLEAN
$sessad \in sessoes \wedge operacad \in ops \wedge objeto \in obs$
$resultado? = (\exists p : papeis \bullet$ $p \in papeis_da_sessao(sessad) \wedge p \mapsto (operacad, objeto) \in pa)$

O esquema *AcessoAutorizado* é um tipo de esquema Z que especifica o efeito da execução de uma operação sobre um estado particular do sistema, sendo denominado

de *esquema de operação*. A primeira linha da parte declarativa do esquema indica que a operação *AcessoAutorizado* atua sobre o estado do modelo básico do CABP e o operador Ξ estabelece que o estado global do sistema não é alterado pela execução da operação. Ou seja, a operação usa o estado apenas para consulta. As variáveis declaradas com o sufixo ! são parâmetros de entrada, enquanto as declaradas com o sufixo ? são parâmetros de saída da operação.

A parte inferior do esquema estabelece a pré e a pós condição para execução da operação. Os parâmetros de entrada de *AcessoAutorizado* devem satisfazer a seguinte pré-condição, imediatamente antes da execução da operação: os identificadores da sessão, da operação e do objeto devem pertencer aos conjuntos de sessões ativas, de operações e de objetos cadastrados, respectivamente. Logo após a execução da operação, os parâmetros de saída devem satisfazer a seguinte pós-condição: o resultado da operação é verdadeiro (acesso autorizado) se, e somente se, existir um papel ativo para a sessão iniciada pelo usuário que tenha uma autorização associada a operação e ao objeto fornecidos como parâmetros de entrada. A especificação da operação *AcessoAutorizado* permanece a mesma quando se atua sobre os modelos hierárquicos ou restritos do CABP. Observa-se ainda que o esquema especifica as condições que devem ser satisfeitas antes e após a execução do algoritmo de decisão de acesso, mas não prescreve algoritmos particulares.

∴

O modelo de referência proposto pelo NIST define um padrão semântico para o CABP, orientando extensões e implementações a não violar suas características essenciais. O padrão prescreve que extensões e implementações do controle de acesso baseado em papéis satisfaçam minimamente o CABP Básico. Modelos e implementações complementares podem ser concebidos combinando-se os modelos CABP Hierárquico ou CABP Restrito. Por ser o CABP uma tecnologia aberta, ainda em evolução, a proposta do NIST permite que extensões do modelo básico incluam características mais avançadas, voltadas à solução de problemas específicos de controle de acesso não atendidos satisfatoriamente pelo padrão.

2.5 Trabalhos relacionados

Vários modelos para controle de acesso baseado em papéis foram propostos recentemente na literatura (Bacon *et al.*, 2002; Bertino *et al.*, 2001; Covington *et al.*, 2002; Covington *et al.*, 2001; Georgiadis *et al.*, 2001; Joshi *et al.*, 2003; Kumar *et al.*, 2002;

Lin & Brown, 2000; Longstaff *et al.*, 2003; Neumann & Strembeck, 2003; Wilikens *et al.*, 2002). Esta seção revisa sucintamente os modelos levantados mais relevantes para aplicação ao problema do controle de acesso ao prontuário eletrônico do paciente, com a identificação das seguintes características:

- *CABP Básico*: indica se o modelo revisado suporta o modelo básico do CABP do NIST. Como todos os modelos considerados suportam o CABP Básico, essa característica não é explicitamente apontada;
- *CABP Hierárquico*: indica se o modelo revisado suporta a hierarquia de papéis. Em caso afirmativo, informa qual o tipo: geral, árvore ou árvore invertida;
- *CABP Restrito*: indica se o modelo revisado suporta restrições, particularmente aquelas da separação de responsabilidade dos tipos estática e dinâmica;
- *Autorizações*: indica se o modelo revisado suporta apenas autorizações positivas, apenas autorizações negativas, ou ambas as modalidades;
- *Regras contextuais*: indica se o modelo revisado suporta regras contextuais e onde elas são aplicadas. Uma regra contextual é uma expressão lógica definida em termos de informações ambientais (contextos) e sua avaliação pode desencadear a execução de uma ação. Regras podem ser aplicadas para: compor a lógica de decisão numa autorização; definir as condições para ativação/desativação de papéis; estabelecer restrições arbitrárias às entidades e relacionamentos que compõe o modelo do CABP;
- *Implementação*: indica se o modelo revisado foi implementado e integrado a uma plataforma. Em caso afirmativo, informa a plataforma e se a implementação é apenas um protótipo ou se é um produto acabado com relatos de uso por organizações em ambientes de produção;
- *Aplicação ao PEP*: indica se o modelo revisado foi aplicado para resolver problemas de controle de acesso ao PEP, seja apenas na forma de exemplos, seja no uso rotineiro em organizações de saúde;

Essas características são usadas para relacionar, na discussão do capítulo 5, o MACA com os modelos revisados e sua aplicação ao PEP. As subseções seguintes apresentam os modelos, com destaque para as características identificadas, cujo sumário é apresentado na *Tabela 1*.

Tabela 1 – Sumário das características identificadas nos modelos para CABP revisados.

Modelos	Características					
	CABP Hierárquico	CABP Restrito	Autorizações	Regras contextuais	Implementação	Aplicação ao PEP
<i>E-RBAC</i> (Covington <i>et al.</i> , 2003)	Geral.	SRD para papéis ambientais.	Positivas e negativas.	Usadas para ativação/desativação de papéis ambientais.	Protótipo desenvolvido na plataforma Java 2 SE.	Não relatada.
<i>C-TMAC</i> (Georgiadis <i>et al.</i> , 2001)	Não.	SRE para papéis e equipes.	Positivas.	Vinculadas a equipes e definidas como cláusulas <i>where</i> de da linguagem SQL.	Protótipo desenvolvido na plataforma Oracle e implementado em PL/SQL e SQL dinâmico.	Sim, em caráter experimental numa intranet.
<i>CS-RBAC</i> (Kumar <i>et al.</i> , 2002)	Não.	Não.	Positivas.	Definidas em papéis como filtros que relacionam os contextos de usuário e de objeto, que são fixos.	Protótipo desenvolvido nas tecnologias JSP e servlets e disponibilizado na plataforma Tomcat.	Não relatada.
<i>TCM</i> (Longstaff <i>et al.</i> , 2003)	Geral, para papéis e identidades, usando o conceito de coleções.	Não.	Positivas e negativas.	Não.	Protótipo desenvolvido com base em SQL e XML na plataforma .NET/C#.	Sim, num estudo de caso contemplando requisitos de confidencialidade para informações clínicas de pacientes na Grã-Bretanha.
<i>DRIVE-RBAC</i> (Wilikens <i>et al.</i> , 2002)	Geral, para papéis e usuários.	SRE e SRD.	Positivas.	Usadas para ativação/desativação de papéis em sessões de um usuário específico, de acordo com o estado de um conjunto fixo de variáveis contextuais.	Não relatada.	Sim, num estudo de caso para um sistema de suprimento de drogas integrado do PEP.
<i>RBPEAC</i> (Lin <i>et al.</i> , 2000)	Não.	Não.	Positivas e negativas.	Definidas nas autorizações por meio de expressões em Prolog.	Disponibilidade bibliotecas para as plataformas Windows NT 4.0 e HP-UX 11.0.	Não relatada.
<i>XORBAC</i> (Neumann <i>et al.</i> , 2003)	Geral.	SRE com papéis e autorizações mutuamente exclusivas. Limites máximos e mínimos para as quantidades de usuários e autorizações permitidas em cada papel.	Positivas.	Definidas como um conjunto de condições (restrição contextual) que determinam a validade de uma autorização.	Disponibilidade de um produto acabado, implementado em XOTcl, para as plataformas <i>Unix-like</i> e Windows, para uso sem fins comerciais.	Não relatada.
<i>OASIS</i> (Bacon <i>et al.</i> , 2002)	Não.	SRD por meio de regras definindo as condições de ativação de papéis.	Positivas.	Definidas como regras (cláusulas de Horn) que determinam as condições para vinculação dinâmica de uma autorização a um papel na relação papel-autorização. Também são usadas para estabelecer as condições para ativação/desativação dinâmica de papéis.	Protótipo desenvolvido em Java com <i>web services</i> usando o SGDB PostgreSQL como suporte a regras ativas e para armazenamento de dados persistentes.	Sim, num estudo de caso contemplando requisitos de confidencialidade para informações clínicas de pacientes na Grã-Bretanha.
<i>GTRBAC</i> (Joshi <i>et al.</i> , 2003)	Geral.	SRE e SRD mediante restrições temporais nas relações do modelo.	Positivas.	Usadas de forma generalizada para impor restrições (temporais) nas entidades e relações que compõem o modelo.	Protótipo com funcionalidade reduzida desenvolvido na plataforma Oracle com implementação em PL/SQL.	Sim, nos exemplos apresentados nos trabalhos que descrevem o modelo.

2.5.1 Modelo E-RBAC

O modelo E-RBAC (*Environment Role-Based Access Control*) foi desenvolvido por Covington *et al.* (2002); Covington *et al.* (2001) no Instituto de Tecnologia da Georgia, nos EUA, para aplicação numa arquitetura de sistema para “residências inteligentes”. O E-RBAC estende o CABP Básico com a introdução de um novo tipo de papel, denominado *papel ambiental*. Papéis ambientais compartilham importantes propriedades com os papéis tradicionais do CABP, como a capacidade de ser ativado, hierarquias e a separação de responsabilidades. Entretanto, diferentemente desses, não têm usuários associados, mas autorizações e uma regra estabelecendo as condições para ativação/desativação automática do papel com base num conjunto de informações ambientais (*e. g.*, horário, temperatura, nível de ruído, localização do usuário) capturadas por sensores distribuídos na residência. Por exemplo, *Sala de Estar Ocupada* e *Horário de Lazer* poderiam ser papéis ambientais que estariam ativos sempre que a sala de estar estivesse ocupada ou quando o horário corrente fosse de lazer, respectivamente. No E-RBAC, uma autorização tem associada um papel tradicional, uma operação, um objeto e um conjunto de papéis ambientais. O acesso para executar uma operação de um objeto é concedido a um usuário quando ele é membro do papel tradicional e todos os papéis, do conjunto de papéis ambientais, estão ativos no momento da solicitação de autorização. Para ilustrar, a autorização positiva $\langle \text{Criança, Assistir, Televisão, \{Sala de Estar Ocupada, Horário de Lazer\}, +} \rangle$ estabelece que indivíduos com o papel tradicional de criança somente podem assistir televisão quando a sala de estar estiver ocupada e no horário de lazer.

O E-RBAC suporta hierarquias gerais para papéis tradicionais e ambientais. Os autores relatam a utilização da separação de responsabilidade dinâmica para evitar que dois papéis ambientais conflitantes consigam se ativar simultaneamente. Ou seja, se o papel *Horário de Lazer* está ativo, o papel *Horário de Estudo* deve estar desativado e vice-versa. As autorizações podem ser positivas ou negativas, embora os autores não relatem qual a política para resolução de conflitos nos casos em que, para um mesmo indivíduo, uma autorização concede o acesso e outra o proíbe. As regras contextuais são suportadas pelo modelo na definição das condições de ativação/desativação dos papéis ambientais. Um protótipo do modelo foi implementado na plataforma Java 2 SE (Sun Microsystems, 2003a) e não foi relatado a sua aplicação no controle de acesso ao prontuário eletrônico do paciente.

2.5.2 Modelo C-TMAC

O modelo C-TMAC (*Contextual Team-Based Access Control*) foi desenvolvido por Georgiadis *et al.* (2001) na Universidade de Tessalônica, na Grécia, para imposição de políticas de acesso em ambientes colaborativos. O C-TMAC estende o CABP Básico com a introdução de *equipes*, *contextos* e dos relacionamentos *usuário-equipe* e *equipe-contexto*. Usuários são membros de equipes que têm contextos associados, ou seja, o conceito de equipe é usado para relacionar usuários com contextos, do mesmo modo que papéis relacionam usuários e autorizações. Um contexto contém uma regra lógica relacionando os objetos usados na execução de uma tarefa com variáveis ambientais, como localização do usuário e horário do acesso. Durante uma sessão, um usuário ativa papéis e equipes e as autorizações disponíveis para ele correspondem à combinação das autorizações associadas a todos os papéis correntes ativados pelos membros das equipes ativadas, filtradas pelas regras contextuais vinculadas às equipes. Por exemplo, suponha-se os papéis *Enfermeiro* e *Médico* e a equipe *Emergência* com uma regra contextual estabelecendo que os acessos somente são permitidos das salas de emergência. Quando um enfermeiro inicia uma sessão, ele ativa o papel *Enfermeiro* e a equipe *Emergência* e tem as autorizações disponíveis para o seu papel, desde que efetue acessos de alguma sala de emergência. Quando um médico inicia uma sessão e ativa o papel *Médico* e a equipe *Emergência*, então, tanto ele, quanto o enfermeiro, vão compartilhar as autorizações de ambos os papéis porque são membros da mesma equipe, desde que os acessos sejam realizados de salas de emergência. Quando o médico encerra a sessão, desativando seu papel, o enfermeiro deixa de ter as autorizações do papel *Médico*. Ou seja, os membros de uma equipe compartilham as autorizações enquanto realizam uma tarefa em comum.

O C-TMAC foi especificado sem suporte a hierarquias de papéis. Os autores relatam a utilização de SRE para evitar usuários em comum entre papéis e entre equipes específicas. Regras contextuais são suportadas pelo C-TMAC associadas a equipes e usadas como critério de filtragem de autorizações disponíveis para os papéis ativos dos membros da equipe. Um protótipo do modelo (Georgiadis *et al.*, 2002) foi desenvolvido na plataforma Oracle (Fernandes, 2002a), com implementação em PL/SQL (Fernandes, 2002b) e SQL (Celko, 1999) dinâmico, para aplicação em intranets na área de saúde. Autorizações são definidas como visões ou consultas a bancos de dados relacionais e as regras contextuais são cláusulas *where* da linguagem SQL, aplicadas às consultas das autorizações para fins de filtragem. Variáveis contextuais são substituídas nas consultas dinâmicas em tempo de execução.

2.5.3 Modelo CS-RBAC

O modelo CS-RBAC (*Context-Sensitive Role-Based Access Control*) foi desenvolvido por Kumar *et al.* (2002) no Instituto de Tecnologia da Índia em cooperação com o Laboratório de Pesquisa da IBM na Índia. O CS-RBAC estende o CABP básico pela introdução dos conceitos de *contextos de papéis* e de *filtros contextuais*. O contexto de um papel é formado pela combinação do contexto do usuário que iniciou uma sessão com o contexto do objeto associado à autorização de acesso solicitada. Contextos são formados por conjuntos de pares de nomes e valores e denotam informações relevantes para segurança, sendo dependentes de aplicações específicas. Por exemplo, numa aplicação comercial, o contexto do usuário pode incluir o nome da empresa (cliente) para qual o usuário trabalha, enquanto o contexto de um objeto do tipo pedido pode conter a identificação da empresa criadora do pedido. Filtros contextuais são definidos em papéis e estabelecem uma regra lógica relacionando os contextos de usuário e de objeto. Autorizações de acesso a um objeto só são permitidas quando a avaliação da regra do filtro contextual é verdadeira. Assim, a concessão ou a proibição de uma autorização dependerá do contexto do papel existente durante a solicitação da autorização de acesso. Para ilustrar, considere-se a autorização $\langle \textit{Cliente}, \textit{alterar}, \textit{Pedido} \rangle$ estabelecendo que um usuário com papel *Cliente* pode alterar um objeto do tipo pedido. Sem a definição de um filtro contextual, não há, no modelo, como impedir que um usuário altere pedidos de uma empresa diferente daquela em que ele trabalha. No caso, a inclusão do filtro contextual `ObjectContext.empresaID = UserContext.empresaID` no papel *Cliente* define a seguinte política: um usuário somente pode alterar os pedidos da empresa para a qual ele trabalha.

O CS-RBAC foi especificado sem hierarquias de papéis e sem restrições de separação de responsabilidades. Suporta autorizações positivas e regras contextuais, que são definidas nos papéis e atuam como filtros de autorizações. Cabe ressaltar que existem apenas dois contextos (de usuário e de objeto) e que eles estão fixos na linguagem usada para expressar os filtros contextuais. O que muda de acordo com a aplicação são os conjuntos de pares que formam cada um dos contextos, ou seja, as variáveis contextuais. Um protótipo do modelo foi desenvolvido nas tecnologias JSP (Sun Microsystems, 2003b) e *servlets* (Sun Microsystems, 2003c) e disponibilizado na plataforma *Tomcat* (Apache Software Foundation, 2003) para utilização num estudo de caso de *web services* (Newcomer, 2002). Os autores não relataram a aplicação do modelo para controlar o acesso ao prontuário eletrônico do paciente.

2.5.4 Modelo TCM

O modelo TCM (*Tees Confidentiality Model*) foi desenvolvido por Longstaff *et al.* (2003) na Universidade de Teesside, na Inglaterra, e é candidato a modelo de autorização para implementar a política nacional de confidencialidade para o PEP proposta pelo Serviço Nacional de Saúde da Grã-Bretanha (National Health Service, 2002). O TCM estende o CABP Básico com a introdução dos conceitos de *coleções*, de *identidades*, e de cinco modalidades de autorizações, que podem ser positivas ou negativas. Coleções possuem elementos que podem ser seus membros ou outras coleções. A capacidade de conter outras coleções (subcoleções) as tornam inerentemente hierárquicas. Uma identidade pode ser membro de coleções de identidades e representa um usuário de um sistema de informação. Hierarquias de coleções de identidades são usadas para representar as estruturas de equipes/subequipes que ocorrem nos organogramas de organizações de saúde, dentre outras estruturas. Também estruturam as hierarquias de papéis e agrupam seus membros, que podem ser identidades ou coleções de identidades. Dois aspectos caracterizam as modalidades de autorizações no TCM: 1) o tipo de entidade ao qual está associada (identidade, coleções de identidades, papéis, coleções de papéis); e 2) o tipo de objeto protegido, que pode ser uma coleção de itens de dados clínicos de um paciente particular ou uma coleção de tipos de dados, que representa um esquema de codificação de informações clínicas, por exemplo. Isso permite que uma autorização associe-se diretamente a uma identidade (ou coleções de identidades), sem a intermediação de papéis. As diferentes modalidades de autorização são avaliadas segundo uma ordem de precedência, de modo que, aquelas relacionadas a identidades prevaleçam sobre as autorizações relacionadas a papéis. Ou seja, autorizações concedidas a identidades sobrepõem autorizações concedidas a papéis. De forma similar, autorizações associadas a objetos que são coleções de itens de dados —mais específicos— prevalecem sobre as autorizações associadas a coleções de tipos de dados, mais genéricos. Por exemplo, o médico responsável por um paciente pode ter, mediante uma autorização direta para sua identidade, acesso a resultados de exames de HIV do paciente, mesmo que os eventuais papéis atribuídos a ele proibam explicitamente o acesso, em geral, a exames do tipo HIV.

O TCM suporta hierarquias gerais para papéis, identidades e objetos por meio de coleções. Uma autorização associada a um papel ou identidade numa hierarquia pode ser herdada por ascendentes ou (exclusivo) descendentes, de acordo com uma diretiva de herança presente na autorização. Ou seja, o mecanismo de he-

rança de autorizações é bidirecional. O modelo não suporta restrições, nem regras contextuais. Uma implementação piloto foi desenvolvida com base em SQL e XML (W3 Consortium, 2000), na plataforma .NET/C# (Lima & Reis, 2002), para demonstrar a aplicação do modelo num estudo de caso contemplando requisitos da política de confidencialidade proposta para informações clínicas de pacientes na Grã-Bretanha.

2.5.5 Modelo DRIVE-RBAC

O modelo DRIVE-RBAC foi desenvolvido por Wilikens *et al.* (2002) no Instituto para Proteção e Segurança do Cidadão, na Itália, para utilização no projeto DRIVE (*Drug in Virtual Enterprise*)⁷. O projeto visa o desenvolvimento de uma infra-estrutura de informação confiável para otimizar os processos clínicos e da cadeia de suprimento de drogas nas organizações de saúde. O DRIVE-RBAC estende o CABP Básico com a modificação do conceito de usuário, a introdução de propriedades dinâmicas baseadas em contextos e em eventos para ativação/desativação de papéis e a estruturação dos objetos protegidos em três níveis de acesso. Um usuário não denota apenas um único indivíduo, mas uma categoria de indivíduos agrupados segundo critérios específicos. Por exemplo, as categorias dos usuários clínicos, dos usuários de apoio clínico ou dos usuários de administração e logística. Categorias de usuários podem ser estruturadas em hierarquias. As propriedades dinâmicas do DRIVE-RBAC definem as condições para ativação/desativação de um papel para um usuário, numa sessão, por meio de *eventos contextuais*. Um evento contextual é definido pela 5-tupla $\langle I, P, L, C, p.E \rangle$, onde I é um intervalo de tempo; P é uma expressão periódica de tempo; L designa um local; C indica se a circunstância é de rotina ou de emergência; e p é a prioridade associada ao evento E , que permite ativação/desativação de um papel para um usuário numa sessão. Por exemplo, a tupla $\langle [-4, 4], \text{diurno ou noturno}, \text{enfermaria}, \text{rotina}, \text{normal}, \text{ativar}(\text{UsuárioClínico}, \text{Enfermeiro}) \rangle$ estabelece que um usuário da categoria *UsuárioClínico*, com o papel de *Enfermeiro*, somente pode ativar esse papel e adquirir as respectivas autorizações se estiver na enfermaria, numa circunstância de rotina, de dia ou de noite, em qualquer horário, numa prioridade normal.

O DRIVE-RBAC suporta hierarquias gerais para papéis e usuários, autorizações positivas e os autores apenas relatam a existência de separação de responsabilidades estática e dinâmica, mas não mostram como está integrada ao modelo. Regras

⁷ O DRIVE é um projeto de P&D financiado pela União Europeia no âmbito do programa EC IST, sob número IST-1999-12040, com informações disponíveis em <http://www.e-mathesis.it/drive/>.

contextuais são usadas para ativação/desativação de papéis em sessões de um usuário específico, de acordo com o estado de um conjunto fixo de variáveis disponíveis nos contextos de tempo (T e P); de local de acesso do usuário (L); e da circunstância no momento do acesso (C). Os autores não relatam a existência de uma implementação do modelo, apenas descrevem sua integração a uma arquitetura e apresentam um estudo de caso para um sistema de suprimento de drogas integrado ao PEP.

2.5.6 Modelo RBPEAC

O modelo RBPEAC (*Role-Based Policy-Enforced Access Control*) foi desenvolvido por Lin *et al.* (2000) nos laboratórios da Hewlett-Packard na Grã-Bretanha. O RBPEAC estende o CABP Básico com a introdução de *regras lógicas* às autorizações e pela definição de políticas para *atribuição dinâmica* de usuários a papéis. As regras, expressas na linguagem de programação Prolog (Clocksin & Mellish, 1984), são avaliadas durante uma solicitação de autorização de acesso: se o resultado da avaliação é verdadeiro, a autorização é positiva, caso contrário, ela é negativa. Quando um usuário inicia uma sessão e tenta acessar um objeto, o mecanismo de decisão verifica se há algum papel com uma autorização positiva (resultado da avaliação da regra) para o objeto. Caso exista, o acesso é concedido, caso contrário, ele é negado. As regras em Prolog permitem expressar políticas de acesso *ad hoc* com base em relações entre variáveis arbitrárias. Por exemplo, podem estabelecer que usuários com papel *Cliente*, num banco, somente podem transferir (operação) um valor de sua conta (objeto) para outra se o valor for inferior a \$1.000,00 e inferior ao saldo da conta de débito. A atribuição dinâmica permite que papéis sejam vinculados a um usuário após a sua autenticação usando um certificado digital no padrão X.509 (ITU-T, 1993). No caso, uma política específica os critérios de atribuição de acordo com os atributos (*e.g.*, profissão, endereço) do certificado do usuário. Essa característica coexiste com a atribuição estática encontrada no CABP Básico.

O RBPEAC não suporta hierarquia de papéis e separação de responsabilidades. As autorizações podem ser positivas ou negativas e, havendo conflitos na solicitação de acesso de um usuário, prevalece a autorização que concede o acesso. O modelo suporta regras nas autorizações por meio de expressões em Prolog, com os contextos definidos nas variáveis e nas relações disponíveis na linguagem. Os autores detalham a integração do modelo numa arquitetura de segurança e mencionam a disponibilidade de uma biblioteca para as plataformas Windows NT 4.0 e HP-UX 11.0. Não há relatos da aplicação do modelo no PEP.

2.5.7 Modelo XORBAC

O modelo XORBAC (*eXtended Object Role-Based Access Control*) foi desenvolvido por Neumann & Strembeck (2003); Neumann & Strembeck (2001) na Universidade Viena de Economia e Administração de Negócios, na Áustria, para utilização em aplicações *web*, particularmente em sistemas de código móvel. O XORBAC estende o CABP Básico com a introdução de *restrições contextuais* para determinar as condições de não violação de uma autorização. Uma restrição contextual é formada por uma conjunção lógica de *condições contextuais*, que definem uma relação booleana entre valores de *atributos contextuais*. Os atributos contextuais são obtidos com a chamada de funções disponíveis em sensores (lógicos ou físicos) e representam propriedades do ambiente que mudam dinamicamente (*e. g.*, hora, data) ou que variam de acordo com a instância de uma mesma entidade abstrata (*e. g.*, um local, o nome e a data de aniversário do usuário que solicita o acesso). Uma ou mais restrições contextuais são associadas a *autorizações condicionais*, que concedem o acesso se, e somente se, cada uma das restrições avaliarem para verdadeiro. Ou seja, caso alguma condição contextual seja violada no momento da tentativa de acesso, a autorização é negada. Por exemplo, a política de acesso de um sistema de testes, de um portal educacional na Internet, poderia estabelecer que um estudante somente pode realizar testes elaborados especificamente para ele, na data marcada, no horário comercial, a partir de estações num domínio de endereço IP preestabelecido. A vinculação da restrição contextual $\{cond; matricula_estudante(usuário) == matricula_exame(teste), cond; data_atual() == data_exame(teste), cond; hora_atual() in horário_comercial(), cond; IP_cliente() == *.universidade.edu.br\}$ à autorização $\langle Realizar, Teste \rangle$ submete a realização de testes pelos usuários a não violação de todas as condições da restrição, ou seja, torna a autorização condicional. A associação da autorização ao papel *Estudante* conclui a configuração da política enunciada anteriormente.

O XORBAC suporta a hierarquia geral de papéis com herança de autorizações e a separação estática de responsabilidades pela definição de papéis mutuamente exclusivos ou de autorizações mutuamente exclusivas. Dois ou mais papéis em exclusão mútua não admitem, direta ou indiretamente, usuários em comum, enquanto duas ou mais autorizações mutuamente exclusivas não podem ser associadas, direta ou indiretamente, a um mesmo papel. O modelo estabelece restrições adicionais para os números máximo e mínimo de usuários e de autorizações associados a um papel. As autorizações do XORBAC são sempre positivas e as respectivas regras

(restrições) contextuais apenas determinam as condições em que as autorizações são válidas. O XORBAC foi implementado em XOTcl (Neumann & Zdum, 2000), uma extensão orientada a objetos da linguagem de programação Tcl (Ousterhout, 1994). É um produto acabado, com versões disponíveis para ambientes *Unix-like* e Windows, mas ainda em evolução. Está livremente disponível para utilização sem fins comerciais no endereço <http://wi.wu-wien.ac.at/home/mark/xoRBAC/index.html>. Os autores não relatam a utilização do XORBAC em aplicações na área de saúde.

2.5.8 Modelo OASIS

O modelo OASIS (*Open Architecture for Securely Interworking Services*) foi desenvolvido por Bacon *et al.* (2002); Bacon & Moody (2002); Bacon *et al.* (2001); Bacon *et al.* (2003) no Laboratório de Computação da Universidade de Cambridge, na Inglaterra, visando atender os requisitos dependentes de contexto encontrados em sistemas grandes e complexos, como o PEP. Assim como o TCM (v. seção 2.5.4), o OASIS é candidato a modelo de autorização para implementar a política nacional de confidencialidade para o PEP proposta pelo Serviço Nacional de Saúde na Grã-Bretanha (National Health Service, 2002). O OASIS estende o CABP Básico com a introdução dos conceitos de *designação*, *papéis parametrizados com ativação condicionada por regras* e *regras de autorização*. Uma designação é uma autorização para um usuário ativar um ou mais papéis. É concedida por meio de certificados X.509 e o conjunto de todas as designações válidas estabelecem a relação usuário-papel no OASIS. Após a autenticação, o usuário inicia uma sessão solicitando a ativação de um ou mais papéis de sua designação. Porém, a ativação de cada papel está condicionada a satisfação de uma *regra de ativação de papel*, especificada em cláusulas de Horn (Horn, 1951), na seguinte forma: $x_1, x_2, \dots, x_n \mid p$, onde x_i para $1 \leq i \leq n$, estabelece todas as condições que devem ser satisfeitas para um usuário ativar o papel p no contexto de uma sessão particular. Por exemplo, a regra seguinte *Médico*(*medID?*), *estáNoTurno*(*medID*), *responsávelPeloPaciente*(*medID*, *pacID?*) $\mid$ *MédicoResponsável*(*medID*, *pacID*) determina que um usuário somente pode ativar o papel parametrizado *MédicoResponsável* para assistir o paciente identificado por *pacID* se ele tem o papel *Médico* previamente ativado (1ª condição), se está no seu turno de trabalho (2ª condição) e se ele é o médico responsável pelo paciente. As variáveis prefixadas com ? são parâmetros de saída e as demais (parâmetros de entrada) têm os seus valores definidos antes da avaliação da regra. Os valores de entrada e de saída de um mesmo parâmetro devem casar para uma avaliação bem

sucedida da regra, num esquema similar ao processo de unificação de Prolog. Uma regra de autorização impõe restrições contextuais que devem ser satisfeitas para associar dinamicamente uma autorização a um papel, sendo também definida por uma cláusula de Horn. A regra de autorização seguinte *MédicoResponsável*(*medID?*, *pacID?*), *pacienteInternado*(*pacID*) | *LerPEP*(*pacID*) estabelece que um usuário com o papel ativado *MédicoResponsável*, parametrizado para o paciente identificado por *pacID*, somente pode ler o PEP se o paciente estiver internado. As regras no OASIS são ativas porque mudanças nos contextos refletem-se dinamicamente nos papéis ativados e nas autorizações associadas a um papel. Por exemplo, se um médico deixa de ser responsável por um paciente, os papéis ativos dependentes dessa condição são desativados de imediato.

O OASIS não suporta hierarquias de papéis, e portanto, não há herança de autorizações. Os autores argumentam que tais hierarquias são impraticáveis em ambientes altamente distribuídos, onde não há uma administração globalmente centralizada de nomes de papéis e autorizações. Pelo mesmo motivo, são usadas as designações por meio de certificados X.509 para estabelecer a relação usuário-papel. A separação de responsabilidades dinâmica pode ser configurada no OASIS por meio de regras de ativação de papéis. As autorizações são positivas e as respectivas regras apenas determinam as condições que devem ser satisfeitas para que uma vinculação papel-autorização exista. Regras contextuais, portanto, podem ser definidas em papéis, como condição para ativação/desativação pelo usuário, ou na relação papel-autorização, como condição para vincular uma autorização a um papel. Os autores relatam a implementação de um protótipo em Java com *web services*, utilizando o sistema de gerenciamento de banco de dados PostgreSQL (Monjian, 2000) para suportar regras ativas e para armazenar os dados persistentes do modelo. O protótipo foi empregado para demonstrar a aplicação do OASIS num estudo de caso contemplando requisitos da política de confidencialidade proposta para informações clínicas de pacientes na Grã-Bretanha.

2.5.9 Modelo GTRBAC

O modelo GTRBAC (*Generalized Temporal Role-Based Access Control*) foi desenvolvido por Joshi *et al.* (2003) na Universidade de Purdue, nos EUA, e generaliza o modelo TRBAC (*Temporal Role-Based Access Control*) proposto por Bertino *et al.* (2001) na Universidade de Milão, na Itália. O GTRBAC estende o RBAC Básico com a incorporação de uma linguagem para especificar várias restrições

temporais em papéis, incluindo limitações para os tempos em que podem estar habilitados ou ativos, e também restrições para as relações usuário-papel e papel-autorização. Em particular, há uma distinção entre habilitar e ativar um papel no GTRBAC. O estado habilitado de um papel é condição para sua ativação por um usuário numa sessão. Por exemplo, a restrição $\langle \text{HorárioDiurno}, \text{enable } \text{MédicoAuditor} \rangle$ determina que o papel *MédicoAuditor* somente está habilitado (pode ser ativado) no horário diurno, por exemplo, das 6:00 às 18:00 horas. Uma restrição na ativação impõe condições na duração máxima que um papel pode permanecer ativo para um usuário, ou no número máximo de usuários que compartilham uma mesma ativação de papel. Restrições temporais nas relações usuário-papel e papel-autorização estabelecem um intervalo específico ou a duração na qual um usuário ou uma autorização permanecem associados a um papel. Por exemplo, a restrição $\langle (S, Q, X), \text{assign}_{\text{u}} \text{João to } \text{MédicoPlantonista} \rangle$ determina que o usuário João somente está associado ao papel *MédicoPlantonista* na segunda-feira, na quarta-feira e na sexta-feira, dias dos seus plantões. O GTRBAC ainda permite a definição de gatilhos (*triggers*) para expressar dependências entre eventos, para capturas de eventos passados, bem como para definir eventos futuros com base em eventos passados. Por exemplo, a restrição $\langle (\text{activate } \text{EnfermeiroPreceptor for Marta}) \& (\text{Location}() = \text{"Enfermaria"}) \rightarrow \text{enable } \text{EnfermeiroResidente after } 5 \text{ min} \rangle$ determina que o papel *EnfermeiroResidente* somente está disponível para ativação (habilitado) na enfermaria 5 minutos após a usuária Marta ativar o papel *EnfermeiroPreceptor*.

O GTRBAC suporta papéis estruturados em hierarquias gerais, onde as relações hierárquicas dependem também de restrições temporais, influenciando a herança de autorizações (Joshi *et al.*, 2002). O modelo também permite especificar numerosas formas de separação de responsabilidade (estática ou dinâmica) baseadas em restrições temporais, por exemplo, determinando que dois ou mais papéis conflitantes não podem estar ativos num mesmo intervalo de tempo. As autorizações são positivas e as regras contextuais são usadas para impor restrições (temporais) nas entidades e nas relações que compõem o modelo, daí sua generalidade. Bertino *et al.* (2001) relatam o desenvolvimento de um protótipo do TRBAC com funcionalidade reduzida, na plataforma Oracle, com implementação em PL/SQL. Joshi *et al.* (2003) e Bertino *et al.* (2001) consideram no futuro implementar completamente a linguagem do GTRBAC com base em SQL ou XML. A aplicação do modelo ao PEP ocorreu somente nos exemplos ilustrados nos trabalhos dos autores, não havendo relatos de estudos de caso mais elaborados.

2.6 Considerações finais

A revisão realizada neste capítulo ofereceu subsídios para apresentação do MACA no capítulo 3. Os conceitos e a terminologia introduzidos destacaram a atuação do controle de acesso em relação a outros serviços de segurança, tais como auditoria e autenticação. Discutiu-se a importância dos modelos de controle de acesso suportarem políticas de acesso *ad hoc* para o PEP e também facilidades para administração viável dessas políticas em grandes organizações de saúde. A revisão ainda examinou detalhadamente o modelo de referência para controle de acesso baseado em papéis proposto pelo NIST, discutindo suas potencialidades e suas limitações para aplicação no PEP. A especificação formal do modelo de referência permitiu conhecer com exatidão as propriedades do CABP, oferecendo elementos para análise dos modelos para CABP propostos recentemente na literatura, a fim de relacioná-los com o MACA na discussão do capítulo 6.

Modelo de autorização contextual

“Every one who has done any kind of creative work has experienced, in a greater or less degree, the state of mind in which, after a long labour, truth or beauty appears, or seems to appear, in a sudden glory —it may be only about a small matter, or it may be about the universe. (...) I think most of the best creative work, in art, in science, in literature, and in philosophy, has been the result of such a moment.”

Bertrand Russell, 1946

O MACA estende o modelo de CABP Básico proposto pelo NIST com a introdução de autorizações contextuais. Uma autorização contextual incorpora uma expressão lógica, denominada de *regra de autorização*, que é avaliada no momento em que um usuário tenta acessar um objeto numa sessão. A autorização é positiva, concedendo o acesso, quando o resultado da avaliação é verdadeiro. Caso contrário, a autorização é negativa e o acesso, proibido.

Uma regra de autorização relaciona informações do contexto de uma tentativa de acesso e que são relevantes para decisão de concedê-lo ou proibi-lo. Em geral, as informações contextuais referem-se ao usuário corrente; à localização, à data e à hora do acesso; ao objeto que se tenta acessar; e aos relacionamentos existentes entre as entidades. Tais informações são empregadas na implementação das políticas *ad hoc* requeridas para disciplinar o acesso ao prontuário eletrônico do paciente em organizações de saúde.

Adicionalmente, uma autorização no MACA pode ser do tipo *forte* ou *fraca*. Autorizações fortes expressam políticas estritas, que não podem ser revogadas; en-

quanto as fracas são usadas para definir políticas mais permissivas. Por exemplo, uma autorização forte pode expressar a seguinte política: *médicos assistentes têm a permissão para ordenar prescrições e essa política não pode ser revogada por outra autorização*. Autorizações fortes e fracas foram originalmente introduzidas por Rabbitt *et al.* (1991) num modelo de autorização para sistemas de gerenciamento de bancos de dados (SGBD) avançados, baseados em conceitos de orientação a objetos e de modelagem de dados semânticos. Posteriormente, Bertino *et al.* (1999) utilizaram os conceitos de autorizações fortes e fracas num modelo discricionário de controle de acesso para sistemas de gerenciamento de bancos de dados relacionais.

O MACA suporta a hierarquia de papéis em estrutura de árvore invertida, com herança de autorizações associadas aos papéis mais gerais pelos papéis descendentes. Entretanto, permite que uma autorização herdada seja revogada pela sobreposição de outra autorização. Assim, uma autorização que proíba o acesso, num papel de menor responsabilidade, pode ser modificada para concedê-lo num papel descendente com mais responsabilidades. A revogação também acontece quando uma autorização forte é definida. No caso, a autorização forte de um papel revoga quaisquer autorizações fracas equivalentes existentes em papéis descendentes. O modelo ainda provê a ativação automática de papéis e a separação estática e dinâmica de responsabilidades com base em conflitos fortes e fracos entre autorizações.

O capítulo está organizado da seguinte forma. A seção 3.1 define o conceito de contextos e a seção 3.2 introduz a linguagem de expressões lógicas das regras de autorização. A especificação formal do MACA, na notação Z, acrescentando contextos e regras às autorizações do CABP do NIST, é apresentada na seção 3.3. A seção 3.4 ilustra, com um exemplo, a capacidade do modelo proposto em atender requisitos para políticas de acesso ao PEP. A seção 3.5 faz considerações sobre alternativas para políticas de acesso administrativas para o CABP utilizando o MACA. Finalmente, a seção 3.6 traz as considerações finais do capítulo.

3.1 Contextos

Um contexto é qualquer informação que pode ser usada para caracterizar uma entidade (pessoa, lugar ou objeto) considerada relevante para a interação entre um usuário e uma aplicação, incluindo-se aí o próprio usuário e a própria aplicação (Dey, 2000). Denota informações ambientais importantes para decidir se um acesso será concedido ou proibido, no momento em que um usuário tenta efetuá-lo. No MACA, há o contexto do usuário corrente (aquele que iniciou uma sessão e que solicita uma

autorização de acesso); o contexto temporal, com informações sobre hora e data; o contexto de rede, com informações de redes e de comunicação; e outros contextos que podem ser livremente definidos e implementados pelos formuladores e administradores da política de acesso. Aqui, apenas os aspectos conceituais e semânticos dos contextos são destacados, enquanto a sessão 4.2.2.2, do capítulo 4, considera os aspectos de implementação.

As informações disponíveis num contexto são recuperadas de acordo com a sintaxe `<nome do contexto>.<nome>`, onde o primeiro termo designa o nome do contexto e o segundo termo designa um dos seguintes componentes:

- *Variável contextual*: denota um valor atômico obtido do ambiente numa tentativa de acesso. Por exemplo, as variáveis `usuárioCtx.nome` e `usuárioCtx.registro_profissional` permitem recuperar, do contexto do usuário corrente (`usuárioCtx`), o nome e o código com que foi registrado num conselho profissional, respectivamente. Outras variáveis permitem recuperar, do contexto temporal (`dtCtx`), a data (`dtCtx.data`), a hora (`dtCtx.hora`) e o dia da semana (`dtCtx.dia_semana`) atuais; e do contexto de rede (`netCtx`), os endereços IP (`netCtx.peer_ip`) ou DNS (`netCtx.peer_dns`) do local onde se tenta acessar uma informação;
- *Conjunto contextual*: denota uma coleção de valores determinados e diferenciáveis encontrados no ambiente de uma tentativa de acesso. Por exemplo, o conjunto que representa os dias úteis da semana (`dtCtx.dias_úteis`); o conjunto que contém os endereços (IP ou DNS) das estações de trabalho das salas de emergência de um hospital (`netCtx.domínios_emergência`); ou o conjunto que representa os pacientes internados num estabelecimento de saúde (`pacCtx.pacientes_internados`), obtido do contexto de paciente (`pacCtx`);
- *Função contextual*: expressa relações entre as entidades existentes no ambiente de uma tentativa de acesso e que não podem ser denotadas por variáveis ou conjuntos contextuais. Por exemplo, a função

```
pacCtx.assistido_por(umCodPac, usuárioCtx.registro_profissional)
```

do contexto de paciente, informa se o paciente identificado pelo parâmetro `umCodPac` é (ou foi) assistido pelo usuário corrente com

registro profissional denotado pela variável
usuárioCtx.registro_profissional.

É importante frisar que os contextos devem refletir, com alto nível de abstração, as entidades e os relacionamentos existentes entre elas no ambiente onde o acesso é realizado, a fim de facilitar a definição das políticas *ad doc*. Recomenda-se que os nomes das variáveis, conjuntos e funções contextuais pertençam ao vocabulário da cultura organizacional para favorecer o diálogo entre os formuladores e os administradores/implementadores da política de acesso. Por exemplo, em vez de utilizar termos de baixo nível para identificar um local, como os endereços IP ou DNS do contexto de rede, aconselha-se usar os nomes habituais da organização, como “Sala de Emergência 1”. Ou seja, a escolha de contextos adequados vai facilitar a interpretação das políticas de acesso institucionais, expressas em linguagem natural, em políticas técnicas, definidas com regras de autorização expressas em linguagem formal.

3.2 Regras de autorização

Uma regra de autorização relaciona informações contextuais em expressões lógicas que especificam uma política de acesso para um objeto protegido. Regras são definidas numa linguagem de expressões lógicas capaz de recuperar os valores provenientes de variáveis, conjuntos e funções contextuais para relacioná-los por meio de operadores aritméticos (+, -, *, /, % – módulo), relacionais (>, <, >=, <=, =, !=, in – pertinência) e booleanos (&, |, !). O Anexo A apresenta a gramática formal que especifica completamente a sintaxe da linguagem. As subseções seguintes introduzem informalmente a linguagem, considerando os aspectos conceituais, semânticos e de ordem prática.

3.2.1 Valores

Os tipos de valores primitivos (ou atômicos) suportados numa regra de autorização são os seguintes:

- *Booleano*: tipo que contém os valores lógicos `true` (verdadeiro) ou `false` (falso);
- *Inteiro*: tipo que contém valores de números inteiros no intervalo de -2147483648 até 2147483647;
- *Texto*: tipo que contém valores que são cadeias de caracteres, variando desde a cadeia vazia "" até cadeias cujo tamanho dependerá da

memória disponível no ambiente onde a linguagem é interpretada. Valores do tipo texto são primitivos porque não pode ser subdivididos, isto é, são atômicos.

Os valores dos tipos primitivos acima podem ser construídos literalmente na linguagem. Por exemplo, as duas regras mais simples expressas na linguagem são os valores booleanos literais

```
true
```

e

```
false
```

exprimindo regras que permitem e proíbem o acesso em quaisquer circunstâncias, respectivamente. Nota-se que uma regra de autorização, quando avaliada, sempre deve retornar um valor do tipo booleano, caso contrário, ocorrerá um erro pela impossibilidade de se decidir pela concessão ou negação do acesso. Por exemplo, as expressões literais abaixo

```
10
```

ou

```
"Sala de emergência"
```

são válidas na linguagem, mas não constituem uma regra de autorização, pois, quando avaliadas, resultam em valores dos tipos inteiro e texto, respectivamente.

A linguagem também permite obter valores primitivos a partir de variáveis ou funções existentes em contextos. A variável contextual

```
dtCtx.dia_semana
```

quando avaliada, resulta num valor do tipo texto informando o dia atual, que pode ser "seg", "ter", "qua", "qui" ou "sex". Já a função contextual

```
pacCtx.idade_em_dias("123456-H")
```

quando aplicada, resulta num inteiro correspondente à idade, em número de dias, do paciente identificado pelo código 123456-H. Conjuntos contextuais são compostos de valores mais simples, que podem ser atômicos. A linguagem permite definir conjuntos a partir de literais usando os valores dos inteiros de uma faixa, conforme exemplificado na expressão

```
10..30
```

que especifica um conjunto de inteiros com os valores variando desde 10 até 30, inclusive. Assim, valores que são conjuntos podem ser construídos literalmente (apenas para faixas de inteiros) ou obtidos a partir de conjuntos ou funções contextuais.

O conjunto contextual

```
usuáriosCtx.papéis_ativos
```

quando avaliado, retorna o conjunto de papéis ativados pelo usuário em suas sessões. Já a função contextual

```
pacCtx.pacientes_agendados(dtCtx.data)
```

quando aplicada, resulta num conjunto de valores que representam os pacientes com consultas ou exames agendados na data passada como parâmetro, no caso, a data corrente.

Portanto, os tipos de valores pré-definidos na linguagem são: booleano, inteiro, texto e conjunto. A existência de outros tipos de valores, primitivos ou compostos, depende da definição de contextos específicos.

3.2.2 Operadores

Os operadores da linguagem manipulam valores transformando-os em novos valores e podem ser aritméticos, relacionais ou booleanos. Além dos operadores pré-definidos, apresentados aqui, novos operadores podem ser criados com a definição de funções contextuais. Essa capacidade, aliada a possibilidade de criação de novos tipos de valores em contextos, torna a linguagem extensível, facilitando o tratamento de valores mais elaborados para estabelecer relacionamentos complexos.

3.2.2.1 Aritméticos

A linguagem suporta os operadores aritméticos para adição (+), subtração (-), multiplicação (*), divisão (/) e módulo (%; resto da operação de divisão) de números inteiros.

3.2.2.2 Relacionais

Os operadores relacionais maior que (>), menor que (<), maior ou igual (>=), menor ou igual (<=), igual (=), diferente (!=) e de pertinência (in) são suportados na linguagem. Os operadores >, <, >=, <= atuam em valores do tipo inteiro, enquanto os operadores = e != são usados com quaisquer valores. A regra de autorização seguinte

```
pacCtx.idade_em_dias(umCodPac) >= 1825
```

estabelece que um acesso somente é concedido se a idade em dias do paciente identificado por `umCodPac` é superior a 1825 dias, o que corresponde a 5 anos. A regra a seguir

```
pacCtx.médico_responsável(umCodPac) = usuárioCtx.registro_profissional
```

define que um acesso somente é concedido ao usuário corrente se ele é o médico responsável pelo paciente identificado por `umCodPac`.

O operador relacional `in` testa a pertinência de um valor num conjunto de valores. Por exemplo, a regra abaixo

```
dtCtx.dia_semana in dtCtx.dias_úteis
```

estabelece que um acesso somente é concedido se o dia da semana atual é um dia útil, ou seja, o acesso é proibido nos fins de semana. A aplicação do operador `in` na expressão

```
umCodPac in pacCtx.pacientes_agendados(dtCtx.data)
```

determina que um acesso somente é concedido se o paciente identificado por `umCodPac` tem uma consulta ou exame agendados na data corrente.

3.2.2.3 Booleanos

A linguagem suporta os operadores booleanos para conjunção (`&`), disjunção (`|`) e negação (`!`). Por exemplo, a regra de autorização seguinte

```
umCodPac in pacCtx.pacientes_agendados(dtCtx.data) |  
umCodPac in pacCtx.pacientes_internados
```

estabelece que um acesso somente é concedido se o paciente identificado por `umCodPac` está internado ou tem uma consulta ou exame agendados na data corrente. Uma versão estendida da regra anterior

```
(umCodPac in pacCtx.pacientes_agendados(dtCtx.data) |  
umCodPac in pacCtx.pacientes_internados) &  
usuárioCtx.está_no_turno_de_trabalho
```

define que um acesso somente é concedido para usuários correntes que estão em seu turno de trabalho quando o paciente identificado por `umCodPac` está internado ou tem uma consulta ou exame agendados na data atual.

3.2.3 Parametrização

Há casos em que uma regra precisa ser parametrizada para permitir a passagem de informações contextuais do ambiente da aplicação (que solicita uma autorização)

para a regra, no momento de sua avaliação. É o que ocorre em regras como a apresentada abaixo

```
umCodPac in pacCtx.pacientes_internados
```

onde o valor do identificador `umCodPac` depende de um momento particular da aplicação que solicita o acesso. A linguagem não permite a ocorrência de identificadores livres como esse, pois não há como saber seu valor durante a avaliação da regra. Logo, todo identificador de variável, de conjunto ou de função deve subordinar-se a um contexto ou ser declarado explicitamente como um parâmetro de uma regra. A regra anterior, definida da seguinte forma,

```
exp-abs(umCodPac) {
  umCodPac in pacCtx.pacientes_internados
}
```

torna-se uma regra parametrizada bem formada. A palavra reservada **exp-abs** indica a definição de uma expressão parametrizada, ou seja, uma função anônima. O parâmetro formal `umCodPac` denota o código de identificação de um paciente passado como argumento no momento da avaliação da regra. Para uma regra parametrizada ser avaliada, é necessário que uma lista de argumentos correspondentes aos parâmetros formais seja fornecida. Por exemplo, a regra parametriza toma a seguinte forma

```
exp-abs(umCodPac) {
  umCodPac in pacCtx.pacientes_internados
} ("123456-H")
```

para ser avaliada passando-se o argumento "123456-H" como código de identificação do paciente. No processo de avaliação, o identificador do parâmetro formal (`umCodPac`) passa a denotar o valor do argumento correspondente fornecido, no caso, "123456-H". Podem ser passados como argumentos de uma regra parametrizada valores primitivos (booleanos, inteiros e texto); variáveis, conjuntos e funções contextuais; e demais expressões da linguagem, incluindo-se aí, regras parametrizadas. Para ilustrar, a expressão abaixo

```
exp-abs(umCodPac, pacientes, é_dia_válido) {
  umCodPac in pacientes &
  é_dia_válido(dtCtx.dia_semana)
} ("123456-H",
  pacCtx.pacientes_internados,
  exp-abs(umDia) {
    umDia in dtCtx.fim_de_semana
  }
)
```

recebe como argumentos um valor do tipo texto, com um código de identificação de paciente; um conjunto contextual, representando uma coleção de pacientes; e uma regra parametrizada, que testa se um dia qualquer é um dia válido. No processo de avaliação, o identificador do parâmetro formal `umCodPac` passa a denotar o valor do argumento "123456-H", o identificador `pacientes` denota o valor do conjunto contextual `pacCtx.pacientes_internados`, e o identificador `é_dia_válido` denota a regra parametrizada correspondente ao terceiro argumento. Isso permite que a aplicação, a cada solicitação de autorização, varie o identificador do paciente, o conjunto de pacientes e a função que testa se um dia é válido. Por exemplo, numa outra circunstância, a aplicação cliente poderia passar, no lugar do conjunto de pacientes internados, o conjunto de pacientes com consulta ou exames agendados na data atual (`pacCtx.pacientes_agendados(dtCtx.data)`).

3.3 A especificação formal do MACA

A especificação formal do MACA estende a especificação em Z do CABP Hierárquico do NIST, vista na seção 2.4.1.1, com o acréscimo de autorizações contextuais, de autorizações positivas e negativas, de autorizações fortes e fracas, da possibilidade de revogar-se autorizações e da separação de responsabilidades estática e dinâmica baseadas em conflitos fortes e fracos entre autorizações. A especificação do MACA é apresentada paulatinamente, nas seguintes seções:

- *MACA básico (seção 3.3.1)*: incorpora contextos e regras às autorizações do CABP Básico, permitindo-as que sejam positivas ou negativas, fortes ou fracas;
- *MACA hierárquico (seção 3.3.2)*: adiciona a hierarquia de papéis com estrutura de árvore invertida ao MACA Básico, estendendo as propriedades do CABP Hierárquico com a introdução dos conceitos de autorizações válidas e de autorizações revogadas;
- *MACA restrito (seção 3.3.3)*: estabelece restrições para separação de responsabilidades estática e dinâmica baseada em conflitos fortes e fracos entre autorizações.
- *Especificação do algoritmo de decisão de acesso (seção 3.3.4)*: define as propriedades do algoritmo de decisão de acesso, que deve considerar: apenas as autorizações válidas (não revogadas); a prioridade das autorizações fortes sobre as fracas; as restrições para separação de responsabilidades dinâmica baseada nos conflitos fortes entre autorizações; a ativação automática de papéis.

Os esquemas Z nas seções 3.3.1 até 3.3.3 descrevem o espaço do estado dos componentes do MACA e constituem um único modelo, mas que é apresentado em partes para facilitar o entendimento. O esquema da seção 3.3.4 apenas descreve como o estado do MACA é utilizado e afetado pelo algoritmo de decisão de acesso.

3.3.1 MACA básico

O MACA Básico inclui os conjuntos de entidades e as relações da especificação do CABP Básico do NIST (v. p. 28). Em adição, introduz os conjuntos com contextos, com regras e com parâmetros de regras, e os tipos para especificação de autorizações contextuais, positivas ou negativas, fortes ou fracas.

Os nomes para identificar as entidades do CABP, os contextos, as regras e seus parâmetros no MACA são representados pelos tipos básicos em Z declarados a seguir

$[NOME, CONTEXTO, REGRA, PARAMETRO],$

respectivamente. O tipo *NOME* representa os identificadores possíveis para usuários, papéis, objetos, operações e sessões. O tipo básico *CONTEXTO* representa o conjunto de todos os contextos possíveis, enquanto o tipo *REGRA* representa todas as regras (expressões cuja avaliação resulta num valor booleano) bem formadas da linguagem descrita na seção 3.2. O tipo *PARAMETRO* representa os possíveis argumentos que podem ser passados como parâmetros de regras. Aos conjuntos de entidades do CABP Básico do NIST (v. p. 26) é acrescentado um conjunto de contextos, conforme especificado no esquema seguinte:

Conjuntos de Entidades do MACA

usuarios: $\mathbb{P} \text{ NOME}$

papeis: $\mathbb{P} \text{ NOME}$

obs: $\mathbb{P} \text{ NOME}$

ops: $\mathbb{P} \text{ NOME}$

sessoes: $\mathbb{P} \text{ NOME}$

ctxs: $\mathbb{P} \text{ CONTEXTO}$

$\forall c_1, c_2: \{usuarios, papeis, obs, ops, sessoes\} \bullet c_1 \cap c_2 = \emptyset$

O conjunto *ctxs* denota os contextos disponíveis para a avaliação de regras de autorização num estado específico do sistema. Para o nível de abstração da especificação, é irrelevante determinar concretamente quais são esses contextos.

A definição do seguinte tipo livre em Z especifica os valores que indicam se uma autorização é forte ou fraca.

$TIPO_AUTORIZACAO ::= forte \mid fraca$

Um tipo livre em Z é especificado pela enumeração dos seus valores separados pelo símbolo “|”. Assim, uma variável declarada com o tipo *TIPO_AUTORIZACAO* só pode assumir um dos seguintes valores: forte ou fraca.

O tipo declarado a seguir especifica os valores que indicam o tipo de privilégio de uma autorização

$$TIPO_PRIVILEGIO == \{ +, - \} \cup REGRA$$

que pode ser positivo, negativo ou uma regra contextual.

O esquema a seguir especifica o MACA Básico, que estende o CABP Básico do NIST (v. p. 28) introduzindo autorizações contextuais e novas funções auxiliares.

MACA_Basico

ConjuntosdeEntidadesdoMACA

$up: usuarios \leftrightarrow papeis$

$usuarios_associados: papeis \rightarrow \mathbb{P} usuarios$

$autorizacoes: \mathbb{P} (TIPO_PRIVILEGIO \times ops \times obs \times TIPO_AUTORIZACAO)$

$pa: papeis \leftrightarrow autorizacoes$

$autorizacoes_associadas: papeis \rightarrow \mathbb{P} autorizacoes$

$op: autorizacoes \rightarrow ops$

$ob: autorizacoes \rightarrow obs$

$tp: autorizacoes \rightarrow TIPO_PRIVILEGIO$

$ta: autorizacoes \rightarrow TIPO_AUTORIZACAO$

$sessoes_do_usuario: usuarios \rightarrow \mathbb{P} sessoes$

$usuario_da_sessao: sessoes \rightarrow usuarios$

$papeis_da_sessao: sessoes \rightarrow \mathbb{P} papeis$

$autorizacoes_da_sessao: sessoes \rightarrow \mathbb{P} autorizacoes$

$\forall p: \text{dom } usuarios_associados \bullet usuarios_associados(p) = \{u \in usuarios \mid u \mapsto p \in up\}$

$\forall p: \text{dom } autorizacoes_associadas \bullet$

$autorizacoes_associadas(p) = \{a \in autorizacoes \mid p \mapsto a \in pa\} \wedge$

$\exists a_1, a_2: autorizacoes_associadas(p) \mid a_1 \neq a_2 \bullet op(a_1) = op(a_2) \wedge ob(a_1) = ob(a_2)$

$\forall a: \text{dom } op \mid a = (tpr, opr, obj, tpa) \bullet op(a) = opr$

$\forall a: \text{dom } ob \mid a = (tpr, opr, obj, tpa) \bullet ob(a) = obj$

$\forall a: \text{dom } tp \mid a = (tpr, opr, obj, tpa) \bullet tp(a) = tpr$

$\forall a: \text{dom } ta \mid a = (tpr, opr, obj, tpa) \bullet ta(a) = tpa$

$\forall s: \text{dom } usuario_da_sessao \bullet s \in sessoes_do_usuario(usuario_da_sessao(s))$

$\forall s: \text{dom } papeis_da_sessao \bullet$

$papeis_da_sessao(s) \subseteq \{p \in papeis \mid usuario_da_sessao(s) \mapsto p \in up\}$

$\forall s: \text{dom } autorizacoes_da_sessao \bullet$

$autorizacoes_da_sessao(s) \subseteq \{a \in autorizacoes_associadas(p) \mid p \in papeis_da_sessao(s)\}$

A primeira linha da declaração do esquema *MACA_Básico* inclui as definições do esquema *ConjuntosdeEntidadesdoMACA*. A principal diferença do *MACA_Básico* em relação ao CABP Básico do NIST é a redefinição da relação *autorizacoes*, possibi-

litando que seus elementos tenham um tipo de privilégio (*TIPO_PRIVILEGIO*) e um tipo de autorização (*TIPO_AUTORIZACAO*). O tipo de privilégio especifica se uma autorização é uma permissão ou uma proibição para executar uma operação relacionada a um objeto. A função auxiliar *tp* foi adicionada ao esquema *MACA_Básico* para obter o tipo do privilégio de uma autorização (v. no invariante a definição da função). Quando o tipo do privilégio é positivo, o acesso é permitido, quando ele é negativo, o acesso é proibido. As duas tuplas abaixo são exemplos de elementos da relação *autorizacoes*

```
⟨+, VerLaudo, PEP, fraca⟩
⟨-, VerLaudo, PEP, fraca⟩
```

e especificam autorizações fracas que concedem e que proíbem o acesso para ver um laudo (operação) de um PEP (objeto), respectivamente. Nos casos em que o tipo de privilégio é uma regra de autorização, a concessão ou a negação do acesso depende do resultado da avaliação da regra no momento em que um usuário tenta acessar o objeto numa sessão particular. Se o resultado da avaliação for verdadeiro, a autorização é considerada positiva e o acesso é concedido. Caso contrário, a autorização é considerada negativa e o acesso é proibido. A autorização contextual

```
⟨exp-abs (umCodPac) {
    umCodPac in pacCtx.pacientes_internados
}, VerLaudo, PEP, fraca⟩
```

especifica que o acesso a laudos do PEP do paciente identificado pelo parâmetro *umCodPac* somente é permitido quando ele está internado, por exemplo. Uma autorização com regra parametrizada é denominada de *autorização parametrizada*.

O tipo da autorização indica se ela é forte ou fraca. Autorizações fortes estabelecem políticas absolutas, que não podem ser revogadas, enquanto as fracas são usadas para definir políticas mais permissivas. Os critérios para revogação de autorizações são estabelecidos na presença da hierarquia de papéis (v. seção 3.3.2). A função auxiliar *ta* foi adicionada ao esquema *MACA_Básico* para saber se o tipo de uma autorização é forte ou fraca (v. no invariante a definição da função).

Em relação ao invariante do esquema *CABP_Básico* do NIST (v. p. 28), o *MACA_Básico* introduz uma restrição adicional para as autorizações associadas a um papel (v. segundo quantificador universal): não pode haver duas ou mais autorizações diferentes, associadas a um mesmo papel, referindo-se a uma mesma operação e a um mesmo objeto. Por exemplo, as duas autorizações a seguir

```
⟨Médico, ⟨+, VerLaudo, PEP, fraca⟩⟩
⟨Médico, ⟨-, VerLaudo, PEP, forte⟩⟩
```

na relação *pa* (papel-autorização), não podem existir associadas ao papel *Médico*. Embora diferentes, referem-se a uma mesma operação (*VerLaudo*) e a um mesmo objeto (*PEP*). A definição delas implicaria na violação do invariante do estado do modelo, o que não é permitido. Por ora, são essas as restrições impostas às autorizações do MACA. Restrições adicionais, dependentes do tipo da autorização e do tipo do privilégio, são introduzidas nas seções seguintes para o MACA Hierárquico e o MACA Restrito.

3.3.2 MACA hierárquico

O MACA Hierárquico organiza os papéis do MACA Básico numa estrutura de árvore invertida, seguindo a especificação do CABP Hierárquico do NIST. O esquema *MACA_HierarquiaGeral* redefine o esquema *CABP_Hierarquico* (v. p. 32) para estabelecer uma hierarquia geral de papéis sobre o modelo do MACA Básico.

<i>MACA_HierarquiaGeral</i>
<i>MACA_Basico</i>
$hp: \text{papeis} \leftrightarrow \text{papeis}$
$\text{usuarios_associados_hp}: \text{papeis} \rightarrow \mathbb{P} \text{ usuarios}$
$\text{autorizacoes_associadas_hp}: \text{papeis} \rightarrow \mathbb{P} \text{ autorizacoes}$
$\forall p_r, p_2: hp \bullet p_1 \succeq p_2$
$\forall p: \text{dom usuarios_associados_hp} \bullet$ $\text{usuarios_associados_hp}(p) = \{u \in \text{usuarios} \mid \exists p_1: \text{papeis} \mid p_1 \succeq p \in hp \bullet u \mapsto p_1 \in up\}$
$\forall p: \text{dom autorizacoes_associadas_hp} \bullet$ $\text{autorizacoes_associadas_hp}(p) = \{a \in \text{autorizacoes} \mid \exists p_1: \text{papeis} \mid p \succeq p_1 \in hp \bullet p_1 \mapsto a \in pa\}$
$\forall p_1 \succeq p_2: hp \bullet \text{autorizacoes_associadas_hp}(p_2) \subseteq \text{autorizacoes_associadas_hp}(p) \wedge$ $\text{usuarios_associados_hp}(p) \subseteq \text{usuarios_associados_hp}(p_2)$

O esquema *MACA_HierarquiaGeral* tem as mesmas declarações e o mesmo invariante de *CABP_Hierarquico*, exceto que atua nos componentes definidos no esquema *MACA_Basico*—incluído na primeira linha. A hierarquia geral anterior é restringida a uma estrutura de árvore invertida no esquema

<i>MACA_ArvoreInvertida</i>
<i>MACA_HierarquiaGeral</i>
$\text{ascendentes}: \text{papeis} \rightarrow \mathbb{P} \text{ papeis}$
$\text{ascendentes_imediatos}: \text{papeis} \rightarrow \mathbb{P} \text{ papeis}$
$\forall p: \text{dom ascendentes} \bullet \text{ascendentes}(p) = \{p_1 \in \text{papeis} \mid p \succeq p_1 \in hp\}$
$\forall p: \text{dom ascendentes_imediatos} \bullet \text{ascendentes_imediatos}(p) =$ $\{p_1 \in \text{ascendentes}(p) \mid \nexists p_2: \text{papeis} \mid p_1 \neq p_2 \wedge p_2 \neq p \bullet p \succeq p_2 \in hp \wedge p_2 \succeq p_1 \in hp\}$
$\forall p, p_r, p_2: \text{papeis} \bullet$ $p_1 \in \text{ascendentes_imediatos}(p) \wedge p_2 \in \text{ascendentes_imediatos}(p) \Rightarrow p_1 = p_2$

Figura 4 – Hierarquia de papéis em estrutura de árvore invertida.

que segue a mesma definição do esquema equivalente do CABP do NIST (v. p. 33). Em suma, os esquemas *MACA_HierarquiaGeral* e *MACA_ArvoreInvertida* apenas especificam a hierarquização dos papéis em estrutura de árvore invertida (e. g., Figura 4), com as mesmas propriedades da hierarquia de papéis do CABP do NIST, resumidas nos itens seguintes:

- As autorizações associadas a papéis de menor responsabilidade (e. g., *Profissional de Saúde*) são herdadas pelos papéis de maior responsabilidade (e. g., *Auxiliar de Enfermagem*), seus descendentes na hierarquia;
- Os usuários associados a papéis de maior responsabilidade (e. g., *Médico Assistente*) são também usuários (indiretos) de papéis de menor responsabilidade (e. g., *Profissional de Saúde*), seus ascendentes na hierarquia;
- Um papel pode ter vários descendentes diretos, mas no máximo um ancestral imediato.

Observe-se que a herança de autorizações —positivas, negativas ou com regras— pode acarretar ambigüidades ao modelo. Por exemplo, considerem-se as seguintes autorizações:

```

<Usuário, <- , VerLaudo, PEP, fraca>>
<Auxiliar de Enfermagem,
  <exp-abs(umCodPac) {
    umCodPac in pacCtx.pacientes_internados
  }, VerLaudo, PEP, fraca>>

```

Como o papel *Auxiliar de Enfermagem* é descendente de *Usuário*, então as autorizações associadas a usuários também estão associadas a auxiliares de enfermagem. O que fazer, então, quando um usuário com o papel de auxiliar de enfermagem tenta ver um laudo do PEP? Qual, das duas autorizações disponíveis, é considerada? Ambas são *equivalentes* porque referem-se ao mesmo objeto (PEP) e a mesma operação (VerLaudo). O MACA responde a questão estabelecendo que, uma autorização herdada, de um papel de menor responsabilidade, é *sobreposta*, num papel de maior responsabilidade, sempre que uma autorização equivalente for definida nesse. Diz-se que a autorização sobreposta num papel é *revogada* pela autorização equivalente. As autorizações não revogadas de um papel são denominadas de *autorizações válidas*, sendo as únicas consideradas para decidir um acesso.

Entretanto, a revogação de autorizações não é arbitrária. O tipo da autorização disciplina o mecanismo de revogação. Apenas as autorizações do tipo fraca podem ser revogadas. No exemplo anterior, com a primeira autorização do tipo forte

```

<Usuário, <- , VerLaudo, PEP, forte>>
<Auxiliar de Enfermagem,
  <exp-abs (umCodPac) {
    umCodPac in pacCtx.pacientes_internados
  }, VerLaudo, PEP, fraca>>

```

a sobreposição dela no papel *Auxiliar de Enfermagem* não tem o efeito de revogá-la. Pelo contrário, uma autorização forte, num papel de menor responsabilidade, tem o poder de revogar as eventuais autorizações fracas equivalentes definidas em papéis de maior responsabilidade, descendentes dele na hierarquia. Ou seja, uma autorização forte prevalece sobre as autorizações fracas equivalentes que a sobrepõem. Nesse exemplo, a proibição de ver laudos do PEP é absoluta e não pode ser revogada por nenhuma sobreposição de autorizações fracas.

Nos casos em que ambas as autorizações são do tipo forte, há um impasse, e o administrador da política deve enfraquecer uma ou as duas autorizações. O MACA não admite a existência de duas ou mais autorizações fortes equivalentes numa mesma linha da hierarquia de papéis.

O esquema seguinte, *MACA_Hierarquico*, especifica os conceitos de autorizações válidas e revogadas para papéis estruturados em árvores invertidas, introduzidos informalmente nos exemplos desta seção.

<i>MACA_Hierarquico</i>
<i>MACA_ArvoreInvertida</i>
<i>autorizacoes_validas</i> : <i>papeis</i> $\rightarrow$ $\mathbb{P}$ <i>autorizacoes</i>
<i>autorizacoes_revogadas</i> : <i>papeis</i> $\rightarrow$ $\mathbb{P}$ <i>autorizacoes</i>
$\forall p_1, p_2 : \text{papeis} \mid p_1 \neq p_2 \wedge p_1 \in \text{ascendentes}(p_2) \bullet$ $\exists a_1 : \text{autorizacoes_associadas}(p_1); a_2 : \text{autorizacoes_associadas}(p_2) \bullet$ $op(a_1) = op(a_2) \wedge ob(a_1) = ob(a_2) \wedge ta(a_1) = ta(a_2) = \text{forte}$ $\forall p : \text{dom autorizacoes_validas} \bullet$ $\text{autorizacoes_validas}(p) =$ $\{ a \in \text{autorizacoes_associadas}(p) \mid (p_1 \in \text{ascendentes}(p)) \wedge$ $(\exists a_2 : \text{autorizacoes_associadas}(p_2) \mid p_2 \in \text{ascendentes}(p) \wedge p_2 \neq p_1 \bullet$ $op(a_2) = op(a) \wedge ob(a_2) = ob(a) \wedge ta(a_2) = \text{forte}) \wedge$ $(ta(a) = \text{fraca} \Rightarrow$ $(\exists a_2 : \text{autorizacoes_associadas}(p_2) \mid p_2 \in (\text{ascendentes}(p) \setminus \text{ascendentes}(p)) \bullet$ $op(a_2) = op(a) \wedge ob(a_2) = ob(a))) \}$ $\forall p : \text{papeis} \bullet$ $\text{autorizacoes_revogadas}(p) = \text{autorizacoes_associadas_hp}(p) \setminus \text{autorizacoes_validas}(p)$

O primeiro quantificador universal do invariante estabelece que, para quaisquer dois papéis distintos, numa mesma linha hierárquica, não podem existir, associados a cada um deles, autorizações equivalentes do tipo forte. Tal propriedade é necessária ao modelo para evitar a revogação de autorizações fortes.

A função *autorizacoes_validas* mapeia todas as autorizações válidas associadas, direta ou indiretamente, a um papel. O invariante do esquema estabelece a seguinte conjunção de três condições, no segundo quantificador universal, para que uma autorização seja considerada válida para um papel *p*:

1. Que a autorização esteja associada a algum papel *p*, ascendente do papel *p*. Observe-se que todo papel é ascendente dele mesmo, de acordo com a definição da função *ascendente* no esquema *MACA_ArvoreInvertida*. Logo, as autorizações diretamente associadas ao papel *p* estão no rol das autorizações potencialmente válidas;
2. Que não exista uma autorização forte, equivalente a essa, associada a algum papel *p₂* ascendente do papel *p*, onde *p₂* é diferente *p*. Por exemplo, sendo *p* o papel *Auxiliar de Enfermagem*, a autorização

```

(Auxiliar de Enfermagem,
  (exp-abs (umCodPac) {
    umCodPac in pacCtx.pacientes_internados
  }, VerLaudo, PEP, fraca))

```

satisfaz essa condição quando não existir uma autorização forte equivalente (definida para a mesma operação e o mesmo recurso) num papel *p₂* estritamente ascendente (*Paramédico*, *Profissional de Saúde*,

Usuário) do papel *Auxiliar de Enfermagem*. A existência de tal autorização é condição suficiente para revogar a autorização exemplificada. Note-se que, o predicado do quantificador existencial incide “recursivamente” sobre as autorizações herdadas pelo papel p . Supondo ainda p igual ao papel *Auxiliar de Enfermagem*, a autorização

```

<Profissional de Saúde,
  <exp-abs (umCodPac) {
    umCodPac in pacCtx.pacientes_internados
  }, VerPrescrição, PEP, fraca>>

```

satisfaz essa condição, no papel *Auxiliar de Enfermagem*, se não existir uma autorização forte equivalente num papel p_2 estritamente ascendente (*Usuário*) do papel *Profissional de Saúde* (p_1). Em suma, a condição do item 2 determina que uma autorização forte de um papel de menor responsabilidade prevalece sobre quaisquer autorizações fracas equivalentes existentes num papel descendente, de maior responsabilidade;

3. Caso essa autorização seja fraca, que não exista uma autorização equivalente, associada a algum papel p_2 descendente de p_1 e ascendente de p , onde p_2 não é o papel p_1 . Por exemplo, supondo p igual ao papel *Médico Assistente*, a autorização

```

<Profissional de Saúde, <-, Prescrever, PEP, fraca>>

```

satisfaz essa condição se ela for do tipo fraca e não existir uma autorização equivalente num papel p_2 pertencente ao conjunto dos papéis ascendentes de *Médico Assistente* e estritamente descendentes do papel *Profissional de Saúde* (p_1). Os papéis ascendentes de *Médico Assistente* são: *Médico Assistente*, *Médico*, *Profissional de Saúde* e *Usuário*. Os papéis estritamente descendentes de *Profissional de Saúde* são: *Médico*, *Médico Assistente*, *Médico Auditor*, *Pesquisador Clínico*, *Paramédico* e *Auxiliar de Enfermagem*. Logo, o conjunto a que p_2 pertence corresponde à intersecção desses dois conjuntos de papéis: *Médico Assistente* e *Médico*. Note-se, que a intersecção dos ascendentes de p com os descendentes estritos de p_1 é igual a diferença (operador $\setminus$) do conjunto dos ascendentes de p pelo conjunto dos ascendentes de p_1 presente no predicado do quantificador existencial.

al. Logo, para a autorização acima satisfazer a condição, não podem existir autorizações equivalentes nos papéis *Médico* ou *Médico Assistente*. Em suma, a condição do item 3 estabelece que, para ser válida, uma autorização fraca, herdada num papel, não pode ter sido sobreposta. Observe-se que esta condição não se aplica a autorizações do tipo forte, que não podem ser sobrepostas.

Recapitulando, na forma de um exemplo, a autorização no papel *Auxiliar de Enfermagem*

$\langle \text{Profissional de Saúde}, \langle -, \text{AssinarLaudo}, \text{PEP}, \text{fraca} \rangle \rangle$

é válida se o papel *Profissional de Saúde* for ancestral do papel *Auxiliar de Enfermagem* (condição do item 1); e se não existir uma autorização forte equivalente num papel estritamente ascendente do papel *Profissional de Saúde* (condição do item 2); e se não houver uma autorização equivalente que a sobreponha num papel estritamente descendente de *Profissional de Saúde* e ascendente ao papel *Auxiliar de Enfermagem* (condição do item 3). A não satisfação de uma das condições revoga a autorização no papel *Auxiliar de Enfermagem*.

Por fim, o último quantificador universal no invariante do esquema *MACA_Hierarquico* estabelece que as autorizações revogadas (função *autorizacoes_revogadas*) de um papel são todas aquelas associadas, direta ou indiretamente, ao papel e que não são autorizações válidas. Pode-se constatar, nas propriedades do invariante, que autorizações do tipo forte não podem ser revogadas, isto é, são sempre válidas. Supondo a existência de uma autorização forte revogada associada (direta ou indiretamente) a um papel, então a condição do item 2 ou a condição do item 3 estaria violada. Caso a condição do item 2 fosse violada, então haveria pelo menos duas autorizações fortes numa mesma linha hierárquica, o que não é permitido. Caso a condição do item 3 fosse violada, então a autorização da hipótese deveria ser fraca. Logo, não pode haver autorizações fortes revogadas.

3.3.3 MACA restrito

O MACA Restrito estabelece relações de separação de responsabilidades que particionam de forma compulsória, por papéis mutuamente exclusivos, a responsabilidade e a autoridade para realizar uma ação em que há conflitos de interesses. As relações de separação de responsabilidades baseiam-se nos conflitos entre autorizações.

Para ilustrar, suponha-se os papéis *Médico Assistente* e *Médico Auditor* (*Figura 4*) num hospital. O médico assistente tem permissão para prescrever terapias

no PEP, enquanto médicos auditores têm permissão para glosar o pagamento de prescrições suspeitas ou sem cobertura, até que a questão seja meticulosamente examinada. Um usuário malicioso, detendo ambos os papéis, poderia prescrever, como médico assistente, procedimentos fraudulentos e, como médico auditor, considerá-los regulares, não os glosando. Para prevenir tal situação de conflitos de interesses, os seguintes pares de autorizações conflitantes podem ser definidos:

- $\langle \text{Médico Auditor}, \langle -, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$
 $\langle \text{Médico Assistente}, \langle +, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$
- $\langle \text{Médico Auditor}, \langle +, \text{GlosarPrescrição}, \text{PEP}, \text{forte} \rangle \rangle$
 $\langle \text{Médico Assistente}, \langle -, \text{GlosarPrescrição}, \text{PEP}, \text{forte} \rangle \rangle$

As autorizações em cada par conflitam porque, num papel, o acesso a uma operação de um objeto é concedido e, em outro papel, o mesmo acesso é proibido. Como o tipo das autorizações é forte, o conflito é denominado de *conflito forte*. Caso contrário, o conflito é denominado *conflito fraco*. Observa-se que, quando os tipos das autorizações são diferentes, não há conflitos, pois as autorizações fortes prevalecem sobre as autorizações fracas equivalentes.

A idéia é utilizar autorizações fortemente conflitantes para representar conflitos de interesses potenciais para um usuário exercendo os papéis onde tais autorizações estão presentes. A separação de responsabilidades, portanto, é estabelecida para proibir que um usuário execute as operações existentes nas autorizações fortemente conflitantes, evitando-se, assim, a ocorrência da situação de conflitos de interesses representada. As seguintes alternativas são consideradas para alcançar esse objetivo:

1. Impor restrições para evitar que papéis conflitantes tenham usuários associados em comum, de modo similar a SR estática do CABP Restrito (v. p. 34). *Papéis conflitantes* são aqueles que possuem autorizações conflitantes entre si. Quando há conflitos fortes, diz-se que os papéis são *fortemente conflitantes*. No exemplo anterior, os papéis *Médico Auditor* e *Médico Assistente* são fortemente conflitantes e não poderiam compartilhar usuários nessa alternativa. Logo, a situação de conflitos de interesses representada não se manifesta porque um mesmo usuário não pode exercer ambos os papéis;
2. Impor restrições para tornar indisponíveis as autorizações positivas fortemente conflitantes existentes em papéis (conflitantes) que estão associados a um mesmo usuário. Ou seja, um usuário pode vincular-se a

papéis fortemente conflitantes, porém, as autorizações positivas com conflitos fortes ficam indisponíveis para evitar a manifestação das situações com conflitos de interesses. No exemplo anterior, um usuário com os papéis *Médico Auditor* e *Médico Assistente* não teria as autorizações positivas fortemente conflitantes disponíveis, logo, estaria impedido de prescrever e de glosar prescrições.

O MACA adota a alternativa do item 2 porque alia a robustez da SRE com a flexibilidade da separação de responsabilidade dinâmica do CABP Restrito (v. p. 35). Da SRE, incorpora a impossibilidade de um mesmo usuário executar ações de autorizações associadas a papéis (fortemente) conflitantes, mas não às custas de impedir que o usuário execute todas as ações. Apenas as ações de autorizações positivas com conflitos fortes, que representam conflitos de interesses potenciais, ficam indisponíveis, restando as ações associadas às demais autorizações. Da SRD, incorpora a possibilidade de um mesmo usuário associar-se a dois ou mais papéis (fortemente) conflitantes, porém, de modo distinto da SRD, permite que tais papéis sejam ativados simultaneamente porque as autorizações (positivas) fortemente conflitantes ficam indisponíveis. A SRD do CABP Restrito, por sua vez, limita-se a tratar os casos de conflitos de interesses que ocorreriam se o usuário ativasse simultaneamente dois ou mais papéis conflitantes, mas que não se manifestariam se os papéis fossem ativados alternadamente.

Portanto, a separação de responsabilidades no MACA estabelece uma política estrita para resolução dos conflitos fortes existentes entre autorizações, quando associadas a um mesmo usuário por meio dos papéis a que está vinculado, conforme enunciado a seguir:

- Havendo duas ou mais autorizações com conflitos fortes, prevalece aquela autorização que *proíbe* o acesso.

Nos casos de ocorrência de conflitos fracos entre autorizações, a política adotada pelo MACA para resolvê-los é mais permissiva, de acordo com a seguinte definição:

- Havendo duas ou mais autorizações com conflitos fracos, prevalece aquela autorização que *concede* o acesso.

Esse dualismo se justifica porque os conflitos fracos não representam conflitos de interesses potenciais. Papéis que conflitam fracamente apenas definem as permissões e as proibições necessárias para um usuário desempenhar suas responsabilida-

des. Por exemplo, os pares de autorizações

- $\langle \text{Pesquisador Clínico}, \langle -, \text{VerDadosDemográficos}, \text{PEP}, \text{fraca} \rangle \rangle$
 $\langle \text{Médico Assistente}, \langle +, \text{VerDadosDemográficos}, \text{PEP}, \text{fraca} \rangle \rangle$
- $\langle \text{Pesquisador Clínico}, \langle +, \text{PesquisarObservaçõesClínicas}, \text{PEP}, \text{fraca} \rangle \rangle$
 $\langle \text{Médico Assistente}, \langle -, \text{PesquisarObservaçõesClínicas}, \text{PEP}, \text{fraca} \rangle \rangle$

conflitam fracamente, mas não sinalizam conflitos de interesses potenciais. Um usuário que atue apenas como pesquisador clínico está proibido de ver dados demográficos capazes de identificar o paciente, mas pode pesquisar observações clínicas (e.g., valor da taxa de colesterol total) para fins epidemiológicos. Já um usuário com ambos os papéis tem as prerrogativas, como médico assistente, de ver dados demográficos e, como pesquisador clínico, de pesquisar observações clínicas. Portanto, com a política de resolução de conflitos fracos, os usuários têm a permissão para realizar as ações que necessitam para cumprir as suas funções organizacionais. A razão de ser da política é atender o princípio da necessidade de saber/fazer (princípio do privilégio mínimo). Por outro lado, conflitos fortes representam conflitos de interesses potenciais e devem ser definidos criteriosamente, para tarefas muito sensíveis, que levem ao cometimento de fraudes ou erros. A razão de ser da política de resolução de conflitos fortes, portanto, é suportar a implementação de políticas de separação de responsabilidades.

Os conflitos entre autorizações vistos nos exemplos anteriores são denominados de *conflitos estáticos* porque podem ser determinados no momento em que as autorizações são definidas. Nos casos em que pelo menos uma das autorizações tem uma regra, é impraticável, em geral, determinar estaticamente se há conflitos. Tais conflitos somente podem ser determinados dinamicamente, no momento em que as regras são avaliadas, e são denominados de *conflitos dinâmicos*. Por exemplo, não é possível determinar estaticamente se as autorizações

```

<Pesquisador Clínico, <-, VerDadosDemográficos, PEP, fraca>>
<Médico Assistente, <exp-abs (umCodPac) {
    umCodPac in pacCtx.pacientes_internados
}, VerDadosDemográficos, PEP, fraca>>

```

conflitam. Apenas quando um usuário com os papéis *Pesquisador Clínico* e *Médico Assistente* associados tenta ver os dados demográficos de um paciente identificado por umCodPac é que se sabe se o par de autorizações conflita. Caso o paciente em questão esteja internado, a autorização será positiva e, neste instante, o conflito o-

corre. Como o conflito é fraco, o acesso é concedido, pois médicos assistentes necessitam ver dados demográficos de pacientes internados, conforme estabelece a política de acesso. No MACA, a separação de responsabilidades determinada em termos de conflitos estáticos é chamada de *separação de responsabilidades estática*, enquanto aquela definida em termos de conflitos dinâmicos é denominada de *separação de responsabilidades dinâmica*.

O esquema *MACA_SeparacaodeResponsabilidadeEstatica*, apresentado na subseção 3.3.3.1, especifica as restrições para SRE no MACA introduzidas informalmente. A SRD é formalizada na seção 3.3.4, que especifica o algoritmo de decisão de acesso, porque os conflitos dinâmicos, em geral, só podem ser determinados no momento em que as regras de autorização são avaliadas durante uma solicitação de permissão de acesso.

3.3.3.1 Separação de responsabilidade estática

A separação de responsabilidade estática é definida como uma relação entre autorizações conflitantes para impor restrições nas autorizações válidas nas sessões de um usuário.

<p><i>MACA_SeparacaodeResponsabilidadeEstatica</i></p> <hr/> <p><i>MACA_Hierarquico</i></p> <p><i>conflitos_estaticos</i>: $pa \leftrightarrow pa$</p> <p><i>sre</i>: $pa \leftrightarrow pa$</p> <hr/> <p>$\forall (b_1, b_2) : \text{conflitos_esticos} \mid b_1 = (p_1, a_1) \wedge b_2 = (p_2, a_2) \bullet$ $op(a_1) = op(a_2) \wedge ob(a_1) = ob(a_2) \wedge ta(a_1) = ta(a_2) \wedge \{tp(a_1), tp(a_2)\} = \{+, -\}$</p> <p>$sre = \{(b_1, b_2) \in \text{conflitos_esticos} \mid b_1 = (p_1, a_1) \wedge b_2 = (p_2, a_2) \wedge ta(a_1) = ta(a_2) = \text{forte}\}$</p> <p>$\forall s : \text{sessoes} \mid u = \text{usuario_da_sessao}(s) \bullet$ $\exists a : \text{autorizacoes_validas}(p) \mid p \in \text{papeis_da_sessao}(s) \wedge tp(a) = + \wedge ta(a) = \text{forte} \bullet$ $\exists (b_1, b_2) : sre \mid b_1 = (p_1, a_1) \wedge b_2 = (p_2, a_2) \wedge$ $u \in \text{usuarios_associados_hp}(p_1) \wedge u \in \text{usuarios_associados_hp}(p_2) \bullet$ $a = a_1 \vee a = a_2$</p>

A especificação no esquema anterior estabelece a SRE para o MACA Hierárquico. A relação *conflitos_estaticos* define conflitos estáticos como o relacionamento entre pares de autorizações associadas a papéis, presentes na relação *pa* definida no esquema *MACA_Básico* (v. p. 62), que satisfazem as seguintes propriedades, estabelecidas no primeiro quantificador universal do invariante do esquema:

1. Ambas as autorizações devem ser equivalentes, isto é, referirem-se a uma mesma operação e a um mesmo objeto;

2. Os tipos de ambas as autorizações devem ser iguais, pois não há conflitos entre autorizações fortes e fracas porque as autorizações fortes prevalecem sobre as autorizações fracas;
3. Os tipos de privilégios de ambas as autorizações não podem ser regras e, necessariamente, devem ser diferentes (um positivo e o outro negativo) para caracterizar um conflito estático.

Pode-se constatar que, para haver conflitos entre autorizações, elas devem estar associadas diretamente a papéis distintos, ou seja, duas autorizações conflitantes não podem coexistir associadas a um mesmo papel. Uma consequência das propriedades estabelecidas para as autorizações associadas a um papel no esquema *MACA_Básico* (v. segundo quantificador universal no invariante da p. 62), que impedem a existência de duas ou mais autorizações equivalentes associadas diretamente a um mesmo papel. Pode-se verificar, também que, decorrente do invariante do esquema *MACA_Hierárquico* (v. primeiro quantificador universal na p. 67), não pode haver autorizações com conflitos estáticos fortes em papéis de uma mesma linha hierárquica. Por exemplo, as autorizações fortes e conflitantes

$\langle \text{Médico Auditor}, \langle -, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$

$\langle \text{Médico}, \langle +, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$

abaixo não podem existir, considerando-se a hierarquia da *Figura 4*, porque uma autorização forte não pode ser revogada. Numa mesma linha de responsabilidades na hierarquia, não pode haver contradição nas ações permitidas (ou proibidas) de modo absoluto por meio de uma autorização forte.

A relação de separação de responsabilidades estática *sre* (esquema *MACA_SeparacaodeResponsabilidadeEstática*) é um subconjunto de *conflitos_estaticos* contendo apenas autorizações com conflitos fortes. Observa-se, portanto, que a SRE somente ocorre entre papéis presentes em linhas hierárquicas distintas. O restante do invariante conclui a definição da SRE estabelecendo que não pode existir uma autorização válida, positiva e forte, numa sessão de um usuário, que esteja em conflito forte com outra autorização associada a algum papel ao qual o usuário também está associado. Por exemplo, a autorização

$\langle +, \text{Prescrever}, \text{PEP}, \text{forte} \rangle$

somente é válida na sessão de um usuário se ele não estiver associado a papéis que possuam a autorização em conflito forte. Caso o usuário esteja associado, direta ou

indiretamente, aos papéis *Médico Assistente* e *Médico Auditor* e existam estas autorizações definidas

$\langle \text{Médico Auditor}, \langle -, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$

$\langle \text{Médico Assistente}, \langle +, \text{Prescrever}, \text{PEP}, \text{forte} \rangle \rangle$

então a autorização $\langle +, \text{Prescrever}, \text{PEP}, \text{forte} \rangle$ torna-se inválida em qualquer sessão do usuário.

Assim como na SRE do CABP Restrito, uma consequência da SRE no MACA, *havendo autorizações com conflitos fortes*, é a impossibilidade da existência superusuário. O superusuário é aquele que tem disponível para si todas as autorizações válidas (não revogadas) existentes. As duas hipóteses para existência de tal usuário são diretamente refutadas pelo esquema *MACA_SeparacaodeResponsabilidadeEstatica*, que inclui todas as declarações e invariantes dos esquemas do MACA:

- *Hipótese 1:* o superusuário é aquele que está associado a todos os papéis existentes a fim de adquirir todas as autorizações válidas associadas a aos papéis. Há pelo menos um par de autorizações com conflitos fortes.

Refutação: havendo pelo menos uma autorização positiva forte conflitante, tal autorização não pode ser válida em nenhuma sessão do usuário. Logo, a autorização fica indisponível e portanto, o superusuário não pode existir;

- *Hipótese 2:* o superusuário é aquele que está associado a um *papel raiz*, descendente de todos os outros papéis, que herda todas as autorizações válidas. Há pelo menos um par de autorizações com conflitos fortes.

Refutação: como o MACA só admite hierarquias em estrutura de árvore invertida e não pode haver autorizações fortes conflitantes numa mesma linha hierárquica, não pode existir tal papel raiz. Logo, também não pode existir um superusuário associado a esse papel.

∴

O esquema a seguir finaliza a especificação do espaço do estado do MACA com a introdução da função *papeis_para_ativar*, que mapeia um usuário para o conjunto dos papéis que ele pode ativar.

<p><i>MACA</i></p> <hr/> <p><i>MACA_Separacao de Responsabilidade Estatica</i></p> <p><i>papeis_para_ativar</i>: $usuarios \rightarrow \mathbb{P} \text{ papeis}$</p> <hr/> <p>$\forall u: \text{dom } \textit{papeis_para_ativar} \bullet$ $\textit{papeis_para_ativar}(u) =$ $\{p_i \in \textit{papeis} \mid u \in \textit{usuarios_associados}(p_i) \wedge$ $\quad \nexists p_j: \textit{papeis} \mid p_i \neq p_j \wedge u \in \textit{usuarios_associados}(p_j) \bullet p_i \in \textit{ascendentes}(p_j)\}$</p> <hr/> <p>$\forall s: \text{dom } \textit{papeis_da_sessao} \bullet \textit{papeis_da_sessao}(s) \subseteq \textit{papeis_para_ativar}(\textit{usuario_da_sessao}(s))$</p> <hr/>
--

O invariante do esquema estabelece que o conjunto de papéis disponíveis para ativação por um usuário é igual ao conjunto dos papéis associados diretamente ao usuário, desde que a seguinte condição seja observada:

- Para cada papel p_i disponível para ativação, não pode existir um outro papel p_j associado diretamente ao usuário, que seja descendente do papel p_i .

Ou seja, o conjunto de papéis disponíveis para ativação nem sempre corresponde ao conjunto de papéis associados ao usuário. Considere-se a hierarquia de papéis na *Figura 4* (v. p. 65) e suponha-se um usuário com o seguinte conjunto de papéis associados:

{Profissional de Saúde, Médico, Médico Assistente,
Pesquisador Clínico}

De acordo com o estabelecido no invariante do esquema *MACA*, os únicos papéis que podem ser ativados para esse usuário são os seguintes:

{Médico Assistente, Pesquisador Clínico}

Assim, quando um usuário tem mais de um papel de uma mesma linha hierárquica associado, somente o papel de maior responsabilidade pode ser ativado. O usuário do exemplo anterior possui papéis de duas linhas hierárquicas: $\langle \textit{Profissional de Saúde}, \textit{Médico}, \textit{Médico Assistente} \rangle$ e $\langle \textit{Profissional de Saúde}, \textit{Pesquisador Clínico} \rangle$. Logo, os papéis terminais (sublinhados) das duas linhas, os de maior responsabilidade, são os disponíveis para ativação. O motivo para tal restrição é que se um usuário está associado a um papel de maior responsabilidade, ele incorpora todas as autorizações dos papéis ancestrais, salvo as autorizações revogadas. Logo, a associação direta dos papéis ancestrais não aumenta o poder do usuário. Porém, se o usuário ativasse algum papel ancestral, ele poderia ter disponível uma autorização que foi revogada no papel de maior responsabilidade. Daí a proibição para o usuário ativar tais papéis.

Por fim, o último quantificador universal do invariante estabelece que os papéis ativos (função *papeis_da_sessao*) numa sessão de um usuário devem ser papéis que o usuário pode ativar.

3.3.4 Especificação do algoritmo de decisão de acesso

O algoritmo de decisão de acesso define um mecanismo para determinar se o usuário de uma sessão tem a permissão para acessar um objeto protegido a fim de executar uma operação específica. A especificação das propriedades do algoritmo no MACA traz significativas diferenças em relação a especificação do algoritmo de decisão de acesso no CABP do NIST (v. esquema *AcessoAutorizado* na p. 37) porque:

- Somente as autorizações válidas, de um papel que o usuário pode ativar, são consideradas para decidir o acesso. As autorizações revogadas e as autorizações positivas com conflitos estáticos fortes são ignoradas;
- As regras nas autorizações são avaliadas de acordo com o contexto existente no momento da tentativa de acesso, incluindo-se os argumentos fornecidos para autorizações parametrizadas;
- Considera a política para resolução de conflitos dinâmicos fortes existentes entre autorizações, no momento da tentativa de acesso, a fim de suportar a separação de responsabilidade dinâmica;
- Considera a prioridade das autorizações fortes sobre as autorizações fracas equivalentes;
- Ativa automaticamente os papéis disponíveis para o usuário, de acordo com a necessidade de acessar um objeto e uma operação particulares, seguindo um enfoque adotado originalmente por Obelheiro *et al.* (2001).

O esquema de operação *MACA_AcessoAutorizado*, definido a seguir, especifica as condições que devem ser satisfeitas imediatamente antes (pré-condição) e após (pós-condição) a execução da operação (algoritmo) de decisão de acesso. A pré-condição estabelece as propriedades que devem ser satisfeitas pelos parâmetros de entrada da operação e pelo estado do sistema para que a execução da operação seja bem sucedida, ou seja, para que o estado do sistema após a execução seja possível. Considerando que a pré-condição é válida, a pós-condição estabelece as propriedades que devem ser satisfeitas pelos parâmetros de saída da operação e pelo estado do sistema. Em princípio, pode haver diferentes algoritmos para decidir o acesso no MACA. Entretanto, todos devem satisfazer as condições especificadas no esquema *MACA_AcessoAutorizado*.

A primeira linha da parte declarativa do esquema indica que a operação atua sobre o estado *MACA* e o operador Δ estabelece que o estado global do sistema

pode ser alterado pela execução da operação, para refletir a ativação automática de papéis. Na notação Z, as variáveis que representam o estado do sistema após a execução da operação são decoradas com o sufixo ($'$), enquanto aquelas que representam o estado antes da execução da operação não possuem sufixo. As variáveis declaradas com o sufixo $!$ são parâmetros de entrada enquanto as declaradas com o sufixo $?$ são parâmetros de saída da operação.

<i>MACA_AcessoAutorizado</i>
$\Delta MACA$
<i>sessad</i> , <i>objeto</i> , <i>operac</i> ! : NOME
<i>parametros</i> ! : PARAMETRO
<i>resultado</i> ? : BOOLEAN
$sessad \in sessoes \wedge operac \in ops \wedge objeto \in obs$ $resultado? = (\exists a : autorizacoes_validas'(p) \mid$ $p \in papeis_para_ativar(u) \wedge u = usuario_da_sessao(sessad) \bullet$ $(op(a) = operac \wedge ob(a) = objeto \wedge avalia_autorizacao(a, ctxs', parametros!) = +) \wedge$ $(ta(a) = forte \Rightarrow (\exists a_i : autorizacoes_validas(p) \mid p_i \neq p \wedge p_i \in papeis_para_ativar(u) \bullet$ $op(a) = op(a_i) \wedge ob(a) = ob(a_i) \wedge ta(a_i) = forte \wedge$ $avalia_autorizacao(a, ctxs', parametros!) = -)) \wedge$ $(ta(a) = fraca \Rightarrow (\exists a_i : autorizacoes_validas(p) \mid p_i \neq p \wedge p_i \in papeis_para_ativar(u) \bullet$ $op(a) = op(a_i) \wedge ob(a) = ob(a_i) \wedge ta(a_i) = forte)) \wedge$ $(papeis_da_sessao' =$ $papeis_da_sessao \oplus \{sessad \mapsto (papeis_da_sessao(sessad) \cup \{p\})\}))$ $\neg resultado? \Rightarrow papeis_da_sessao' = papeis_da_sessao$ $usuarios' = usuarios \wedge papeis' = papeis \wedge obs' = obs \wedge ops' = ops \wedge sessoes' = sessoes \wedge$ $ctxs' = ctxs \wedge up' = up \wedge autorizacoes' = autorizacoes \wedge pa' = pa \wedge hp' = hp \wedge$ $usuario_da_sessao' = usuario_da_sessao$

A operação *MACA_AcessoAutorizado* recebe como parâmetros de entrada a sessão do usuário, os nomes do objeto e da operação protegidas e os parâmetros com informações do contexto da aplicação relevantes para decisão do acesso. Após a execução da operação, o parâmetro de saída *resultado?* especifica se a decisão foi pela concessão ou proibição do acesso.

Os parâmetros de entrada de *MACA_AcessoAutorizado* devem satisfazer a seguinte pré-condição: os identificadores da sessão, da operação e do objeto devem pertencer aos conjuntos de sessões ativas, de operações e de objetos cadastrados, respectivamente. Imediatamente depois da execução da operação, o parâmetro *resultado?* deve satisfazer a seguinte pós-condição: o resultado da operação é verdadeiro (acesso autorizado) se, e somente se, existir uma autorização *a*, válida para um papel *p*, onde *p* é um papel que o usuário pode ativar, tal que as seguintes condições sejam observadas:

- A operação e o objeto relativos à autorização a são os mesmos que o usuário pretende acessar, no caso, os parâmetros de entrada $operac\alpha d$ e $objeto d$, respectivamente;
- A avaliação de a é *positiva* no contexto existente, incluindo os parâmetros da autorização ($parametros$!);
- Se o tipo da autorização a é forte, então não pode existir uma outra autorização equivalente, forte e negativa, associada a algum papel, diferente de p , que o usuário possa ativar. A existência de tal autorização configura um conflito dinâmico forte e, nesse caso, o acesso deve ser negado, segundo a política de resolução de conflitos fortes da separação de responsabilidade dinâmica;
- Se o tipo da autorização a é fraca, então não pode existir uma outra autorização equivalente e forte, associada a algum papel, diferente de p , que o usuário possa ativar, porque uma autorização fraca não pode prevalecer sobre autorizações fortes equivalentes, sejam elas positivas ou negativas;
- O papel p passa a integrar o conjunto de papéis ativos associado a sessão $sessao d$ pela função $papeis_da_sessao'$, após a execução da operação. Ou seja, o papel é ativado automaticamente. O operador $\oplus$ de sobreposição relacional (v. definição no Anexo C, p. 180) estabelece o seguinte: a função $papeis_da_sessao'$ (após a execução da operação) é igual a função $papeis_da_sessao$ (antes da execução da operação), com exceção do valor mapeado para $sessao d$, que passa a ser o valor definido no conjunto (de fato, uma função com um único par) à direita do operador $\oplus$ no invariante.

Caso a autorização a não exista, o valor de $resultado?$ é falso e o acesso é negado. Conseqüentemente, nenhum papel é ativado e o valor da função $papeis_da_sessao$ não é alterado na execução da operação ($papeis_da_sessao' = papeis_da_sessao$). As demais variáveis do estado *MACA* não são alteradas pela execução da operação, independente do resultado obtido. A especificação da não alteração de valor não é explícita para todas as variáveis existentes no estado sempre que os seus valores puderem ser determinados pelos valores das variáveis explicitadas no esquema. Por exemplo, a função $usuarios_associados$ tem seus valores determinados pelos valores da relação up (v. esquema *MACA_Básico* na p. 62). Como o valor de up não é modifi-

cado pela operação ($up' = up$), então, de acordo com o invariante de *MACA_Básico*, o valor de *usuarios_associados* também permanece inalterado.

A função que avalia se uma autorização é positiva ou negativa, usada no esquema da operação *MACA_AcessoAutorizado*, é especificada a seguir:

$$\begin{array}{l}
 \text{avalia_autorizacao} : (TIPO_PRIVILEGIO \times NOME \times NOME \times TIPO_AUTORIZACAO) \times \\
 \quad \quad \quad \mathbb{P} CONTEXTO \times PARAMETRO \rightarrow \{+, -\} \\
 \hline
 \forall (a, ctxs, prms) : \text{dom avalia_autorizacao} \mid a = (tp, opr, obj, ta) \bullet \\
 \quad tp \in \{+, -\} \Rightarrow \text{avalia_autorizacao}(a, ctxs, prms) = tp \wedge \\
 \quad tp \in REGRA \Rightarrow \text{avalia_autorizacao}(a, ctxs, prms) \in \{+, -\}
 \end{array}$$

Caso o tipo do privilégio da autorização seja positivo, a função retorna o valor positivo, caso ele seja negativo, a função retorna o valor negativo. Quando o tipo do privilégio é uma regra, então a função retorna um valor que pode ser positivo ou negativo, dependendo da regra, do contexto e dos argumentos passados como parâmetro. Essa especificação é intencionalmente abstrata porque os detalhes da avaliação das regras de autorização não são importantes para especificar o algoritmo de decisão de acesso. Basta saber que o resultado da avaliação de uma autorização ou é positivo, ou é negativo, exclusive.

∴

O modelo de autorização contextual e o algoritmo de decisão de acesso especificados nesta seção definem rigorosamente os conceitos, as entidades e as propriedades essenciais do MACA. É a referência usada para orientar o desenvolvimento de implementações do MACA que preservem a sua integridade. A seção 4.2, no capítulo 4, descreve uma implementação do MACA para evidenciar a exeqüibilidade do modelo para uso industrial. As seções 3.4 e 3.5 ilustram, com exemplos, a capacidade do modelo em atender requisitos de políticas de acesso ao PEP e de políticas administrativas para modelos de controle de acesso baseados em papéis, respectivamente.

3.4 Aplicação do MACA a um PEP exemplo

Esta seção apresenta um exemplo didático para ilustrar a capacidade do MACA em satisfazer políticas de acesso para o PEP, que poderiam ser definidas num hospital com atendimentos em níveis secundário ou terciário. A subseção 3.4.1 relaciona os serviços oferecidos pelo PEP exemplo, onde a política de acesso estabelecida pelo hospital, definida na subseção 3.4.2, deve ser imposta. Em seguida, a subseção 3.4.3 descreve a interpretação dessa “política de acesso de alto nível” em termos de relacionamentos formais entre as entidades do MACA (v. *Figura 5* na p. 85) para especificar uma “política de acesso técnica” (v. seção 2.1.3 na p. 16).

3.4.1 Serviços do PEP exemplo

O prontuário eletrônico do paciente exemplificado funciona na intranet corporativa do hospital e pode ser acessado a partir de rede interna ou através da Internet. A lista a seguir descreve parcialmente os serviços oferecidos pelo PEP:

- *Serviço de Informações Cadastrais*: permite ver informações do cadastro do paciente em duas diferentes categorias: 1) dados capazes de identificar diretamente o paciente (*e. g.*, nome, filiação); e 2) dados demográficos anônimos (*e.*, *g.*, sexo, cor da pele). Ainda oferece serviços para manutenção (criar, alterar e excluir) das informações cadastrais;
- *Serviço de Prescrição Médica*: permite ver e criar prescrições médicas relativas a pacientes específicos;
- *Serviço de Auditoria Médica*: permite glosar o pagamento de solicitações e prescrições médicas consideradas suspeitas ou sem cobertura;
- *Serviço de Agendamento e Internação*: permite agendar consultas, exames e internações, bem como proceder à internação de pacientes;
- *Serviço de Pesquisa Clínica*: permite pesquisar prontuários de pacientes com base em observações clínicas, como valores medidos em exames (*e. g.*, taxa de colesterol total, percentual de estenose da artéria descendente anterior esquerda), diagnósticos e tipos de prescrições.

3.4.2 Requisitos da política de acesso

Considerando os princípios da preservação da privacidade do indivíduo, da liberação de informações clínicas identificadas em função da necessidade de cuidado ao paciente e dos benefícios para toda a sociedade da realização de estudos epidemiológicos, os responsáveis pela direção clínica do hospital deliberaram a seguinte política de acesso para o prontuário eletrônico do paciente:

- Somente usuários autenticados e devidamente autorizados podem acessar o PEP e seus respectivos serviços. As categorias de usuários autorizados a acessar o PEP são: profissionais de saúde que assistem diretamente o paciente, incluindo médicos e paramédicos; pesquisadores clínicos; administradores hospitalares, incluindo auxiliares e analistas de registro de saúde; e os próprios pacientes;
- Em geral, os profissionais de saúde e os administradores hospitalares podem acessar as informações cadastrais do paciente, incluindo os da-

dos de identificação e os dados demográficos anônimos. Porém, a manutenção de tais cadastros é permitida apenas a auxiliares e a analistas de registros de saúde: os auxiliares só podem alterar cadastros, enquanto os analistas, em adição, têm permissão para criá-los e excluí-los. É vedado a um pesquisador clínico ver os dados que identificam diretamente o paciente; no entanto, tem o direito de ver os dados demográficos anônimos. Já um paciente tem acesso aos próprios dados cadastrais, incluindo o direito de alterá-los, mas não pode criar ou excluir cadastros, incluindo o próprio, nem ver cadastros de terceiros;

- Médicos assistentes são os únicos usuários com a prerrogativa de criar prescrições médicas. Porém, médicos assistentes só podem prescrever para pacientes internados, com exames, consultas ou internações agendadas nos últimos 60 dias ou em atendimento na emergência. Já a visualização de prescrições é facultada para as seguintes categorias de usuários, desde que obedecidas as condições estabelecidas abaixo:
 - ◆ Médicos assistentes: têm acesso irrestrito para ver prescrições;
 - ◆ Paramédicos: só podem ver prescrições de pacientes internados ou em atendimento na emergência, desde que o acesso seja durante o seu turno de trabalho;
 - ◆ Analistas e Auxiliares de registros de saúde: não têm direito de ver prescrições;
 - ◆ Médicos Auditores: só podem ver as prescrições para os pacientes dos planos que eles representam;
 - ◆ Pesquisadores Clínicos: têm acesso irrestrito para ver prescrições;
 - ◆ Pacientes: podem ver exclusivamente as próprias prescrições;
- Médicos auditores são os únicos usuários com a prerrogativa de glosar o pagamento de prescrições médicas, desde que a prescrição esteja faturada para o plano que eles representam. Para reduzir as possibilidades de conflitos de interesses, médicos auditores estão incondicionalmente proibidos de efetuar prescrições médicas, mesmo na hipótese de atuarem como médico assistente. Por outro lado, médicos assistentes também estão incondicionalmente proibidos de glosar o pagamento de prescrições médicas, mesmo na hipótese de atuarem como médico au-

ditor. Ou seja, um usuário não pode atuar ao mesmo tempo com todos os privilégios de médico assistente e de médico auditor;

- Auxiliares e analistas de registros de saúde são os únicos usuários que podem agendar consultas, exames e internações para pacientes. Porém, cabe exclusivamente ao analista de registros de saúde os procedimentos formais para internar um paciente;
- A pesquisa aos prontuários de pacientes com base em observações clínicas somente pode ser realizada por pesquisadores clínicos. Para as demais categorias de usuários, o acesso é proibido.

3.4.3 Especificação da política de acesso no MACA

Em geral, para interpretação de políticas de acesso de alto nível em políticas técnicas formais do MACA, recomenda-se a execução das seguintes atividades:

- *Definição dos recursos protegidos*: nesta etapa, uma representação abstrata dos objetos a proteger é especificada. Recomenda-se que a equipe responsável pela política de acesso, em conjunto com a equipe encarregada pela aplicação (formada pelos objetos a proteger), definam os nomes dos objetos e respectivas operações. Para cada objeto e operação, deve-se especificar quais as informações, do contexto da aplicação, são relevantes para decisão de acesso, para que sejam utilizadas na definição de regras de autorização parametrizadas;
- *Definição dos papéis organizacionais*: análise das funções organizacionais exercidas pelos usuários e das necessidades estabelecidas nas políticas de acesso para definição dos papéis. As linhas de autoridade e responsabilidade de cada função, bem como os benefícios da herança de autorizações, determinam a disposição hierárquica dos papéis identificados;
- *Configuração das autorizações contextuais*: atribuição das autorizações para cada papel de acordo com a política de acesso especificada, levando em consideração a herança de autorizações. Três etapas constituem esta atividade: 1) identificar e configurar as autorizações estáticas; 2) identificar e implementar os contextos necessários à definição de regras; 3) identificar e configurar as autorizações contextuais com respectivas regras;

- *Definição da separação de responsabilidades:* análise da política de acesso para identificar possíveis situações de conflitos de interesses. A partir daí, identificam-se ou se definem as operações independentes/redundantes ou operacionais/de verificação (v., na p. 36, o último parágrafo da seção 2.4.1.1) requeridas para implementação da separação de responsabilidades. Em seguida, as autorizações fortes e conflitantes para as operações são definidas nos papéis onde os casos de conflitos de interesses podem ocorrer.

Cabe ressaltar que essas atividades não constituem um processo sistemático para especificação de políticas de acesso no MACA, tampouco estão completas ou estabelecem uma ordem de execução. Apenas relacionam as atividades consideradas importantes para interpretar as políticas de acesso do PEP exemplo em políticas de acesso formais do MACA, apresentadas na *Figura 5*. As subseções seguintes comentam a execução de cada uma das atividades, descrevendo a aplicação do MACA.

A investigação de processos para definir papéis, autorizações, restrições e hierarquias de papéis é crescente nos meios acadêmico e industrial (Coyne, 1996; Epstein & Sandhu, 1999; Epstein & Sandhu, 2001; Fernandez & Hawkins, 1997; Kern *et al.*, 2002; Kuhlmann *et al.*, 2003; Neumann & Strembeck, 2002; Roeckle *et al.*, 2000; Shin *et al.*, 2003) e reflete a importância de sistematizar a tarefa de interpretar políticas de acesso de alto nível em políticas técnicas formais do CABP em grandes organizações. No entanto, a investigação de tais processos está fora do escopo do presente trabalho.

3.4.3.1 Representação dos serviços do PEP exemplo

Os serviços disponíveis no PEP exemplo são representados numa estrutura de árvore invertida, conforme ilustrado na *Figura 5* (b). Do ponto de vista do MACA, cada nó da árvore descreve um objeto, com respectivas operações, onde uma política de acesso é imposta. A hierarquia representa um relacionamento de agregação (composição) entre os objetos. Por exemplo, o objeto *PEP* é composto pela agregação dos objetos *Serviço de Informações Cadastrais*, *Serviço de Prescrição Médica*, *Serviço de Auditoria Médica*, *Serviço de Agendamento e Internação*, *Serviço de Pesquisa Clínica*, e assim sucessivamente nos ramos da árvore. Os objetos componentes de um outro objeto somente podem ser alcançados se o acesso à esse objeto for concedido. Ou seja, o acesso aos objetos que compõem o *PEP* só é possível se o acesso ao *PEP* for permitido. Observa-se que a hierarquia de acesso não é suportada pelo MACA, mas sim

Figura 5 – Aplicação do MACA para um PEP exemplo. (a) hierarquia parcial de papéis dos usuários de um hospital em estrutura de árvore invertida. (b) Objetos do PEP representados parcialmente numa estrutura de árvore invertida, com a lista das respectivas operações associadas. (c) Autorizações contextuais associadas a cada papel.

pelo monitor de referência que consulte o serviço de autorização do MACA (v. seção 4.1.3) para impor o controle de acesso. No caso do PEP exemplo, a função do monitor de referência é executada explicitamente pela própria aplicação, que limita o acesso aos seus componentes de acordo com a política estabelecida no MACA.

Os objetos com a operação *executar* representam aplicações ou serviços do PEP que levam a outras aplicações ou serviços, que finalmente, oferecem funcionalidades finais. Por exemplo, a aplicação *PEP* viabiliza o acesso ao *Serviço de Prescrição Médica* que possibilita efetuar as operações *criar* ou *ler* no objeto *Prescrição*. Em geral, são nos objetos que oferecem funcionalidades finais que residem as informações, do contexto da aplicação, relevantes para decisão de acesso e usadas na definição de regras de autorização parametrizadas. A partir da análise das funcionalidades do PEP exemplo e de sua política de acesso, identificaram-se dois parâmetros a serem fornecidos pela aplicação durante solicitações de autorização de acesso:

- *umCodPac*: denota o código de identificação única de um paciente. É passado como parâmetro sempre que a operação de um objeto referir-se a um paciente específico. Por exemplo, a operação *ler* do objeto *Dados de Identificação do Paciente* lê os dados de um paciente particular. Portanto, sempre que uma solicitação para leitura de dados de identificação de um paciente é realizada, o respectivo código de identificação é passado como argumento. Assim, em toda autorização associada a operação *ler* do objeto *Dados de Identificação do Paciente*, pode-se usar *umCodPac* como parâmetro de uma regra (v. a primeira autorização do papel *Paciente* na *Figura 5 (c)*). Nos casos em que a autorização não tem regra, o parâmetro passado é descartado (v. a primeira autorização do papel *Pesquisador Clínico* na *Figura 5 (c)*). O parâmetro *umCodPac* é passado às autorizações associadas aos seguintes pares de operações e objetos: $\langle \textit{ler}, \textit{Dados de Identificação do Paciente} \rangle$, $\langle \textit{ler}, \textit{Dados Demográficos} \rangle$, $\langle \{\textit{alterar}, \textit{excluir}\}, \textit{Cadastro do Paciente} \rangle$, $\langle \{\textit{criar}, \textit{ler}\}, \textit{Prescrição} \rangle$ e $\langle \textit{glosar}, \textit{Pagamento Prescrição} \rangle$;
- *umCodPresc*: denota o código que identifica uma prescrição. É passado como parâmetro sempre que a operação de um objeto referir-se a uma prescrição em particular. Por exemplo, a operação *ler* do objeto *Prescrição* lê o conteúdo de uma prescrição específica. Portanto,

sempre que uma solicitação para leitura de prescrição é realizada, o respectivo código de identificação é passado como argumento. Assim, em toda autorização associada a operação *ler* do objeto *Prescrição*, pode-se usar `umCodPresc` como parâmetro de uma regra (v. a última autorização do papel *Paciente* na *Figura 5 (c)*). Nos casos em que a autorização não tem regra, o parâmetro passado é descartado. O parâmetro `umCodPresc` é passado às autorizações associadas aos seguintes pares de operações e objetos: $\langle \textit{ler}, \textit{Prescrição} \rangle$ e $\langle \textit{glosar}, \textit{Pagamento Prescrição} \rangle$;

Finalmente, cabe frisar que a representação do PEP na *Figura 5 (b)* é parcial e abstrata. O PEP exemplo poderia ter seus serviços implementados independentemente em plataformas distintas, mas, do ponto de vista do MACA, ser considerado uma única aplicação, onde uma política de acesso é imposta de modo unificado e coerente.

3.4.3.2 Hierarquia de papéis

A estruturação da hierarquia de papéis (*Figura 5 (a)*) para atender as políticas de acesso para o PEP exemplo considera dois aspectos básicos: 1) as linhas de autoridade e de responsabilidade das funções tradicionalmente existentes em estabelecimentos de saúde; e 2) o compartilhamento de autorizações comuns nos papéis de menor responsabilidade. Em relação ao primeiro aspecto, por exemplo, a autoridade e a responsabilidade crescentes da função de auxiliar de enfermagem para a função de enfermeiro determina a disposição na hierarquia do papel *Enfermeiro* como descendente do papel *Auxiliar de Enfermagem*, conforme ilustrado na *Figura 5 (a)*. A razão principal é que, em geral, um enfermeiro tem todos os poderes do auxiliar de enfermagem, mais os poderes que são exclusivos de sua função. Esse também é o caso dos papéis de auxiliar e analista de registro de saúde, onde também há uma linha de autoridade e responsabilidade comumente encontrada nos hospitais. Uma característica de tais papéis é que eles denotam funções que são efetivamente exercidas por profissionais de estabelecimentos de saúde e que sempre possuem usuários diretamente associados. Portanto, são considerados *papéis concretos*. Os papéis *Cirurgião*, *Clínico*, *Médico Auditor*, *Auxiliar de Enfermagem*, *Enfermeiro*, *Nutricionista*, *Pesquisador Clínico*, *Auxiliar de Registro de Saúde* e *Analista de Registro de Saúde* (*Figura 5 (a)*) são exemplos de papéis concretos.

O segundo aspecto considerado refere-se a criação de papéis com o objetivo de compartilhar autorizações comuns a vários papéis de maior responsabilidade. Ca-

so do papel *Profissional de Saúde*, ilustrado na hierarquia da *Figura 5 (a)*, que não denota uma função real existente em hospitais. Em geral, tais tipos de papéis não possuem usuários diretamente associados porque descrevem parcialmente uma função organizacional, sendo considerados *papéis abstratos*. A sua razão de ser é facilitar a gerência da atribuição de autorizações para papéis. Por exemplo, como a autorização para executar o serviço de prescrição médica é comum a todos os profissionais de saúde, ela é definida uma única vez no papel *Profissional de Saúde* (*Figura 5 (c)*) para ficar indiretamente definida nos seus descendentes. Os papéis *Usuário*, *Médico*, *Médico Assistente*, *Paramédico* e *Administrador* (*Figura 5 (a)*) são também exemplos de papéis abstratos.

Uma vez estabelecida a hierarquia, a atribuição das autorizações aos papéis deve beneficiar-se das autorizações positivas e negativas aliadas a possibilidade da revogação de autorizações no MACA. Quando o acesso é permitido (proibido) para a maioria dos papéis, então uma autorização positiva (negativa) deve ser definida num papel ascendente. Nos casos de exceção, se houver, a autorização herdada é revogada. Por exemplo, segundo a política de acesso do PEP exemplo, apenas os usuários de uma minoria dos papéis não podem ler irrestritamente os dados de identificação de pacientes. São os pesquisadores clínicos (proibidos de ler quaisquer dados de identificação) e dos pacientes (só podem ler os próprios dados). Logo, a autorização $\langle +, \text{ler, Dados de Identificação do Paciente, fraca} \rangle$ é associada ao papel *Usuário* (*Figura 5 (c)*) e, apenas nas situações de exceção, ela é revogada por autorizações negativas (ou com regras). É o caso das autorizações equivalentes a anterior definidas nos papéis *Pesquisador Clínico* e *Paciente* (*Figura 5 (c)*). A existência de autorizações negativas é importante para revogar autorizações que concedem o acesso e também para explicitar proibições, visto que, no MACA, o que não é explicitamente permitido, está proibido.

3.4.3.3 Autorizações contextuais

A *Figura 5 (c)* define formalmente no MACA a política de acesso de alto nível do PEP exemplo descrita na subseção 3.4.2. Usando apenas autorizações estáticas (positivas ou negativas), pôde-se expressar que usuários em geral, salvo os casos de exceção, estão autorizados a acessar o PEP e o serviço de informações cadastrais, mas apenas para ler os dados demográficos e de identificação do paciente. A manutenção dos cadastros é permitida para os usuários dos papéis *Auxiliar de Registro de Saúde* (alterar) e *Analista de Registro de Saúde* (criar, alterar e excluir), conforme estabelecem as respectivas autorizações associadas. Os auxiliares de registro de saúde também

podem acessar o serviço de agendamento e de internação, mas apenas para realizar agendamentos. Somente analistas de registro de saúde podem executar o sistema de internação. Já o acesso ao serviço de prescrição médica é facultado a todos os profissionais de saúde e também para os pacientes, para que possam acessar as próprias prescrições, como se observa nas autorizações dos papéis *Profissional de Saúde* e *Paciente* (Figura 5 (c)). Médicos assistentes e pesquisadores clínicos podem ler prescrições incondicionalmente. Porém, o pesquisador clínico está proibido de ler dados de identificação do paciente, mas é o único autorizado a executar o serviço de pesquisa clínica. Médicos em geral têm acesso ao serviço de auditoria médica.

A utilização exclusiva de autorizações positivas ou negativas, como descrito no parágrafo anterior, não é suficiente para atender os requisitos da política de acesso do PEP exemplo. É o que ocorre com as autorizações para criar e ler prescrições médicas. Por exemplo, usuários com o papel *Médico Auditor* só podem ler as prescrições dos pacientes dos planos de saúde que eles representam. Usando apenas autorizações positivas ou negativas, ou o médico auditor poderia ler as prescrições de todos os pacientes, ou ele seria proibido de ler qualquer prescrição. Portanto, é necessário dispor de autorizações mais exatas, flexíveis, que usem regras baseadas em informações contextuais disponíveis durante a tentativa de acesso para decidir pela concessão ou proibição da leitura de uma prescrição. As regras de autorização que estabelecem as condições para leitura de prescrições em cada papel da Figura 5 são descritas a seguir:

- *Médico Auditor*: a regra estabelece que o acesso é concedido para *ler Prescrição* somente quando o plano de saúde do paciente pertence ao conjunto de convênios para os quais o usuário corrente trabalha. `pacCtx.plano_saude` é uma função contextual que retorna a identificação do plano de saúde do paciente denotado por `umCodPac`. `usuárioCtx.convenios` denota o conjunto das identificações dos convênios para os quais o usuário trabalha;
- *Paramédico*: a regra estabelece que o acesso é concedido para *ler Prescrição* somente se o paciente estiver internado ou quando a solicitação de acesso for realizada de estações das salas de emergência e o usuário corrente estiver em seu turno de trabalho. `pacCtx.pacientes_internados` denota o conjunto de identificadores de pacientes internados. `netCtx.peer_dns` retorna endereço DNS da estação de trabalho onde a tentativa de acesso é

realizada. `netCtx.domínios_emergência` denota o conjunto contendo os endereços (IP ou DNS) das estações de trabalho das salas de emergência do hospital. A variável booleana `usuárioCtx.está_no_turno_de_trabalho` indica se o usuário corrente está em seu turno de trabalho;

- *Paciente*: a regra estabelece que o acesso é concedido para *ler Prescrição* somente quando a prescrição referir-se ao próprio usuário como paciente. A variável `usuárioCtx.codPac` retorna o código que identifica unicamente o usuário como paciente. Caso o usuário não seja paciente, retorna o valor nulo.

A criação de prescrições só pode ser feita por médicos assistentes e seus descendentes, de acordo com a seguinte autorização do papel *Médico Assistente* (Figura 5 (c)):

- A regra da autorização estabelece que o acesso é concedido para *criar Prescrição* somente se o paciente estiver internado, ou se teve consultas, exames ou internações agendadas nos últimos 60 dias, ou quando a solicitação de acesso for realizada de estações das salas de emergência. `pacientes_internados` denota o conjunto de identificadores de pacientes internados. `pacCtx.pacientes_agendados` é uma função contextual que retorna o conjunto de identificadores de pacientes com exames, consultas ou internações agendadas nos últimos n dias. `netCtx.peer_dns` retorna endereço DNS da estação de trabalho onde a tentativa de acesso é realizada. `netCtx.domínios_emergência` denota o conjunto contendo os endereços (IP ou DNS) das estações de trabalho das salas de emergência do hospital.

A autorização definindo as condições para glosar os pagamentos de prescrições está associada ao papel *Médico Auditor* (Figura 5 (c)) e sua descrição segue abaixo:

- A regra estabelece que o acesso é concedido para *glosar Pagamento Prescrição* somente se a prescrição estiver faturada para algum convênio onde o usuário corrente trabalha. A função `pacCtx.plano_saude_presc` retorna a identificação do convênio para o qual a prescrição `umCodPresc` do paciente identificado por `umCodPac` foi faturada. `usuárioCtx.convenios` denota o conjunto das identificações dos convênios para os quais o usuário trabalha.

Por fim, o restante das autorizações baseadas em regras associadas ao papel *Paciente* estabelecem que o usuário, como paciente, somente pode ler os próprios dados demográficos e de identificação, assim como alterar o próprio cadastro. A descrição da *Figura 5 (c)* conclui-se na próxima seção, com os comentários à representação da separação de responsabilidades da política do PEP exemplo, em termos de conflitos entre autorizações fortes do MACA.

3.4.3.4 Separação de responsabilidades

A política de acesso do PEP exemplo (v. seção 3.4.2, p. 82) explicita a possibilidade de conflitos de interesses entre médicos auditores e médicos assistentes. Proíbe que um usuário, atuando como médico auditor, audite as prescrições que realizou trabalhando como médico assistente. A separação de responsabilidades foi implementada definindo-se operações de verificação e operacionais cujo acesso não pode ser compartilhado por um mesmo usuário. As operações são mutuamente exclusivas para um mesmo usuário. Ou seja, quem audita prescrições não pode criá-las e vice-versa. A operação de verificação, *glosar* o pagamento de prescrição, é realizada pelo médico auditor e portanto, não pode ser realizada pelo médico assistente. A proibição para *glosar* pagamentos está definida na autorização forte e negativa associada ao papel *Médico Assistente* (*Figura 5 (c)*). Usuários no papel de *Médico Auditor*, por sua vez, têm a autorização forte de *glosar* o pagamento de prescrições faturadas para os planos de saúde que eles representam. Para um usuário exercendo ambos os papéis, *Médico Auditor* e *Médico Assistente*, haverá um conflito dinâmico forte sempre que ele tentar *glosar* o pagamento de uma prescrição faturada para um plano de saúde para o qual ele trabalha. Porém, a autorização forte negativa no papel *Médico Assistente* assegura que o usuário não tem poder para *glosar* nenhuma prescrição, de acordo com a política de resolução de conflitos dinâmicos fortes do MACA.

Por outro lado, a tarefa operacional, *criar* uma prescrição, é realizada pelo médico assistente, devendo ser verificada para fins de auditoria, com a operação de *glosar* o pagamento de prescrição. Logo, o médico auditor não pode criar prescrições. A autorização forte e negativa associada ao papel *Médico Auditor* (*Figura 5 (c)*), proíbe a criação de prescrições. A criação de prescrições é facultada aos usuários com o papel *Médico Assistente*, desde que satisfeitas condições da regra na autorização forte correspondente (*Figura 5 (c)*). Havendo um conflito dinâmico forte para um usuário com ambos os papéis, ele não poderá criar prescrições.

Em suma, obtém-se a separação de responsabilidades pela limitação do poder dos usuários que, eventualmente, detenham os papéis *Médico Assistente* e *Médi-*

co *Auditor*: ficam proibidos de criar e de glosar o pagamento de prescrições. Note-se que, se um usuário fosse alternativamente atribuído, ao longo do tempo, aos papéis *Médico Assistente* e *Médico Auditor*, ele poderia, ora como médico assistente, criar uma prescrição, ora como médico auditor, revisar o seu pagamento. Ou seja, a situação de conflitos de interesses poderia acontecer. Entretanto, esse caso poderia ser evitado alterando-se a regra da autorização para glosar o pagamento de prescrição da seguinte forma:

```

<exp-abs (umCodPac, umCodPresc) {
  pacCtx.plano_saude_presc(umCodPac, umCodPresc) in usuárioCtx.convenios
  &
  pacCtx.registro_profissional_presc(umCodPac, umCodPresc) !=
                                     usuárioCtx.registro_profissional
}, glosar, Pagamento Prescrição, forte),

```

Assim, para glosar uma prescrição, além do usuário precisar trabalhar para o plano de saúde faturado, ele não pode ser o profissional que criou a prescrição. Ainda que atue, num certo período, apenas como médico auditor, não poderá revisar o pagamento das prescrições que criou quando atuava exclusivamente como médico assistente. Ademais, o cometimento de fraude pela alternância dos papéis *Médico Assistente* e *Médico Auditor*, em geral, exigiria a participação de uma segunda pessoa, o administrador do MACA. Contudo, a separação de responsabilidades no MACA, assim como no CABP do NIST, busca prevenir a ocorrência de erros ou fraudes decorrentes da ação isolada de um indivíduo e não por conluíus.

3.5 O MACA e as políticas administrativas no CABP

No CABP, uma política administrativa define diretrizes de alto nível que determinam como o acesso às funções administrativas é controlado e como as decisões de permissão de acesso são estabelecidas. As funções administrativas possibilitam a criação e a manutenção (ler, alterar, excluir) das entidades (*e. g.*, usuários, papéis, autorizações, objetos e operações) e dos respectivos relacionamentos (*e. g.*, usuário-papel e papel-autorização) existentes em implementações do CABP. A *Figura 6 (b)* ilustra as funcionalidades administrativas básicas do MACA, onde uma variedade de políticas de acesso pode ser configurada. O objeto *MACA AD* representa a ferramenta administrativa do MACA que permite criar, ler, alterar ou excluir contas de usuários, papéis, autorizações, objetos e operações, bem como vincular (desvincular) usuários e autorizações para (de) papéis nas relações usuário-papel e papel-autorização, respectivamente. A operação *criarNovaSenha* atribui uma nova senha para um usuário que terá conhecimento dela sem a intermediação do administrador.

Figura 6 – Exemplo da uma política administrativa usando o MACA. (a) hierarquia dos papéis administrativos. (b) Representação das funções administrativas do MACA. (c) Autorizações contextuais associadas a cada papel.

3.5.1 Descentralização e autonomia

Os poderes disponíveis nas funções administrativas do *MACA AD* (Figura 6 (b)) podem acarretar abusos ou conflitos de interesses quando concentrados num único indivíduo. É necessário descentralizar a autoridade e a responsabilidade administrativas por múltiplos indivíduos para evitar a existência de superusuários. Ademais,

Figura 7 – Organograma parcial da estrutura hierárquica de um hospital. Existem quatro tipos de unidades organizacionais: 1) a própria organização; 2) institutos; 3) divisões; e 4) serviços. Todo serviço está subordinado a uma divisão, toda divisão está subordinada a um instituto e todo instituto está subordinado a organização, que é única.

exige-se também mais autonomia para os administradores a fim de viabilizar a gerência da política de acesso em grandes organizações, onde o número de papéis pode alcançar valores da ordem de centenas ou de milhares e o número de usuários pode variar de dezenas de milhares até centenas de milhares. Valores de tal monta exigem das organizações a desconcentração do poder para um considerável número de administradores, mas com a autonomia de cada um deles limitada. O desafio é descentralizar os detalhes operacionais da administração de modelos de CABP, mas sem perder o controle unificado das políticas administrativas. Segundo Sandhu *et al.* (1999), existe uma tensão entre o desejo de escalabilidade obtido com a descentralização e a manutenção de um controle firme das políticas.

Por exemplo, uma organização pode optar por centralizar a execução das funções administrativas para criação e manutenção de papéis, de objetos e operações, e de autorizações, assim como a vinculação (desvinculação) de autorizações para (de) papéis. Porém, as unidades constitutivas da organização (*e. g.*, o organograma na *Figura 7*) teriam a autonomia para criar e manter seus próprios usuários, podendo vinculá-los (desvinculá-los) aos (dos) papéis existentes. Ou seja, os administradores do Serviço de Hemodinâmica só teriam permissão para gerenciar as contas

dos usuários associados diretamente ao serviço, não podendo manipular contas de usuários de outras unidades. Por sua vez, um administrador de divisão teria autonomia para gerenciar contas de usuários associados diretamente à divisão, assim como, dos usuários associados aos serviços a ela subordinados. Do mesmo modo, um administrador de instituto poderia gerenciar as contas de qualquer usuário subordinado ao instituto. Já um administrador da organização poderia gerenciar as contas de qualquer usuário da organização. Tal política estabeleceria uma hierarquia de acesso, cuja abrangência é limitada pela estrutura organizacional.

A descentralização e a autonomia administrativas também podem ser entendidas para o gerenciamento das autorizações (autorizações, objetos, operações e respectivos relacionamentos), tomando por base a estrutura organizacional, como no exemplo anterior. Entretanto, como em geral os sistemas (objetos e operações) da organização (*e. g.*, os componentes do PEP) permeiam todas as unidades organizacionais, o mais razoável é distribuir o poder de gerência das autorizações para os responsáveis (proprietários) pelos sistemas. Isto é, os responsáveis pelo “Serviço de Prescrição Médica” só podem gerenciar as autorizações relacionadas e este serviço, incluindo a capacidade de vinculá-las a papéis.

De modo similar, a criação e a manutenção de papéis pode ser desconcentrada e autônoma, mas limitada pela estrutura hierárquica da organização. Por exemplo, um administrador de papéis do Serviço de Hemodinâmica poderia criar ou manter papéis, desde que fossem descendentes do papel *Hemodinamicista*.

Em suma, as organizações podem requerer desde políticas administrativas amplamente descentralizadas e autônomas, até políticas totalmente centralizadas. Porém, em geral, demandam políticas mistas, limitadas, descentralizando e dando autonomia para executar parte das funções administrativas, mas mantendo concentrados os poderes para execução de outras funções.

3.5.2 Execução de uma política administrativa usando o MACA

A execução de políticas administrativas descentralizadas e autônomas, com grande número de usuários, papéis e administradores, como observado na subseção anterior, justifica a utilização do CABP para controlar o acesso a implementações do CABP. O objetivo é beneficiar-se das vantagens do CABP (v. seção 2.4 na p. 20) utilizando papéis administrativos para controlar e distribuir a autoridade e a responsabilidade na realização das funções administrativas do CABP numa organização.

A *Figura 6* exemplifica a utilização do MACA para definir uma política administrativa para o próprio MACA num hospital, cujo organograma é o ilustrado na

Figura 7. Os papéis administrativos (*Figura 6 (a)*) para os usuários do MACA AD estão divididos em três categorias principais: *Administrador de Usuário*, *Administrador de Papel* e *Administrador de Autorização*. A política administrativa do hospital determina que a administração de papéis e de autorizações é centralizada nos papéis *Administrador de Papel* e *Administrador de Autorização*, respectivamente. Os administradores de papéis podem criar, ler, alterar ou excluir papéis sem restrições, conforme estabelecem respectivas autorizações na *Figura 6 (c)*. Já um administrador de autorizações pode criar, ler, alterar ou excluir autorizações, objetos e operações, e vincular (desvincular) autorizações a (de) papéis (v. autorizações associadas na *Figura 6 (c)*). Como se observa, a administração de papéis e de autorizações não depende da estrutura organizacional do hospital.

Porém, a política administrativa determina a distribuição do poder de gerenciar as contas de usuários por diferentes papéis e a sua descentralização de acordo com as unidades organizacionais do hospital. As políticas configuradas para cada um dos papéis na *Figura 6 (c)* são descritas a seguir:

- *Administrador de Usuário*: pode ler a conta e ver os papéis associados a qualquer usuário do hospital, independente de unidade organizacional onde está lotado;
- *Administrador de Help Desk*: único papel com o poder de definir uma nova senha para qualquer usuário do hospital. A função de *help desk* é exercida por funcionários terceirizados;
- *Administrador de Contas*: pode alterar ou excluir contas de usuários, desde que os usuários sejam lotados numa unidade organizacional igual ou subordinada a unidade organizacional onde o administrador está lotado. Porém, um administrador de contas está impedido de alterar ou excluir a própria conta (v. regra). O parâmetro *atributos*, na regra da autorização para alterar, denota a lista dos novos valores dos atributos a serem armazenados na conta do usuário. O parâmetro *login* traz a identificação única no MACA da conta a ser alterada ou excluída. A função *LotaçãoAbaixoOuIgualA*, do contexto *organogramaCtx*, retorna *true* se a unidade organizacional passada como seu segundo argumento for igual ou subordinada à unidade organizacional passada como primeiro argumento. A variável *lotação*, do contexto *usuárioCtx*, retorna a identificação da unidade organizacional onde usuário corrente (o administrador de

contas) é lotado. A função contextual `ValorAtributo`, do contexto `macaCtx`, recupera o valor de um atributo específico (segundo argumento - "lotação") da lista de valores de atributos (atributos) passados como primeiro argumento. Quando o login do usuário é passado como primeiro argumento, então a função `ValorAtributo` retorna o valor do atributo especificado que está armazenado na conta do usuário no MACA. Note-se, portanto, na regra da primeira autorização, que a alteração só é permitida se a lotação já existente na conta do usuário e a nova lotação, presente nos atributos a serem armazenados, são iguais ou subordinadas a lotação do administrador. Isto é, um administrador de contas do Instituto do Coração pode transferir a lotação de um usuário do Serviço de Hemodinâmica para a Divisão de Cirurgia Cardiovascular, mas não para o Serviço de Neuropediatria, por exemplo;

- *Criador de Contas*: pode criar a conta de um usuário ou desvincular seus papéis, desde que o usuário seja lotado numa unidade organizacional igual ou subordinada a unidade organizacional onde o administrador está lotado. Entretanto, está impedido de desvincular os próprios papéis. A última autorização proíbe que ele vincule papéis para usuários, ou seja, ele não pode conceder poderes;
- *Vinculador de Papéis*: pode vincular papéis para um usuário, desde que o usuário seja lotado numa unidade organizacional igual ou subordinada a unidade organizacional onde o administrador está lotado e a conjunção das seguintes condições seja satisfeita: 1) ele não pode atribuir papéis para si mesmo; 2) a unidade organizacional de lotação do administrador não pode ter um tipo igual ou subordinando ao tipo *divisão*; 3) se a unidade organizacional (lotação) do administrador for do tipo *instituto*, então o papel a ser atribuído deve ser igual ou descendente do papel *Administrador de Contas* ou deve ser igual ou descendente do papel *Usuário*, excluindo-se o papel *Médico Auditor* e seus descendentes (considerar a hierarquia de papéis organizacionais –não administrativos– da *Figura 5 (a)* na p. 85). O parâmetro `papel` na regra da primeira autorização denota o identificador do papel a ser vinculado ao usuário identificado pelo parâmetro `login`. A função `TipoLotaçãoAbaixoOuIgualA`, do contexto `organogramaCtx`, retorna `true` se o tipo de unidade

organizacional passado como seu segundo argumento for igual ou subordinado ao tipo passado como primeiro argumento. A função `TipoLotação`, do contexto `organogramaCtx`, retorna o tipo da unidade organizacional passada como parâmetro. A função `PapelAbaixoOuIgualA`, do contexto `macaCtx`, retorna `true` se o papel passado como seu segundo argumento for igual ou descendente do papel passado como primeiro argumento.

A política especificada na *Figura 6 (c)* descentraliza a gerência de contas de usuários porque permite que administradores setoriais (nos serviços ou divisões) realizem as funções administrativas operacionais nas contas de seus próprios usuários, porém de maneira limitada. Por exemplo, um criador de contas do Serviço de Hemodinâmica, pode criar, alterar ou excluir a conta de um usuário do serviço. Porém, uma função de maior responsabilidade, que atribui poderes mediante a vinculação de papéis, só é permitida para um vinculador de papéis do Instituto do Coração. Acresça-se que o vinculador de papéis pode conceder o poder de gerenciar contas para outros usuários do instituto pela atribuição de papéis descendentes (ou iguais) ao papel *Administrador de Contas*. No entanto, está proibido de atribuir os papéis *Administrador de Help Desk* ou *Médico Auditor*. Apenas vinculadores de papéis lotados diretamente no Hospital de Clínicas pode fazê-lo, e para qualquer usuário da organização. Nessa política, a unidade organizacional de lotação do administrador determina os usuários abrangidos por sua autoridade.

Independente da descentralização baseada numa estrutura organizacional, a política da *Figura 6 (c)* distribui o poder de gerenciar contas por diferentes papéis, com destaque à separação de responsabilidades existente entre os papéis *Criador de Contas* e *Vinculador de Papéis*. Há conflitos dinâmicos fortes nas autorizações para criar usuários e para vincular e desvincular papéis (v. autorizações associadas a esses papéis na *Figura 6 (c)*). Portanto, a política de resolução de conflitos do MACA estabelece a seguinte separação de responsabilidade: 1) quem cria contas e desvincula papéis de usuários (*Criador de Contas*) está terminante e incondicionalmente proibido de vincular papéis (i. e., conceder poderes); 2) quem pode vincular papéis (*Vinculador de Papéis*) está terminante e incondicionalmente proibido de criar contas e de desvincular papéis de usuários. A intenção é impedir que um mesmo indivíduo concentre os poderes para criar contas de usuários e para vincular e desvincular papéis. Consequentemente, evita-se que um administrador malicioso possa, de forma isolada, criar fraudulentamente a conta de um usuário “fantasma” para conceder-lhe poderes com a vinculação de papéis. Mesmo existindo um administrador vinculado

aos papéis *Criador de Contas* e *Vinculador de Papéis*, o MACA assegura que ele não poderá criar contas, vincular e desvincular papéis, evitando-se assim que ele cometa fraudes dessa forma.

Observa-se, na política ilustrada, que o poder do administrador depende do papel que ele exerce e da unidade organizacional onde ele está lotado: quanto mais específico o papel na hierarquia (*e. g.*, *Criador de Contas* e *Vinculador de Papéis*) e mais abrangente a unidade organizacional (*e. g.*, Hospital de Clínicas), maior será o poder. É importante notar que o uso das autorizações contextuais do MACA permitiu separar a abrangência da autoridade administrativa das unidades organizacionais sobre os usuários, dos papéis a que esses estão vinculados. Não se precisou criar, por exemplo, os papéis “Criador de Contas de Serviço” ou “Criador de Contas de Instituto”, tampouco o papel “Criador de Contas do Instituto do Coração”. Ou seja, a hierarquia de papéis e a estrutura organizacional são ortogonais: a modificação de um não implica na alteração do outro e vice-versa. Isto confere mais flexibilidade para definição e evolução de políticas administrativas porque uma mesma hierarquia de papéis pode ser utilizada em instituições com diferentes estruturas organizacionais.

3.6 Considerações finais

A concepção do MACA visou satisfazer as necessidades de controle de acesso para o prontuário eletrônico do paciente em grandes organizações de saúde. A capacidade de incluir regras relacionando informações contextuais em autorizações confere ao MACA flexibilidade e poder expressivo para estabelecer políticas de acesso para o PEP e políticas administrativas para o CABP que se adaptam à diversidade ambiental e cultural das organizações de saúde. Outra contribuição é a possibilidade dos sistemas de PEP utilizarem o CABP do MACA de forma transparente e não ostensiva para o usuário final. Em comparação a outros modelos de CABP, o MACA apresenta características inovadoras, como utilizar a distinção entre conflitos fortes e fracos para implementar políticas de separação de responsabilidades. O capítulo seguinte especifica a arquitetura de software para integração do MACA, descreve seus aspectos de implementação, de desempenho e apresenta resultados de sua aplicação no InCor – HC-FMUSP.

4

Arquitetura e implementação

“(...) the design of computing systems can only properly succeed if it is well grounded in theory, and that the important concepts in a theory can only emerge through protracted exposure to application.”

Robin Milner, 1986

Este capítulo define a arquitetura de software idealizada para integração do MACA, descreve seus aspectos de implementação e de desempenho e também discute os resultados de sua aplicação prática. A seção 4.1 especifica os componentes de software e de comunicação da arquitetura, com respectivos padrões, protocolos e APIs, discutindo características relevantes da integração. A seção 4.2 expõe detalhes da representação, no serviço de diretórios LDAP, dos dados persistentes do MACA e do desenvolvimento em Java das suas funcionalidades, incluindo a implementação de contextos. A seção 4.3 mostra e discute os resultados dos experimentos para avaliação de desempenho do MACA e apresenta e discute os resultados da sua aplicação prática no InCor – HC-FMUSP.

4.1 Arquitetura de software

A arquitetura ilustrada na *Figura 8* especifica os componentes de software e de comunicação para suportar a implementação do MACA e sua relação com o prontuário eletrônico de pacientes num âmbito de uma organização de saúde. É uma arquitetura cliente-servidor multicamada com os seguintes componentes principais: um servidor LDAP, encarregado de manter a base de gerência de informações de segurança (BIGS); um servidor de segurança encarregado de oferecer os serviços de autenticação de usuário e de decisão de acesso à recursos, dentre outros; e finalmente as

Figura 8 – Esquema da arquitetura de software para integração do MACA.

aplicações cliente do PEP que requisitam tais serviços. Buscou-se adotar padrões de processamento aberto e distribuído a fim de alcançar maior interoperabilidade e mais portabilidade dos serviços de segurança. As subseções seguintes descrevem o papel de cada um destes componentes, destacando suas relações com o MACA.

4.1.1 Prontuário eletrônico de pacientes

Um prontuário eletrônico de pacientes efetivo é, em geral, um sistema distribuído composto por aplicações heterogêneas (Smith & Eloff, 1999). Na arquitetura proposta (Figura 8), os serviços padrão CORBA para área de saúde (CORBA Healthcare)—PIDS (*Person Identification Service*) (Object Management Group, 2001d), COAS (*Clinical Observation Access Service*) (Object Management Group, 2001b) e CIAS (*Clinical Image Access Service*) (Object Management Group, 2001a)—são a infraestrutura de *middleware* usada para integração das informações do prontuário eletrônico existentes em fontes heterogêneas ou legadas. O PIDS permite localizar identificadores de pessoas e registros associados, incluindo dados demográficos, por múltiplos bancos de dados e em diferentes instituições. O COAS oferece um mecanismo para recuperar observações clínicas a partir de repositórios de informação ou aplicações. O CIAS permite a obtenção de imagens médicas de um paciente com base nas características da imagem, tais como, parte do corpo, equipamento e modalidade.

A infra-estrutura do CORBA *Healthcare* é usada como base para integrar outros componentes do PEP, como os visualizadores de imagem, os serviços de prescrição médica, de gerência de laudos, de agendamento. Além de permitir a integração do PEP numa organização, também pode ser usada para integrar o PEP entre diferentes organizações de saúde. Entretanto, os serviços do CORBA *Healthcare* são insuficientes para a integração efetiva do PEP. Excetuando-se o PIDS, os demais serviços apenas oferecem facilidades para leitura de informações. Conseqüentemente, serviços CORBA dedicados podem ser empregados para integrar funcionalidades que permitam a cooperação dos componentes do PEP. Tais componentes podem ser novos sistemas desenvolvidos ou servidores de aplicação (*e. g.*, CORBA, *Enterprise JavaBeans* (Adatia *et al.*, 2001) ou *Web Services* (Newcomer, 2002)) encapsulando (*wrapping*) funcionalidades disponíveis em sistemas legados. Como se nota, os componentes do PEP podem ser aplicações, possivelmente legadas, que não utilizam o *middleware* do CORBA *Healthcare*. A despeito disso, é desejável que os componentes do PEP sejam capazes de impor uma política unificada e coerente para controle de acesso e autenticação de usuários.

A arquitetura (*Figura 8*) considera o prontuário eletrônico do paciente interno ao perímetro da *intranet*⁸ da organização de saúde que o mantém sob guarda. A *intranet* protege o PEP limitando os acessos externos apenas para terceiros confiáveis através de conexões seguras. Contudo, tanto os acessos externos, quanto os acessos internos, são regulados pela política de acesso armazenada na BIGS da instituição.

4.1.2 Base de informações de gerência de segurança

Os dados persistentes do MACA, definindo uma política de acesso para o PEP, são mantidos na BIGS (*Figura 8*), que armazena os papéis e sua hierarquia, as autorizações contextuais, a representação das operações e dos objetos do PEP, as contas de usuários, com dados para autenticação, e os relacionamentos usuário-papel e papel-autorização. A BIGS é um serviço de diretórios hierarquizado, cujo acesso e esquemas de descrição de dados são padronizados pelo protocolo LDAP (Yeong *et al.*, 1995). Por ser hierárquico e flexível, é simples representar no LDAP a hierarquia de papéis do MACA. As informações sobre usuários, papéis, autorizações, operações, objetos e seus relacionamentos, são descritas nos esquemas de dados padronizados (v. seção 4.2.1.2, p. 112).

⁸ Rede corporativa, de uso privado, que se utiliza da tecnologia da Internet para transferência de dados.

Todo acesso à BIGS é realizado através do protocolo LDAP sobre TLS (*Transport Layer Security*) (Dierks & Allen, 1999) para assegurar confidencialidade e integridade na comunicação. O TLS também pode assegurar autenticação mútua entre os clientes e o servidor LDAP. Ademais, a autenticação em sistemas operacionais e gerenciadores de bancos de dados mais comuns pode ser efetuada por intermédio do servidor LDAP usando o protocolo SASL (*Simple Authentication and Security Layer*) (Myers, 1997), que fortalece a segurança da autenticação em redes inseguras.

Com o servidor LDAP, pode-se armazenar, num único local, os dados de contas de usuários do PEP, do MACA, de sistemas operacionais e de gerenciadores de banco de dados, de modo que um usuário consiga se autenticar em cada um desses ambientes usando os mesmos login e senha. No entanto, os ambientes devem ser configurados para autenticação usando o protocolo LDAP. Nos casos de sistemas legados ou de retaguarda, que não interoperam com o LDAP, é possível utilizar a BIGS para implementar um serviço de login unitário, que armazena e recupera com segurança dados (legados) para autenticação (*e. g.*, login e senha) (Hartman *et al.*, 2001). Assim, evita-se armazenar informações sensíveis, como login e senha, no código das aplicações ou em arquivos de configuração, minimizando ameaças à segurança. Como o serviço também mapeia as contas no LDAP para logins e senhas existentes nesses sistemas, viabiliza-se a autenticação unificada de usuários numa organização, independentemente de plataforma ou de aplicação, uma recomendação para segurança de acesso do PEP, segundo a National Academy of Sciences (1997). Essa solução ainda facilita a gerência das contas, incluindo os procedimentos para admissão (demissão) dos usuários de uma organização, que requer a liberação (bloqueio) imediata do acesso a múltiplas plataformas e aplicativos.

O módulo *Ferramentas Administrativas* (Figura 8) é utilizado por administradores para gerenciar a política de acesso do PEP armazenada na BIGS. Os administradores têm o acesso controlado de acordo com a política administrativa estabelecida pela organização. A autenticação deles requer um método robusto de identificação, baseado em biometria ou em certificados digitais com cartões inteligentes. Em adição, a BIGS deve ser fisicamente protegida, com todos os acessos não locais proibidos. As exceções são para as conexões LDAP seguras (TLS), provenientes de domínios ou endereços IP preestabelecidos.

A unificação de políticas de segurança na BIGS facilita a administração, mas introduz uma sobrecarga adicional, visto que todos os clientes LDAP usam um mesmo servidor. Ademais, um servidor central introduz um único ponto de falha e abre oportunidades para ataques visando colocá-lo fora de serviço. Como LDAP é um ser-

viço distribuído e as implementações disponíveis têm mecanismo de réplica automático, é viável construir uma BIGS logicamente centralizada e mais tolerante a falhas, embora redundante e fisicamente distribuída por múltiplos servidores.

4.1.3 Servidor de segurança

O servidor de segurança (*Figura 8*) integra o serviço de segurança da arquitetura CORBA (SSC) (Object Management Group, 2001c), o serviço de decisão para acesso a recursos (RAD – *Facility*) (Object Management Group, 2000b) e o serviço de autorização do MACA. O SSC oferece interfaces padronizadas para autenticação de usuários, gerência de sessões e outros serviços de segurança, tais como, auditoria, controle de acesso, confidencialidade e não repudição. O RAD – *Facility* especifica um serviço padrão de decisão de acesso a recursos capaz de suportar diferentes políticas de acesso. O serviço do MACA implementa o modelo de autorização contextual para o CABP proposto neste trabalho, que decide o acesso aos recursos do PEP com base na política armazenada na BIGS e nos contextos disponíveis no momento do acesso, que podem ser locais ou remotos.

Os padrões SSC e o RAD – *Facility* embutem o serviço de autorização do MACA e outros serviços de segurança, como o de autenticação, para permitir que os componentes do PEP os utilizem independentemente de plataforma de software e de linguagem de programação. Porém, há diferenças na finalidade e na forma como o SSC e RAD – *Facility* integram o serviço de autorização do MACA nessa arquitetura.

4.1.3.1 O serviço de autorização do MACA e o controle de acesso no SSC

O SSC foi projetado de modo que os desenvolvedores de software tenham que programar um mínimo de código para tornar uma aplicação CORBA segura. A confidencialidade na interação entre os objetos CORBA, o controle de acesso e a auditoria, ao nível das interfaces dos objetos, são impostas pelo ambiente de segurança CORBA externo à aplicação, a fim de que essa não seja “ciente” da segurança. Para tanto, o SSC deve ser apropriadamente configurado por meio de interfaces padrão, voltadas à administração da política segurança.

Entretanto, o serviço de autenticação requer, das aplicações clientes, a programação de um *User Sponsor* (*Figura 9*), que provê os meios para o usuário entrar com seu login, dados de autenticação (*e. g.*, senha pessoal, cartão) e, opcionalmente, requisições para atributos de privilégio, tal como, um papel inicial para ativar. A autenticação é realizada por uma instância da interface *PrincipalAuthenticator* (*Figura 9*) do SSC, que verifica a validade da identidade do

Figura 9 – Caso de uso para autenticação do usuário em CORBA (Object Management Group, 2001c).

usuário no servidor LDAP. Caso seja bem sucedida, a autenticação resulta na criação de uma instância da interface *Credentials* (Figura 9), que permite requisitar, ao serviço de autorização do MACA (Figura 8), a execução de tarefas relativas a sessão do usuário. *Current* representa o usuário corrente responsável pela invocação de uma operação. Permite acessar, tanto no cliente, quanto no objeto CORBA servidor, a instância da interface *Credentials* do usuário corrente.

A integração do serviço de autorização do MACA ao SSC, portanto, requer implementações específicas das interfaces *PrincipalAuthenticator* e *Credentials*. Sempre que o *PrincipalAuthenticator* cria uma credencial, ele solicita que serviço de autorização do MACA inicie uma sessão correspondente para o usuário recém-autenticado. Assim, para cada sessão existente no serviço de autorização do MACA, há uma única credencial associada no SSC. É por meio da sessão associada que uma instância da interface *Credentials* obtém os papéis ativos e outras informações sobre o usuário, incluindo os papéis disponíveis para ativação. Ela também possibilita a ativação explícita de um papel, embora tal funcionalidade não seja necessária porque o MACA ativa os papéis automaticamente.

Com o usuário autenticado e a sua credencial criada, o controle de acesso é automaticamente imposto pelo ORB (*Object Request Broker*) às chamadas de operações das interfaces dos objetos CORBA servidores (Figura 10). Tal controle pode ser imposto no lado do cliente que invoca a operação, no lado do objeto servidor, ou em ambos os lados, a depender de configurações apropriadas. Ou seja, o ORB atua como o monitor de referência que impõe obrigatoriamente o controle de acesso. Ele usa uma implementação da interface *AccessDecision* do SSC para obter, do serviço de autorização do MACA, os privilégios de acesso conferidos pelos papéis do usuário.

Figura 10 – Modelo de controle de acesso do SSC (Object Management Group, 2001c).

O controle de acesso também pode ser explicitamente imposto, pelo cliente ou pelo servidor, mediante chamadas diretas a operação para decisão de acesso da interface *AccessDecision*.

Cada operação controlada pelo SSC tem associado um conjunto global de direitos de acesso requeridos e um combinador, que pode ser *any* ou *all* (Object Management Group, 2001c). Quando um usuário tenta executar uma operação, ele deve possuir pelo menos um (caso o combinador seja *any*) ou todos (caso o combinador seja *all*) os direitos requeridos para que o acesso seja concedido. Por exemplo, considere-se uma operação com os direitos (requeridos) *consulta* e *modificação* associados, e o combinador *all*. Para invocar esta operação, o usuário deve possuir uma autorização positiva para todos (*all*) os direitos de acesso estabelecidos.

No CORBA, combinadores e direitos de acesso não são associados diretamente a operações ou a objetos, mas sim a domínios de objetos. Um domínio agrupa objetos que estão sujeitos a uma política de segurança comum (Object Management Group, 2001c). Beznosov, (2000); Beznosov & Deng (1999) demonstraram que o controle de acesso baseado em papéis pode ser integrado ao serviço de segurança CORBA. Segundo os autores, as autorizações de um papel no CABP também não são associadas diretamente a operações e objetos CORBA, mas sim a direitos de acesso de um domínio de objetos. Por exemplo, o papel *Administrador* (Figura 4, p. 65) poderia ter o privilégio *consulta* associado ao domínio dos objetos do serviço PIDS, de modo que apenas as operações que consultam têm o acesso liberado. Aquelas que alteram o estado do sistema não poderiam ser executadas, em geral, pelos administradores. Observa-se que privilégios de acesso e domínios de objetos no CORBA são mapeados para operações e objetos de autorizações no CABP. Esse enfoque viabiliza a adminis-

tração e a imposição de uma política de controle de acesso a uma multitude de objetos CORBA em larga escala. Sempre que um novo objeto CORBA é criado, basta incluí-lo num domínio com direitos e combinador apropriados para submetê-lo a uma política de acesso. Nenhuma configuração adicional é exigida nas autorizações do MACA armazenadas na BIGS. Porém, o SSC deve manter os domínios com respectivos direitos de acesso requeridos, combinadores e objetos associados.

O controle de acesso imposto pelo *middleware* do ORB é confiável porque não depende da disciplina do desenvolvedor da aplicação (objeto CORBA servidor) em programar código de segurança. Ele oferece uma solução para o controle de acesso com granularidade grossa nas operações de objetos. Tem a desvantagem de não permitir que informações do contexto da aplicação sejam usadas como critério de decisão de acesso. Consequentemente, as autorizações contextuais parametrizadas, oferecidas pelo serviço de autorização do MACA, não podem ser usadas. Por outro lado, recursos específicos do PEP, acessados via operações de objetos CORBA, demandam um controle de acesso com baixa granularidade, inviável de ser imposto pelo ORB. Beznosov, (2000); Beznosov & Deng (1999); Hartman *et al.* (2001) recomendam, em casos como esse, usar o RAD – *Facility* para complementar o controle de acesso efetuado pelo ORB.

4.1.3.2 O serviço de autorização do MACA e o RAD – *Facility*

O RAD – *Facility* suporta um controle de acesso detalhado, ao nível da aplicação, mas isolando a lógica e as idiosincrasias da decisão de acesso das operações clientes nas aplicações. É usado por aplicações que são “cientes” da segurança, isto é, que têm a responsabilidade de impor explicitamente o controle de acesso. O RAD – *Facility* permite a combinação de múltiplas políticas de acesso, de acordo com o recurso a ser protegido. A interface do objeto *AccessDecision* (Figura 11) do RAD – *Facility* padroniza o modo como as aplicações clientes (objeto alvo) solicitam autorizações para um usuário acessar um recurso. As interações entre o cliente, o objeto CORBA servidor e o serviço de decisão do RAD – *Facility* dá-se da seguinte forma (Figura 11):

1. Um cliente solicita que o objeto servidor (objeto alvo) a execute de uma operação, por exemplo, prescrever uma medicação para um paciente;
2. Durante a execução da operação, o objeto servidor solicita ao objeto *AccessDecision* uma autorização para prescrever a medicação indicada para um paciente específico. Os identificadores da medicação e do paciente são passados como parâmetros da autorização, que pode utilizá-los para decidir o acesso;

Figura 11 – Interações entre cliente, objeto servidor (alvo) e o RAD *Facility* (Object Management Group, 2000b).

3. O objeto *AccessDecision* retorna um valor booleano indicando se a autorização foi concedida ou proibida;
4. Com base no retorno de *AccessDecision*, o objeto servidor responde à aplicação cliente, completando a execução da operação solicitada, se o acesso foi autorizado, ou então, interrompendo a execução e sinalizando para o cliente que o acesso foi proibido.

Na arquitetura da Figura 8, o RAD – *Facility* usa o serviço de autorização do MACA como a política de acesso para proteger os recursos do PEP. Portanto, o serviço de autorização do MACA é usado pelo *middleware* do ORB e pelo RAD – *Facility*. Para os componentes do PEP implementados como objetos servidores CORBA, o controle de acesso com granularidade grossa é imposto automaticamente pelo *middleware* do ORB. O RAD – *Facility* é usado para o controle de acesso detalhado, de responsabilidade dos objetos servidores, considerando o contexto da aplicação. Já os componentes legados do PEP, não acessados por meio de objetos CORBA, podem usar os serviços de autenticação do SSC e de autorização do RAD – *Facility* de forma independente. Entretanto, tais aplicações devem interoperar com a arquitetura CORBA para utilizar os serviços.

∴

Todas as interações dos clientes com o servidor de segurança devem acontecer por meio do protocolo IIOP (*Internet Inter-ORB Protocol*) (Object Management Group, 2002a) sobre TLS, para assegurar a confidencialidade e a autenticação mútua das partes. Embora logicamente centralizado, o servidor de segurança da arquitetura (*Figura 8*) pode ser configurado para atuar fisicamente distribuído e redundante, com maior tolerância a falhas. A arquitetura CORBA especifica um mecanismo para essa finalidade (Object Management Group, 2002b). Também existem produtos de software que implementam a arquitetura CORBA com suporte a mecanismos de tolerância a falhas, embora com soluções proprietárias, o caso do Visibroker (Borland Software Corporation, 2003), um ORB para a linguagem de programação Java.

4.2 Implementação

Esta seção descreve a implementação dos principais componentes da arquitetura concebida para integração do MACA. A seção 4.2.1 apresenta os detalhes da representação dos dados persistentes do MACA no serviço de diretórios LDAP. A seção 4.2.2 expõe os aspectos mais relevantes da implementação do servidor de segurança e, por fim, a seção 4.2.3 ilustra as funcionalidades disponibilizadas nas ferramentas administrativas implementadas para o MACA.

4.2.1 Representação do MACA no LDAP

As informações persistentes do MACA foram representadas na BIGS com esquemas do protocolo LDAP versão 3. A subseção 4.2.1.1 introduz sucintamente o padrão LDAP e a representação do modelo é descrita na subseção 4.2.1.2. O modelo de dados que representa o MACA foi implementado no produto de código fonte aberto *OpenLDAP* (OpenLDAP Foundation, 2003) versão 2.1.17, para funcionar em sistemas operacionais *Linux* (Linux Online, 2003), também um produto de código fonte aberto. As instruções de instalação e configuração do OpenLDAP para suportar o MACA, com suporte ao protocolo LDAP sobre TLS, estão no manual “MACA: guia de instalação e configuração” (v. item Manuais na p. 195).

4.2.1.1 Introdução ao LDAP

O LDAP é um padrão aberto para serviços distribuídos de diretórios na Internet. Foi originalmente especificado para acessar, sobre TCP/IP (*Transport Control Protocol/Internetwork Protocol*), o serviço padrão de diretórios distribuídos X.500. Isso porque o protocolo de acesso para o diretório X.500, o DAP (*Directory Access*

Figura 12 – Serviço de diretórios distribuídos LDAP.

Protocol), usa métodos de codificação complexos e requer o uso da pilha de protocolo OSI (*Open Systems Interconnection*), que não obteve amplo uso no mercado de redes; enquanto o LDAP pode ser implementado sobre o TCP/IP, de forma simples, eficiente e funcional. Como o modelo de dados do diretório X.500 é complexo, a *Internet Engineering Task Force* (IETF) decidiu adotar o LDAP, especificando-o, não apenas como um protocolo de acesso, mas também como um serviço de diretórios.

O serviço de diretórios LDAP (*Figura 12*) consiste em um conjunto de servidores de diretórios interconectados que se comunicam por meio do protocolo LDAP para atender requisições de clientes —*e. g.*, pesquisa, comparação, inserção, modificação e remoção. Se o servidor contactado não conseguir responder à requisição diretamente, ele tenta obter uma resposta de outros servidores para encaminhá-las ao solicitante. Assim, os servidores suportam um serviço de diretório distribuído, mas com transparência de localização para os clientes.

O diretório organiza as informações numa estrutura hierarquizada, denominada de *Árvore de Informações de Diretório* (AID), numa versão simplificada do modelo de dados do X.500 (*Figura 13*). Cada nó da AID corresponde a um registro armazenado e pode distribuir-se fisicamente em repositórios de vários servidores, mas mantendo sua unidade lógica. A estrutura de dados de uma entrada é determinada pelas *classes de objetos* a que ela pertence. Uma classe de objetos no LDAP especifica os tipos de atributos de uma entrada que pertence à classe. Tipos de atributos podem ser obrigatórios ou opcionais. Uma entrada pode ser definida por mais de uma classe de objetos, sendo obrigatória a classe padrão `top`. Por exemplo, as

Figura 13 – Exemplo parcial da árvore de informações de diretório do InCor – HC-FMUSP.

entradas abaixo do nó `ou=people` (Figura 13) especificam as contas de pessoas (usuários) cadastradas na AID e são definidas pelas classes padrão `person` e `organizationalPerson`, ilustradas abaixo:

```
objectclass ( 2.5.6.6
  NAME 'person'
  SUP top
  MUST ( sn $ cn )
  MAY ( userPassword $ telephoneNumber $ seeAlso $ description )
)

objectclass ( 2.5.6.7
  NAME 'organizationalPerson'
  SUP person
  MAY ( title $ x121Address $ registeredAddress $ destinationIndicator $
    preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
    telephoneNumber $ internationalISDNNumber $ facsimileTelephoneNumber $
    street $ postOfficeBox $ postalCode $
    postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l )
)
```

Toda definição de classe no LDAP é iniciada com um OID (*Object Identifier*), que especifica um código padrão de identificação de esquemas no âmbito da ITU-T (*International Telecommunication Union - Telecommunication Standardization Sector*). A cláusula `MUST` indica que os atributos `sn` (sobrenome) e `cn` (prenome) são de preenchimento obrigatório e a cláusula `MAY` lista os atributos opcionais. A cláusula `SUP` lista as classes das quais a classe definida herda os atributos. Portanto, a classe `person` herda da classe `top`, enquanto a classe `organizationalPerson` herda os atributos definidos na classe `person`.

Uma entrada é armazenada na AID como um conjunto de pares *nome do atributo* e *valor do atributo*, conforme ilustrado no exemplo seguinte:

```
dn: uid=josé.antônio,ou=people,dc=incor,dc=usp,dc=br
objectclass: top
objectclass: person
objectclass: organizationalperson
uid: josé.antônio
cn: José Antônio
sn: Andrade
mail: jose.antonio@incor.usp.br
userpassword: {SHA}YNN0ZFkRxVtEna76ZG0xkctwUbE=
```

Cada entrada é identificada unicamente numa AID pelo seu *Distinguished Name* (DN). O DN é similar ao caminho completo que identifica unicamente um arquivo num sistema de arquivos, exceto que é lido de forma reversa. No caso acima, o DN que identifica a pessoa de nome “José Antônio de Andrade” é `dn: uid=josé.antônio,ou=people,dc=incor,dc=usp,dc=br` (ver a entrada correspondente na AID da *Figura 13*). Quando a identificação do DN é relativa a um certo nó da AID, ela é denominada de RDN (*Relative Distinguished Name*), por exemplo o RDN `uid=josé.antônio,ou=people`.

4.2.1.2 Representação do MACA: aspectos de projeto

Dois aspectos são considerados na representação do MACA: o primeiro refere-se à como estruturar hierarquicamente as informações de usuários, papéis, autorizações e recursos na AID da BIGS (subseção Projeto do *Namespace*) e a segunda diz respeito à quais tipos de informações serão armazenados e com quais nomes de atributos (subseção Projeto de Esquemas).

Projeto do Namespace

É o *namespace* que estrutura hierarquicamente as informações numa AID, provendo os meios pelos quais os registros armazenados são identificados univocamente através de um *distinguished name*. Seu projeto deve ser criterioso, pois afeta o modo como as informações são referenciadas, organizadas, mantidas, replicadas, distribuídas, enfim, administradas (Howes *et al.*, 1999). O projeto envolve inicialmente a definição do sufixo do diretório (nó raiz, ou domínio) e depois a estrutura hierárquica do *namespace*.

O sufixo situa um diretório no espaço de nomes de um sistema de diretório global. O sufixo `dc=incor,dc=usp,dc=br` foi o escolhido para a AID da BIGS utilizada no InCor – HC-FMUSP (*Figura 13*). `dc` é o atributo que descreve os componen-

tes de um domínio, onde cada valor associado indica os nomes que formam o domínio DNS da instituição onde o diretório está situado, no caso, `incor.usp.br`. Essa forma de sufixação é a recomendada pela IETF (Yeong *et al.*, 1995). Além de identificar a instituição num serviço de diretórios LDAP global, o sufixo complementa os *distinguished names* dos registros armazenados no diretório, de acordo com o seguinte padrão: `<nome>,dc=incor,dc=usp,dc=br`, onde `<nome>` corresponde ao RDN do registro.

O *relative distinguished name* é afetado pela estrutura hierárquica do *namespace*, que pode variar desde uma forma plana, linear, até uma estrutura altamente hierarquizada, com múltiplos níveis. Howes *et al.* (1999) recomendam utilizar um *namespace* mais achatado porque ele acarreta menos mudanças nos RDNs e facilita a administração do diretório. A *Figura 13* ilustra a estrutura hierárquica do *namespace* do MACA no InCor – HC-FMUSP. Os registros de todos os usuários da instituição são armazenados sob o nó `ou=people`, num espaço de nomes linear, isto é, uma lista de registros. Assim, o padrão de RDN de usuário é `uid=<login>,ou=people` e, conseqüentemente, o valor do atributo `uid` identifica univocamente cada usuário cadastrado. Abaixo do nó `ou=groups` estão os registros das estruturas de agrupamento hierárquicas, usadas para representar a hierarquia de papéis (como uma estrutura de árvore invertida) do MACA (seção 3.3.2). Os objetos (recursos) que têm o seu acesso controlado são armazenados sob o nó `ou=resources`, também de modo hierárquico. É o caso dos objetos que representam o PEP e o MACA, parcialmente ilustrados na *Figura 13*. Finalmente, as autorizações são armazenadas sob o nó `ou=authorizations`, agrupadas de acordo com os papéis a que estão associadas. Por exemplo, as autorizações vinculadas diretamente ao papel *Usuário* são mantidas numa lista de autorizações abaixo da entrada `ou=usuario,ou=authorizations`. Logo, a relação papel-autorização é estabelecida pela estrutura hierárquica da AID.

Projeto de Esquemas

O esquema de um diretório é um conjunto de regras que determina os tipos de valores que podem ser armazenados e como o servidor e seus clientes devem tratar tais valores durante a execução de operações, como a de consulta, por exemplo (Howes *et al.*, 1999). Antes de um servidor de diretório armazenar um novo registro ou atualizar um já existente, ele verifica se o que vai ser inserido está de acordo com as regras do esquema. Sempre que compara os valores de atributos, ele consulta o

esquema para determinar o algoritmo de comparação a usar (Howes *et al.*, 1999). Por exemplo, o algoritmo que compara números é diferente do algoritmo que compara valores alfanuméricos.

Cada registro armazenado numa AID tem associado um conjunto de classes de objetos que especifica o esquema do registro. O LDAP possui uma multitude de classes de objetos e atributos predefinidos adequados para descrever usuários, papéis, estruturas de agrupamento, etc. Para o diretório possuir maior interoperabilidade com os clientes LDAP existentes, incluindo os principais sistemas operacionais, recomenda-se reutilizar ao máximo as classes e atributos preexistentes (Howes *et al.*, 1999).

A *Tabela 2* relaciona as classes de objetos LDAP utilizadas para representar as entidades do MACA, no caso, usuários, papéis, recursos e autorizações, com respectivos relacionamentos. O tipo *Usuário* possui um esquema que suporta os atributos comumente encontrados em contas de usuários numa organização. Desses atributos, os únicos que são requeridos para o MACA funcionar são o nome (cn), o sobrenome (sn), a senha (userPassword), o login (uid) e o atributo employeeNumber, que armazena um identificador único do usuário nos sistemas de informação da organização, como, por exemplo, o número do cadastro de pessoa física (CPF). Logo, uma entrada mínima de uma conta de usuário no MACA poderia ser a seguinte:

```
dn: uid=maria.augusta,ou=people,dc=incor,dc=usp,dc=br
objectClass: top
objectClass: person
objectClass: inetorgperson
uid: maria.augusta
cn: Maria
sn: Augusta
employeeNumber: 00000000000
userPassword:: {SHA}YNN0ZFkRxVtEna76ZGOxkctwUbe=
```

Devido a sua flexibilidade, o LDAP permite que diferentes entradas tenham diferentes classes e atributos, num mesmo ou em servidores distintos, o que facilita a configuração personalizada de contas, de acordo com o usuário e a instituição. Por exemplo, as classes *posixAccount* e (*Tabela 2*) *mailRecipient* são usadas para permitir a unificação da autenticação de usuários em sistemas operacionais *Unix-like* e em servidores de e-mail, respectivamente. Diferentes ambientes podem requerer outros atributos para autenticar (*e. g.*, certificado digital), porém, eles podem coexistir numa mesma conta de usuário no LDAP.

Tabela 2 – Esquemas LDAP para representar entidades do MACA.

Entidade	Classes de Objetos	Descrição	OID
Usuário			
	person	Define registros que representam genericamente pessoas, através de atributos para nome (cn), sobrenome (sn), senha (userPassword), dentre outros. Obrigatória.	2.5.6.6
	inetOrgPerson	Define registros que representam pessoas numa intranet institucional, com destaque para os atributos uid (nome de login do usuário), userCertificate (certificado digital de identificação), mail (correio eletrônico), labeledUrl (página na Internet) e outros atributos de caracterização de um empregado, como matrícula, departamento, foto, etc. Obrigatória.	2.16.840.1.113730.3.2.2
	organizationalPerson	Define registros que representam pessoas empregadas ou associadas a uma instituição, com atributos para descrever cargo (title), endereços e telefones comerciais, etc. Obrigatória, mas é herdada pela classe inetOrgPerson.	2.5.6.7
	posixAccount	Representa uma conta nos sistemas operacionais <i>Unix-like</i> , com atributos para indicar o login do usuário (uid), números de identificação para usuário (uidNumber) e grupos (gidNumber), <i>shell</i> de preferência (loginShell) e diretório padrão (homeDirectory), dentre outros. Opcional.	1.3.6.1.1.1.2.0
	mailRecipient	Possui atributos para armazenar informações da conta de correio eletrônico do indivíduo, incluindo diretório de <i>e-mails</i> (mailMessageStore) e quota de espaço de armazenamento (mailQuota). Opcional.	2.16.840.1.113730.3.2.3
Papel			
	incorRole	Define um registro que nomeia (atributo cn) e descreve (atributo description) um papel do MACA. Obrigatória.	1.3.6.1.4.1.8651.1.4
	incorGroup	Define um registro que permite o armazenamento dos usuários que são membros de um papel, isto é, que estabelece a relação usuário-papel. Os DNs dos usuários associados ao papel são armazenados no atributo multivalorado member. Obrigatória só quando o papel tem usuários associados, isto é, membros.	1.3.6.1.4.1.8651.1.5
Recurso			
	incorResource	Classe que descreve um recurso protegido. Os atributos cn, initials e description armazenam o nome, a sigla (ou abreviação) e a descrição do recurso (objeto) representado, respectivamente. O atributo incorResourceType especifica o tipo do recurso (<i>e.g.</i> , arquivo, tabela, objeto, aplicação, menu, etc.), enquanto incorResourceAccessMode especifica as formas de acesso (<i>e.g.</i> , leitura, escrita, execução, etc.) que podem ser realizadas no mesmo, isto é, a operação associada ao objeto. Por fim, o atributo l armazena o local onde o recurso reside na instituição. Obrigatória.	1.3.6.1.4.1.8651.1.2
Autorização			
	IncorAuthorization	Classe que descreve uma autorização. Os atributos incorRoleDN e incorResourceDN referenciam os DNs do papel e do recurso (objeto), respectivamente, para os quais a autorização se aplica. Os atributos incorAuthorizationType e incorPrivilegeType especificam o tipo da autorização (<i>e.g.</i> , forte ou fraca) e o tipo de privilégio (<i>e.g.</i> , positivo, negativo ou uma expressão contextual). A operação do objeto na autorização (<i>e.g.</i> , leitura, escrita, execução, alteração) é armazenado no atributo incorPrivilege. O atributo incorResourceType especifica o tipo do recurso protegido. A identificação da autorização é armazenada no atributo cn. Obrigatória.	1.3.6.1.4.1.8651.1.3

As classes de objetos `incorRole`, `incorGroup`, `incorResource` e `incorAuthorization` foram definidas para atender necessidades específicas do MACA. Por exemplo, a entrada abaixo

```
dn: cn=Profissional de Saúde,cn=usuario,cn=Papeis,ou=Groups,
  dc=incor,dc=usp,dc=br
objectClass: top
objectClass: incorrole
cn: Profissional de Saúde
description: Profissionais que assistem o paciente diretamente.
```

especifica o papel *Profissional de Saúde* ilustrado na *Figura 13*. Como não possui usuários diretamente associados, por ser um papel abstrato, a classe `incorGroup` não está presente. Já a entrada

```
dn: cn=Médico,cn=Profissional de Saúde,cn=usuario,cn=Papeis,ou=Groups,
  dc=incor,dc=usp,dc=br
objectClass: top
objectClass: incorrole
objectClass: incorgroup
cn: Profissional de Saúde
description: representa as funções e responsabilidades dos médicos.
member: uid=maria.augusta,ou=people,dc=incor,dc=usp,dc=br
member: uid=josé.antônio,ou=people,dc=incor,dc=usp,dc=br
```

especifica o papel *Médico* (*Figura 13*) com dois usuários associados no atributo multivalorado `member`, da classe `incorGroup`. Utilizando a linguagem de consulta do LDAP é possível recuperar com facilidade, tanto os usuários associados a um papel, quanto os papéis associados a um determinado usuário. Portanto, a relação usuário-papel fica estabelecida.

A classe `incorResource` possibilita a representação dos objetos e operações protegidos. Por exemplo, a entrada

```
dn: cn=Prescrição,cn=Serviço de Prescrição Médica,cn=PEP,ou=Resources,
  dc=incor,dc=usp,dc=br
objectClass: top
objectClass: incorresource
incorresourcetype: Servidor de Aplicação
cn: Prescrição
incorresourceaccessmode: criar
incorresourceaccessmode: ler
```

denota o objeto *Prescrição*, do serviço de prescrição médica (*Figura 13*), com as operações para criar e ler prescrições, definidas pelo atributo multivalorado `incorresourceaccessmode`. Por fim, usa-se a classe `incorAuthorization` para representar uma autorização do MACA. A entrada a seguir, por exemplo,

```
dn: cn=autorização 1,ou=usuario,ou=Authorizations,dc=incor,dc=usp,dc=br
objectClass: top
objectClass: incorauthorization
incorprivilege: -
incorprivilege: ler
incorresourcedn: cn=Prescrição,cn=Serviço de Prescrição Médica,cn=PEP,
ou=Resources,dc=incor,dc=usp,dc=br
incorauthorizationtype: weak
incorroledn: cn=Usuario,cn=Papeis,ou=groups,dc=incor,dc=usp,dc=br
cn: autorização 1
```

especifica a primeira autorização do papel *Usuário* (atributo `incorroledn`), ilustrada na (Figura 13). Ela estabelece que *Usuário*, em geral —pode ser revogada, pois é fraca (atributo `incorauthorizationtype`)—, está proibido (atributo `incorprivilege`) de executar a operação *ler* (atributo `incorprivilege`) no objeto *prescrição* (atributo `incorresourcedn`). A relação papel-autorização é representada pelo atributo `incorroledn`. Assim, dada uma autorização qualquer, é possível saber todos os papéis para os quais ela se aplica e, por conseguinte, os usuários associados a cada um destes papéis. De modo oposto, dado um usuário, é possível saber todos os seus papéis e, para cada papel, todas as autorizações aplicáveis, conforme o modelo para CABP preconiza.

4.2.2 Servidor de segurança

O servidor de segurança implementa os aspectos transientes do MACA, como o gerenciamento de sessões, a ativação de papéis e a autorização de acesso, e também a autenticação de usuários. Ele foi desenvolvido na linguagem Java (Arnold *et al.*, 2000), usando a plataforma Java 2 SE (Sun Microsystems, 2003a), versão 1.4.1. O produto de código fonte aberto JacORB (Brose & Noffke, 2002), um ORB para linguagem Java, foi utilizado na implementação das interfaces CORBA de autenticação e de autorização de acesso, do SSC e do RAD – *Facility*, respectivamente. O JacORB foi configurado para estabelecer, nas interações entre os objetos CORBA, a comunicação usando o protocolo IIOP sobre TLS, com autenticação mútua das partes. O servidor LDAP é acessado através de conexões LDAP sobre TLS usando a API JNDI (*Java Naming Directory Interface*) (Lee & Seligman, 2000) da plataforma Java 2 SE.

As instruções de instalação e configuração do servidor de segurança do MACA, versão 3.2, incluindo as configurações para conexão com LDAP sobre TLS e dos clientes CORBA com IIOP sobre TLS estão nos manuais “MACA: guia de instalação e configuração” e “MACA cliente: guia do programador”, respectivamente (v. item Manuais na p. 195). A implementação do MACA está disponibilizada como software livre (Hexsel, 2003) sob licença GNU GPL (*General Public License*).

A *Figura 14* ilustra, com um diagrama de seqüência na notação UML (Booch *et al.*, 1999), as interações existentes entre os principais objetos do servidor de segurança, quando um cliente solicita os serviços de autenticação de usuário e de autorização de acesso. Cabe ressaltar que a atual implementação do MACA não foi integrada como política de autorização para o controle de acesso imposto automaticamente pelo ORB implementado pelo JacORB.

4.2.2.1 Autenticação: objetos e interações

A autenticação inicia-se quando uma aplicação cliente (objeto *c* na *Figura 14*) invoca a operação `authenticate`, da instância da interface `PrincipalAuthenticator` (objeto *pa* na *Figura 14*), após a obtenção do login e da senha do usuário. A operação `authenticate`, com a IDL (*Interface Definition Language*) CORBA abaixo definida,

```
interface PrincipalAuthenticator {  
    ...  
    AuthenticationStatus authenticate(  
        in AuthenticationMethod method,  
        in SecurityName security_name,  
        in Opaque auth_data,  
        in AttributeList privileges,  
        out Credentials creds,  
        out Opaque continuation_data,  
        out Opaque auth_specific_data  
    );  
    ...  
};
```

recebe os seguintes parâmetros de entrada: `method`—o método de autenticação (*e.g.*, senha simples, certificado digital); `security_name`—login do usuário; `auth_data`—a senha ou outras informações que provem a identidade do usuário; e `privileges`—os atributos de segurança requeridos pelo usuário (*e.g.*, um papel inicial para ativação). A atual implementação do servidor de segurança suporta apenas a autenticação simples, baseada em login e senha. Portanto, a primeira ação da operação de autenticação é tentar estabelecer uma conexão para o usuário no servidor LDAP (objeto `ldap` na *Figura 14*) pela chamada da operação `bind`. É essa operação do protocolo LDAP quem efetivamente autentica o usuário, de acordo com as informações presentes em sua conta (atributos `uid` e `userpassword`). Caso bem sucedida, a operação `bind` retorna a conexão LDAP estabelecida. Em seguida, a autenticação prossegue com a criação de um objeto transiente, instância da interface `Credentials` do SSC, que se adiciona como uma sessão no serviço de autorização do MACA (objeto `maca` na *Figura 14*) com a chamada da operação `addNewSession`.

Figura 14 – Diagrama de sequência com as interações entre os objetos do servidor de segurança, para prover autenticação e autorização de acesso.

A credencial recém-criada retorna (parâmetro de saída `creds`) para o cliente após a execução da operação `authenticate` e passa a representar a sessão do usuário no MACA. Os dois últimos parâmetros de retorno são usados em mecanismos de autenticação com múltiplos estágios. A resposta `AuthenticationStatus` indica se o usuário foi ou não autenticado. Em caso de insucesso, a credencial não é criada e, portanto, nenhuma sessão é iniciada para o usuário no serviço do MACA.

Com a credencial pode-se obter informações da conta do usuário no servidor LDAP, tais como o login e os papéis associados. Também permite recuperar os papéis ativos e ativá-los explicitamente. Um usuário possui tantas credenciais instanciadas quantas forem as suas sessões abertas e o conjunto de seus papéis ativos corresponde aos papéis ativados em cada sessão. Os papéis somente são desativados quando as credenciais são destruídas (operação `destroy` na *Figura 14*), encerrando-se as sessões do usuário no serviço de autorização do MACA (objeto `maca` na *Figura 14*) com a chamada à operação `closeSession`. Nesse momento, a conexão do usuário no servidor LDAP também é encerrada (operação `unbind` na *Figura 14*).

4.2.2.2 Autorização de acesso: objetos e interações

Com o usuário autenticado e de posse de sua credencial, o cliente pode solicitar autorizações de acesso ao objeto `ado` (*Figura 14*), instância da interface `AccessDecision` do RAD – *Facility*, com a IDL CORBA a seguir:

```
interface AccessDecision {

    boolean access_allowed(
        in ResourceName resource_name,
        in string operation,
        in AttributeList attribute_list
    )
    raises (InternalError);

    BooleanList multiple_access_allowed(
        in AccessDefinitionList access_requests,
        in AttributeList attribute_list
    )
    raises (InternalError);
};
```

A interface `AccessDecision` oferece a operação `access_allowed` para solicitar uma única autorização para executar uma operação (parâmetro de entrada `operation`) num objeto protegido (parâmetro de entrada `resource_name`). Ela

retorna um booleano para indicar se autorização foi concedida ou negada. A operação `multiple_access_allowed` permite solicitar múltiplas autorizações numa só chamada, passando-se como parâmetro uma lista de pares de nome de objeto e operação (parâmetro de entrada `access_requests`). Ela retorna uma lista booleanos com cada posição correspondendo ao resultado de uma solicitação. Em caso de erro, uma exceção é levantada para diferenciar uma falha de uma negação de acesso. Além da operação e do nome do objeto protegido, ambas as operações requerem o parâmetro `attribute_list` —um conjunto de atributos válidos do usuário que solicita autorização de acesso, tais como, o login ou um certificado digital. No caso específico da implementação do MACA, o atributo requerido é a referência da sessão corrente associada a credencial do usuário.

Portanto, antes de solicitar uma autorização de acesso (operação `access_allowed` na *Figura 14*), o cliente deve invocar a operação `get_attributes` da credencial do usuário para obter a referência da sessão correspondente. Em seguida, o cliente solicita uma ou mais autorizações de acesso chamando a operação `access_allowed` com os parâmetros correspondentes ao objeto protegido, a operação a executar e a referência da sessão. Nos casos de autorizações parametrizadas, a lista de argumentos é passada concatenada ao nome do objeto. Em sua execução, `access_allowed` extrai a lista dos eventuais argumentos a serem passados para as regras de autorizações parametrizadas e invoca a operação `hasAccess` do serviço de autorização do MACA (objeto `maca` na *Figura 14*). A operação `hasAccess` pesquisa no servidor LDAP as autorizações de acesso aplicáveis ao usuário de acordo com o algoritmo de decisão de acesso do MACA especificado no esquema *MACA_AcessoAutorizado* (seção 3.3.4, p. 78). Ou seja, `hasAccess` implementa o algoritmo de decisão de acesso.

O Algoritmo de Decisão de Acesso do MACA

O algoritmo de decisão de acesso implementado na operação `hasAccess` (*Figura 14*) e especificado no esquema *MACA_AcessoAutorizado* recebe os seguintes parâmetros de entrada: a operação a executar (`opr`), o objeto protegido (`obj`), a lista de parâmetros da autorização (`params`) e a referência da sessão (`session_ref`). Retorna um valor booleano indicando se o acesso deve ser permitido ou proibido. Os seguintes passos são executados para decidir o acesso:

1. Caso o conjunto de papéis para ativar do usuário seja vazio, o acesso é negado (retorna falso) e o algoritmo pára;

2. Caso contrário, verifica-se se existe alguma autorização válida, forte, associada a algum papel ativo ou para ativação do usuário que proíba a execução a operação `opr` no objeto `obj`. Caso exista, o acesso é negado (retorna falso) e o algoritmo pára. A execução desse passo compreende os seguintes subpassos:
 - 2.1. Verificar a existência de alguma autorização válida, forte e negativa (estática), primeiro nos papéis ativos, depois nos papéis disponíveis para ativação. Caso exista, o acesso é negado (retorna falso) e o algoritmo pára;
 - 2.2. Caso contrário, verifica a existência de alguma autorização válida, forte, com regra de autorização (primeiro nos papéis ativos, depois nos papéis disponíveis para ativação) que, quando avaliada pelo interpretador de regras com a lista parâmetros `params`, resulte numa proibição de acesso. Caso exista, o acesso é negado (retorna falso) e o algoritmo pára;
3. Caso o algoritmo não tenha parado, então verifica-se a existência de alguma autorização válida, forte, associada a algum papel ativo ou para ativação do usuário que permita executar a operação `opr` no objeto `obj`. Caso exista, o acesso é concedido (retorna verdadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára. A execução desse passo compreende os seguintes subpassos :
 - 3.1. Verificar a existência de alguma autorização válida, forte e positiva (estática), primeiro nos papéis ativos, depois nos papéis disponíveis para ativação. Caso exista, o acesso é concedido (retorna verdadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára;
 - 3.2. Caso contrário, verifica a existência de alguma autorização válida, forte, com regra de autorização (primeiro nos papéis ativos, depois nos papéis disponíveis para ativação) que, quando avaliada pelo interpretador de regras com a lista parâmetros `params`, resulte numa permissão de acesso. Caso exista, o acesso é concedido (retorna ver-

dadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára;

4. Caso o algoritmo não tenha parado, então verifica-se a existência de alguma autorização válida, fraca associada a algum papel ativo ou para ativação do usuário que permita executar a operação `opr` no objeto `obj`. Caso exista, o acesso é concedido (retorna verdadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára. A execução desse passo compreende os seguintes subpassos :

- 4.1. Verificar a existência de alguma autorização válida, fraca e positiva (estática), primeiro nos papéis ativos, depois nos papéis disponíveis para ativação. Caso exista, o acesso é concedido (retorna verdadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára;

- 4.2. Caso contrário, verifica a existência de alguma autorização válida, fraca, com regra de autorização (primeiro nos papéis ativos, depois nos papéis disponíveis para ativação) que, quando avaliada pelo interpretador de regras com a lista parâmetros `params`, resulte numa permissão de acesso. Caso exista, o acesso é concedido (retorna verdadeiro), o eventual papel inativo envolvido é ativado na sessão `session_ref` e o algoritmo pára;

5. Caso o algoritmo não tenha parado, então o acesso é negado (retorna falso) e o algoritmo pára.

Pode-se verificar informalmente que o algoritmo anterior satisfaz a especificação do esquema *MACA_AcessoAutorizado* (seção 3.3.4, p. 78), supondo que a sessão, a operação e o objeto atendem a pré-condição da operação. Primeiro, somente autorizações válidas, de papéis que o usuário tem disponíveis para ativar são consideradas (passos 1, 2, 3 e 4). Segundo, uma autorização positiva somente concede o acesso caso as duas seguintes condições sejam verificadas: 1) se a autorização é forte (o algoritmo pára no passo 3), então não existe uma autorização forte equivalente que proíba o acesso (o algoritmo não parou no passo 2); 2) se a autorização é fraca (o algoritmo pára no passo 4), então não existe uma autorização forte equivalente que proíba (o algoritmo não parou no passo 2) ou permita (o algoritmo não parou no passo 3) o acesso. Se o acesso for autorizado, o papel correspondente é ativado, caso ainda não

esteja ativo. Por fim, o acesso somente é negado por um dos seguintes motivos: 1) quando o algoritmo pára no passo 2 porque existe uma autorização forte que proíbe o acesso; ou 2) quando o algoritmo pára no passo 5 porque não existe uma autorização que conceda o acesso, forte ou fraca.

Note-se, pela sequência de execução dos passos do algoritmo, que as autorizações fortes e negativas prevalecem sobre as demais autorizações, inclusive as eventuais autorizações equivalentes, fortes e positivas, atendendo, portanto, a política de resolução de conflitos fortes do MACA (v. seção 3.3.3, p. 69). Por sua vez, autorizações fortes prevalecem sobre as autorizações fracas equivalentes e as autorizações fracas positivas prevalecem sobre as autorizações fracas negativas equivalentes.

A operação `hasAccess` pode ser configurada no serviço de autorização do MACA para funcionar com uma memória *cache* dos resultados das últimas n solicitações de autorização de um usuário. A memória *cache* permanece ativa enquanto as sessões do usuário permanecem abertas, sendo destruída quando elas são encerradas. O objetivo é reduzir o tempo de resposta das solicitações de autorização mais freqüentemente chamadas por um usuário e também diminuir o número global de consultas no servidor LDAP, visando um melhor desempenho (v. avaliação de desempenho do serviço de autorização do MACA na seção 4.3.1, p. 135).

Os aspectos relevantes da implementação de contextos no servidor do MACA e do interpretador de regras usado na avaliação das autorizações contextuais são apresentados a seguir.

Contextos e o Interpretador de Regras: Aspectos de Implementação

Um contexto é criado no MACA com a implementação de uma interface de programação, cuja IDL CORBA é apresentada a seguir:

```
module Contexts {
    interface Context {
        Any getValue (in string index) raises (ContextException);
        boolean inEvaluation(in Any element, in Any aSet)
                                raises (ContextException);
        Any functionApplication(in string functionName,
                                in Vector argumentList)
                                raises (ContextException);
    };
};
```

Portanto, cada contexto disponível para o servidor do MACA implementa a interface `Context` acima, com as seguintes operações:

- `getValue`: retorna um objeto do tipo `Any` associado à *string* do parâmetro de entrada `index`, cujo valor denota o nome de uma variável contextual. Por exemplo, as variáveis contextuais associadas ao usuário corrente, como “login” e “matrícula” poderiam ser implementadas nessa operação, retornando valores dos tipos *string* e inteiro, respectivamente. O tipo do retorno da operação, `Any`, admite valores de qualquer tipo;
- `inEvaluation`: é usada para implementar a operação que testa a pertinência de um elemento —`element`— num conjunto denotado pelo valor de `aSet`. Por exemplo, uma implementação dessa operação pode verificar se um paciente está internado. Assim, `element` denota o identificador único de um paciente e `aSet` determina a submissão de uma consulta em SQL para verificar se ele está internado. Em caso afirmativo, o valor booleano verdadeiro é retornado, caso contrário, o valor falso;
- `functionApplication`: permite a implementação de contextos mais complexos mediante chamadas de funções. O valor do parâmetro `functionName` especifica o nome da função, que tem a lista de argumentos denotada pelo parâmetro do tipo vetor (seqüência de valores do tipo `Any`), chamado `argumentList`. Uma função para verificar se o usuário corrente faz parte da equipe que atende um paciente específico é um exemplo de sua aplicação.

Em caso de erro na execução de uma das operações, uma exceção é levantada para informar o interpretador de regras a ocorrência de uma situação anormal. Esse, por sua vez, propaga a exceção para o algoritmo de decisão de acesso, do serviço de autorização do MACA, que informa o cliente sobre o erro.

O código em Java a seguir exemplifica uma possível implementação do contexto de pacientes numa organização de saúde:

```
package contexts;
...
public class PatientContext implements Context {
    private DBConnectionManager connectionMgr = DBConnectionManager.getInstance();

    public PatientContext() {}

    public Object getValue(String index) throws ContextException {
        if (index.equalsIgnoreCase("totalLeitos")) {
            Connection con = connectionMgr.getConnection("base de dados");
            if (con != null) { try {
```

```

        Statement stm = con.createStatement();
        ResultSet rs = stm.executeQuery("Select count(*) From Leitos");
        int total = rs.getInt(1);
        connectionMgr.freeConnection("base de dados", con);
        return new Integer(total);
    } catch (java.sql.SQLException exc) { ... }
}

else if (index.equalsIgnoreCase("leitosOcupados")) { ... }
throw new ContextException("Variável '"+index+"' não definida");
}

public boolean inEvaluation(Object elem, Object aSet) throws ContextException {
    if ((aSet instanceof String) && (aSet.equalsIgnoreCase("pacientes_internados"))) {
        Connection con = connectionMgr.getConnection("base de dados");
        if ((con != null) && (elem instanceof String)) { try {
            Statement stm = con.createStatement();
            ResultSet rs = stm.executeQuery("Select pac_id From pacs_intern "+
                                           "Where pac_id = '"+elem.toString()+"'");

            boolean result = rs.next();
            connectionMgr.freeConnection("base de dados", con);
            return result;
        } catch (java.sql.SQLException exc) { ... }
        }
    }
    throw new ContextException("Conjunto '"+aSet+"' não definido");
}

public Object functionApplication(String functionName, Vector argumentList) throws ... {
    if (functionName.equalsIgnoreCase("assistido_por")) {
        Connection con = connectionMgr.getConnection("base de dados");
        if ((con != null) && (argumentList.size() == 2)) { try {
            String pacID = (String) argumentList.get(0), regProf = (String) argumentList.get(1);
            Statement stm = con.createStatement();
            ResultSet rs = stm.executeQuery("Select pac_id From pacientes, profissionais, procedimentos "+
                                           "Where pacientes.pacs_id = procedimentos.pacs_id and "+
                                           "profissionais.reg_prof = procedimentos.reg_prof and "+
                                           "pacientes.pacs_id = '"+pacID+"' and "+
                                           "profissionais.reg_prof = '"+regProf+"'");

            boolean result = rs.next();
            connectionMgr.freeConnection("base de dados", con);
            return result;
        } catch (java.sql.SQLException exc) { ... }
        }
    }
    throw new ContextException("Função '"+functionName+"' não definida");
}
}

```

As interfaces CORBA das operações estão mapeadas para interface Java Context (não mostrada), implementada pelos métodos Java correspondentes na classe PatientContext. A primeira operação (getValue) possui duas variáveis contextuais definidas: uma que retorna a quantidade total de leitos do estabelecimento (totalLeitos) e outra que retorna o número total de leitos ocupados (leitosOcupados). A segunda operação (inEvaluation) testa se um paciente (denotado pelo parâmetro elem) pertence ao conjunto contextual dos

pacientes internados (`pacientes_internados`). A última operação (`functionApplication`) implementa a função `assistido_por` com dois parâmetros: o primeiro denotando uma identificação única de paciente e o segundo indicando o número do registro de um profissional de saúde. A função testa se um paciente já foi assistido pelo profissional indicado. As informações contextuais são obtidas por meio de consultas SQL submetidas através de conexões ao SGBD, usando um gerenciador de *pool* de conexões apropriado (`DBConnectionManager`). Recomenda-se que tais conexões sejam realizadas sobre TLS para obter confidencialidade da comunicação e autenticação confiável das partes.

Uma vez implementado e compilado, um contexto é carregado no serviço de autorização do MACA usando o mecanismo de extensão Java (Sommerer, 2002). O mecanismo de extensão permite a adição de novas classes de contextos ao serviço de autorização do MACA sem que se precise modificá-lo ou recompilá-lo. Apenas configurações externas são requeridas. Contextos inteiramente implementados em Java, que compartilham o mesmo espaço de memória do serviço de autorização do MACA, são denominados de contextos locais. Contextos remotos são aqueles que executam as operações da interface CORBA Context num espaço de memória separado, sendo acessados por clientes Java através do protocolo IIOP sobre TLS.

O interpretador de regras de autorização do MACA, no momento em que é instanciado, detecta, nas configurações do serviço de autorização, os contextos carregados disponíveis. Durante a avaliação de uma expressão contextual, o interpretador de regras invoca as operações implementadas no contexto correspondente. A operação a ser chamada vai depender da forma da expressão, que pode denotar uma variável, um conjunto ou uma função contextuais. Por exemplo, na avaliação da expressão

```
exp-abs(umCodPac, regProf) {
    pacCtx.leitosOcupados < pacCtx.totalLeitos &
    !(umCodPac in pacCtx.pacientes_internados) &
    pacCtx.responsavel_por(umCodPac, regProf)
} ("123456-H", "99145SP")
```

o interpretador de regras executa o equivalente ao seguinte código Java:

```
Vector params = new Vector();
params.add("123456-H"); params.add("99145SP");
...
(((Integer)pacCtx.getValue("leitosOcupados")).intValue() <
 ((Integer)pacCtx.getValue("totalLeitos")).intValue()) &&
!(pacCtx.inEvaluation("123456-H", "pacientes_internados")) &&
(pacCtx.functionApplication("responsavel_por", params))
```

No processo de interpretação, o nome do contexto (`pacCtx`) passa a identificar o objeto da classe que o implementa (`PatientContext`) em Java, de modo que as operações do contexto apropriado sejam invocadas. Os parâmetros formais da regra (`umCodPac` e `regProf`) são substituídos pelos argumentos fornecidos ("123456-H" e "99145SP"), conforme ilustrado no código Java anterior. O interpretador de regras foi implementado como uma extensão do produto de código fonte aberto *SableCC* (SableCC, 2003), versão 2.16.2, tomando por base a sintaxe especificada no Anexo A. O *SableCC* é um *framework* orientado a objetos para auxiliar a construção de compiladores e interpretadores em Java (Gagnon & Hendren, 1998).

4.2.3 Ferramentas administrativas

O MACA dispõe de três ferramentas administrativas. O MACA AD *Web*, para gerenciar as contas de usuários, o MACA AD, para administrar as autorizações na BIGS e o MACA CS Monitor, para monitorar a atividade do servidor de autorização do MACA. As instruções para instalar e configurar as ferramentas estão no manual "MACA: guia de instalação e configuração", incluindo aquelas para estabelecer a comunicação com os protocolos IIOP e LDAP através de conexões TLS. O manual "MACA administrativo: manual do usuário" traz um tutorial sobre a configuração de políticas de acesso no MACA AD e no MACA AD *Web* (v. item Manuais na p. 195).

As subseções seguintes descrevem as principais funcionalidades disponíveis nas ferramentas e especifica as respectivas plataformas de implementação e operação.

4.2.3.1 O MACA AD *Web*

O MACA AD *Web* foi implementado em Java, na plataforma Java 2 SE (versão 1.4.1), usando a tecnologia de páginas dinâmicas JSP (Sun Microsystems, 2003b) disponível no servidor *Web/JSP Tomcat* (Apache Software Foundation, 2003), versão 4.0.3, um produto de código fonte aberto. A página de consulta no MACA AD *Web* (*Figura 15*) permite que um usuário vinculado a um papel administrativo apropriado pesquise por contas utilizando valores presentes nos atributos de usuários armazenados no servidor LDAP. A consulta retorna uma lista com os atributos marcados para exibição (*e. g.*, nome, e-mail, login na *Figura 15*). Com a lista, o administrador pode selecionar uma ou mais contas para excluir da base ou então pode ir diretamente para a página que permite gerenciar uma conta específica (*Figura 16*).

A partir da página de gerência de contas, o administrador pode alterar os atributos armazenados para o usuário, incluindo o seu login, vincular/desvincular

Figura 15 – Página para consultar contas de usuários no MACA AD Web.

Figura 16 – Página para gerência de contas de usuários no MACA AD Web.

Figura 17 – Página do MACA AD Web para vinculação/desvinculação de papéis de usuários.

papéis (Figura 17), e gerenciar senhas. O MACA AD Web é instalado com um esquema de dados pré-definido em XML que especifica quais atributos constarão na conta de um usuário. O esquema pré-definido obriga o armazenamento do login (atributo `uid`), do nome completo (atributo `givenname`), do prenome (atributo `cn`), do sobrenome (atributo `sn`), do CPF (atributo `employeenumber`) e do e-mail (atributo `mail`). Porém esse esquema pode ser estendido para incluir novos atributos, de acordo com as necessidades de cada organização onde o MACA será instalado. No esquema, pode-se especificar o nome de exibição do atributo (e.g., `employeenumber` poderia ser exibido como CPF ou Matrícula), o tamanho máximo, valores *default*, se ele é obrigatório ou multivalorado, tabelas para consulta/validação de valores (e.g., consultar/validar o CPF ou matrícula na base de dados do recursos humanos de uma organização), etc. Em adição, o esquema permite agrupar os atributos em pastas, por afinidade, para facilitar a visualização. Por exemplo, na Figura 16, eles estão agrupados nas pastas de identificação, de dados funcionais, de endereço e de atributos requeridos para contas do sistema operacional Linux.

O MACA AD Web ainda permite a configuração de políticas que determinam o tamanho mínimo das senhas, o algoritmo de *hash* para o seu armazenamento (e.g., SHA, MD5), o prazo de expiração, o histórico de senhas que não podem ser reu-

tilizadas, dentre outras. A opção para gerenciar senhas (*Figura 16*) permite que o administrador de contas, com privilégios apropriados, gere automaticamente uma senha, desconhecida dele, mas que é fornecida de modo sigiloso ao usuário. Também possibilita, dependendo da política administrativa, que o administrador altere a senha de um usuário ou redefina prazos de expiração. Por fim, o MACA AD *Web* suporta a personalização do módulo para geração automática de senhas, a fim de atender necessidades específicas das organizações. Em adição, a implementação do MACA AD *Web* suporta a imposição de elaboradas políticas administrativas para as funções de gerência de contas, tais como a política ilustrada na *Figura 6* (p. 93).

4.2.3.2 O MACA AD

O MACA AD é uma aplicação Java implementada na plataforma Java 2 SE, versão 1.4.1. Permite a um administrador configurar políticas de acesso para o gerenciamento de recursos (objetos e operações), papéis e autorizações, com respectivos relacionamentos. Com o MACA AD (*Figura 18*), pode-se criar, ler, renomear, alterar e remover os recursos que representam as operações e os objetos protegidos, armazenados em atributos da classe `incorresource` (v. *Tabela 2* na p. 115) no servidor LDAP. Ele também disponibiliza, na pasta “Descrição” (*Figura 19*) as funções para criar, ler, renomear, alterar e remover papéis. Na pasta “Membros” (não exibida na *Figura 19*), é possível saber todos os usuários vinculados, direta e indiretamente, a um papel selecionado. Já na pasta “Conflitos” (não exibida na *Figura 19*), é possível identificar todos os papéis que possuem conflitos fortes estáticos com o papel selecionado. A pasta “Autorizações” (*Figura 20*), que permite a criação, leitura, alteração e remoção de autorizações associadas direta ou indiretamente ao papel selecionado. Em destaque (*Figura 20*), aparece a janela para configuração de uma autorização, com o editor de regras habilitado.

Cabe ressaltar que o MACA AD (e o MACA AD *Web*) foi implementado para reduzir as chances de violação da integridade do modelo do MACA armazenado no servidor LDAP. Os servidores LDAP, em geral, não são transacionais, tampouco têm capacidade de impor restrições de integridade referencial. Acessando diretamente o servidor LDAP, em princípio, é possível excluir um papel e as referências a seus membros, mesmo restando autorizações referenciando um papel que não mais existe. Por outro lado, é possível excluir uma conta de usuário deixando as eventuais referências de suas associações com papéis. Ou seja, um papel pode estar vinculado a um usuário que não mais existe. Por isso, o MACA AD (e o MACA AD *Web*) foi implementado para preservar a integridade do modelo. Por exemplo, só permite a

Figura 18 – Janela do MACA AD para gerência de recursos (objetos e operações).

Figura 19 – Janela do MACA AD para gerência de papéis.

Figura 20 – Janela do MACA AD para gerência de autorizações.

exclusão de papéis sem usuários, direta ou indiretamente, vinculados e sem autorizações diretamente associadas. Somente usuários que não têm papéis vinculados podem ser excluídos. Por conta disso, somente as ferramentas administrativas podem alterar o modelo do MACA no servidor LDAP. A implementação do MACA AD também suporta a imposição de elaboradas políticas administrativas para as funções de gerência de recursos, de papéis, de autorizações e de suas relações.

4.2.3.3 O MACA CS Monitor

O MACA CS Monitor foi implementado em Java, na plataforma Java 2 SE (versão 1.4.1), usando a tecnologia de páginas dinâmicas JSP disponível no servidor *Web/JSP* Tomcat (Apache Software Foundation, 2003), versão 4.0.3. Ele oferece informações sobre a atividade corrente do sistema (Figura 21), como os valores médios por minuto e totais de autenticações de usuário e de solicitações de autorizações de acesso, e também os valores médios por minuto de usuários e de *threads* correntes. O MACA CS Monitor ainda fornece a lista de usuários correntes com respectivas sessões abertas (Figura 22). A lista informa a data e a hora de início de cada sessão e permite a um administrador, com privilégios apropriados, encerrar compulsoriamente uma sessão particular. O administrador também pode limpar o *cache* dos resultados das últimas autorizações de acesso solicitadas por um usuário.

Figura 21 – Página do MACA CS Monitor com estatísticas do sistema.

Figura 22 – Página do MACA CS Monitor com a lista parcial de usuários com sessões abertas.

4.3 Resultados

O principal resultado deste trabalho é a disponibilidade de uma implementação do MACA como um produto de software estável, integrado a uma arquitetura baseada em padrões de processamento aberto e distribuído, apto para utilização em organizações reais. As subseções seguintes apresentam e discutem esse resultado sob os aspectos do seu desempenho e da sua aplicação prática no PEP do InCor-HC.FMUSP e num projeto piloto do DATASUS/MS.

4.3.1 Avaliação de desempenho

O desempenho do servidor de segurança que incorpora o MACA foi avaliado medindo-se os tempos de resposta às solicitações de autenticação de usuário e de autorização de acesso. O objetivo foi observar a variação do tempo de resposta em função de um aumento progressivo da carga de solicitações ao servidor realizadas pela simulação de múltiplos usuários simultâneos. Um conjunto de três experimentos foi realizado num ambiente de testes simulando as solicitações de um número crescente de usuários. O ambiente de testes possuía a seguinte configuração:

- Um servidor com dois processadores Pentium III 1133 MHz, com 1 Gbytes de memória principal, conectado em rede Ethernet a 100 Mbits/s, com os seguintes componentes de software em funcionamento: 1) sistema operacional Linux Suse 8.0, núcleo versão 2.4.18-64GB-SMP; 2) servidor OpenLDAP, versão 2.1.17; e 3) servidor de segurança com o serviço de autorização do MACA executando na máquina virtual da plataforma Java 2 SE, versão 1.4.1;
- Quatro estações de simulação, cada uma com um processador Pentium III 500 MHz, com 196 Mbytes de memória principal, conectado em rede Ethernet a 100 Mbits/s, com o sistema operacional Windows 2000 e o programa para simulação de solicitações de usuários, desenvolvido em Java para execução na plataforma Java 2 SE, versão 1.4.1, instalado;
- Uma estação de medição com um processador Pentium III 800 MHz, com 326 Mbytes de memória principal, conectado em rede Ethernet a 100 Mbits/s, com o sistema operacional Windows 2000 e o programa para medição dos tempos de resposta, desenvolvido em Java para execução na plataforma Java 2 SE, versão 1.4.1, instalado;

O servidor OpenLDAP armazenava um total de 21.274 contas de usuários fictícios e 84% das contas possuíam, em média, 1,2 papéis associados. Foram configurados 77 papéis e, desse total, 81% possuíam 340,8 usuários vinculados em média. Em cada papel havia, em média, 3,0 autorizações diretamente vinculadas e 18,6 autorizações herdadas. Do total de 231 autorizações definidas, 66% eram positivas, 4% eram negativas, 30% possuíam regras e 98% eram fracas (2% fortes). As autorizações estabeleciam uma política de acesso para um total de 99 objetos com diferentes operações associadas.

O programa de simulação podia ser configurado para simular solicitações (autenticação e autorização de acesso) de um número arbitrário de usuários. Os usuários eram representados por agentes, instâncias da classe `Thread`, da biblioteca de classes do Java 2 SE, que possuíam ramos de execução independentes dentro do programa. Cada agente realizava as seguintes tarefas indefinidamente:

1. Antes de realizar uma autenticação, o agente esperava um tempo aleatório variando entre zero a 5 minutos;
2. Em seguida, solicitava uma autenticação em nome de um usuário escolhido de forma aleatória no servidor OpenLDAP. Todos os usuários possuíam a senha igual ao login e 100% das solicitações de autenticação eram bem sucedidas;
3. Após a autenticação, de posse das credenciais, o agente obtinha a referência da sessão correspondente aberta no MACA e chamava a operação `multiple_access_allowed` uma vez, passando uma lista com um número aleatório de pares de objeto/operação, que podia variar entre 1 a 40% do total de 99 pares disponíveis para o simulador utilizar, obtidos previamente do servidor OpenLDAP;
4. Em seguida, o agente chamava a operação `access_allowed` a intervalos que variavam aleatoriamente entre zero a 2 minutos, por um número de vezes nunca inferior a 30. O número máximo de vezes dependia do total de usuários do simulador com sessões abertas. Por exemplo, se o simulador fosse iniciado com 60 “usuários”, então, enquanto não se alcançassem 90% (54) dos “usuários” em atividade, cada agente com sessão aberta ficaria chamando a operação `access_allowed`. Quando este limiar fosse ultrapassado, então aqueles agentes que houvessem realizado pelo menos 30 chamadas poderiam encerrar sua sessão, voltando a executar o passo 1 anterior. Objetiva-se

manter um mínimo (90%) de “usuários” com sessões abertas solicitando autorizações de acesso, a fim de submeter o servidor de segurança a uma carga permanente;

Supondo o serviço de autorização do MACA submetido a uma certa carga, a meta do programa de medição era aferir os tempos médios de resposta que um usuário obtería em solicitações de autenticação e de autorização de acesso. Esse tempo foi calculado como a média dos tempos de resposta obtidos em 100 chamadas sequenciais às operações para autenticação (`authenticate`) e para autorização (`access_allowed`). Porém, a operação `access_allowed` só era chamada para usuários que possuíam papéis disponíveis para ativação, daí o valor de n variar em cada cálculo de média. Cada chamada era realizada para um usuário e para um par de objeto/operação escolhidos aleatoriamente. A carga submetida ao servidor era indicada pelo número médio de solicitações de autenticação e de autorização respondidas por minuto desde que o servidor foi iniciado. Também foi usado como indicador de carga a quantidade de solicitações de autenticação e de autorização de acesso respondidas no minuto anterior ao cálculo do tempo médio de resposta. A carga crescia à medida que o número de “usuários” simultâneos aumentava.

Em cada experimento, a quantidade de “usuários” simultâneos variou de forma progressiva entre 0 e 700, em incrementos de 60 novos “usuários” a cada 5 min, aproximadamente. O valor de 700 baseou-se no número de usuários simultâneos estimado para acessar, em momentos de pico, o PEP do InCor-HC.FMUSP e a aplicação piloto no DATASUS/MS. Para o PEP estimou-se 250 usuários simultâneos, enquanto que, para a aplicação piloto, a estimativa foi de 500. O valor de 700, portanto, inclui uma margem de contingenciamento. Em cada experimento realizado adotou-se a sistemática enumerada a seguir:

1. Reinicializava-se os serviços do OpenLDAP e do serviço de autorização do MACA para zerar as informações estatísticas do sistema, incluindo as médias de autenticações e de autorizações de acesso respondidas por minuto;
2. Na estação de medição, executava-se o programa para calcular os tempos médios de resposta para autenticação e autorização de acesso;
3. Em seguida, numa estação de simulação, executava-se o simulador com 60 novos “usuários”.
4. Quando o servidor de autorização do MACA alcançava a marca de 50 usuários (aproximadamente 5 minutos depois), executava-se o pro-

grama de medição dos tempos médios de resposta indicado no passo 2. Então, o passo 3 era repetido e esperava-se que 50 novos “usuários” iniciassem uma sessão no MACA para medir os tempos de resposta (passo 2) e assim sucessivamente. Quando a marca de 700 usuários simultâneos era alcançada, efetuava-se a última medição e o experimento encerrava-se.

Os experimentos duravam cerca de 1 hora e 20 minutos, havendo, ao final, uma média de 225 “usuários” por estação de simulação. Cada um dos três experimentos foi realizado com uma configuração distinta do servidor de segurança. No experimento 1 (*Tabela 3*), o serviço de autorização do MACA foi configurado para ativar a memória *cache* do usuário (mantendo os resultados das 10 últimas solicitações) e para estabelecer a comunicação usando o protocolo IIOP sobre TLS. No experimento 2 (*Tabela 4*), a memória *cache* do usuário permaneceu ativada, mas a comunicação utilizou o protocolo IIOP sem TLS. Por fim, o experimento 3 (*Tabela 5*) foi realizado com a memória *cache* desativada, mas mantendo a comunicação com o protocolo IIOP sem TLS.

Pode-se notar que os tempos médios de resposta de autenticação (*Gráfico 2*) e de autorização (*Gráfico 3*) quando o *cache* não está ativo (experimento 3) é ligeiramente superior àqueles observados com o *cache* ativo (experimentos 1 e 2). Na autenticação, a diferença acentua-se a partir de 350 “usuários” simultâneos (carga média > 700 solicitações/min no *Gráfico 1*), enquanto na autorização ela ocorre a partir de 150 “usuários” simultâneos (carga média > 380 solicitações/min). A diferença entre o experimento 3 e os experimentos 1 e 2 ocorre mais precocemente para autorização (*Gráfico 3*) porque uma parte das respostas nos exp. 1 e 2 já se beneficiam do *cache*, não contabilizando o tempo de consulta ao servidor LDAP. Por outro lado, como as autenticações sempre consultam o servidor LDAP, não há diferenças significativas entre as respostas nos três experimentos até que se alcance a marca de 350 “usuários”. A partir desse momento, a diferença acentua-se em relação aos experimentos 1 e 2 porque os tempos de resposta às consultas no servidor LDAP começam a ser afetados pela quantidade de acessos simultâneos, visto que o *cache* não é usado. Os tempos de resposta para autenticação nos experimentos 1 e 2 permanecem relativamente estáveis porque, com o *cache*, a quantidade de acessos simultâneos para consulta ao servidor LDAP é menor que no experimento 3. Porém, a partir de 650 “usuários” (carga média de 1250 solicitações/min no *Gráfico 1*), a quantidade de consultas ao servidor LDAP também começa a afetar os tempos de resposta para autenticação (e autorização) nos experimentos 1 e 2.

Tabela 3 – Experimento 1: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com *cache* do usuário ativado e IIOP sobre TLS.

Carga no servidor					Tempos médios de resposta com <i>cache</i> e IIOP sobre TLS (ms)		
Nº de usuários simultâneos	Média de autenticações/min	Autenticações no min. anterior	Média de autorizações/min	Autorizações no min. anterior	Autenticação (N = 100)	Autorização	N
0	0	0	0	0	10	19	91
50	61	13	146	220	13	23	90
100	53	20	244	398	11	23	83
150	50	18	331	546	13	22	85
200	48	26	404	786	15	25	82
250	47	23	488	1130	15	28	89
300	52	48	584	1236	15	25	85
350	52	42	658	1328	14	34	84
400	55	57	750	1622	16	40	83
450	56	49	833	1754	16	50	82
500	64	68	951	2071	18	51	90
550	68	60	1034	1872	19	51	86
600	70	64	1116	2061	20	52	87
650	72	66	1239	2279	19	64	84
700	75	71	1307	2531	40	82	86

Tabela 4 – Experimento 2: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com *cache* do usuário ativado e IIOP sem TLS.

Carga no servidor					Tempos médios de resposta com <i>cache</i> e IIOP sem TLS (ms)		
Nº de usuários simultâneos	Média de autenticações/min	Autenticações no min. anterior	Média de autorizações/min	Autorizações no min. anterior	Autenticação (N = 100)	Autorização	N
0	0	0	0	0	10	20	86
50	52	10	201	360	13	25	81
100	46	18	334	563	15	22	87
150	50	28	425	812	13	24	78
200	59	17	534	731	17	27	83
250	63	27	622	1000	15	27	83
300	65	29	716	1141	15	26	87
350	64	36	840	1219	17	34	85
400	67	44	905	1522	16	32	90
450	67	79	957	2208	16	46	82
500	71	62	1041	1977	17	40	86
550	72	72	1108	2156	18	53	86
600	72	64	1180	2072	18	62	87
650	75	76	1305	2535	41	71	81
700	78	77	1403	2559	63	74	86

Tabela 5 – Experimento 3: resultados dos tempos médios de resposta para solicitações de autenticação e de autorização de acesso com *cache* do usuário desativado e IIOP sem TLS.

Carga no servidor					Tempos médios de resposta sem <i>cache</i> e com IIOP sem TLS (ms)		
Nº de usuários simultâneos	Média de autenticações/min	Autenticações no min. anterior	Média de autorizações/min	Autorizações no min. anterior	Autenticação (N = 100)	Autorização	N
0	0	0	0	0	14	23	91
50	52	22	216	412	15	25	88
100	53	15	298	400	16	26	90
150	56	24	405	659	16	30	88
200	56	29	477	819	15	40	86
250	57	38	575	1206	15	34	84
300	60	40	660	1221	16	38	87
350	63	44	756	1360	43	59	86
400	65	49	836	1458	34	47	86
450	66	56	913	1837	34	59	90
500	67	54	1009	1868	23	68	84
550	68	67	1105	2138	41	63	88
600	70	73	1165	2256	50	72	80
650	71	70	1295	2503	48	80	83
700	75	88	1426	2515	74	90	87

Gráfico 1 – Carga média de solicitações por minuto atendidas pelo servidor de segurança em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).

Gráfico 2 – Tempos médios de resposta (ms) a solicitações de autenticação em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).

Gráfico 3 – Tempos médios de resposta (ms) a solicitações de autorização em função do número de usuários simultâneos para os experimentos 1 (com cache e IIOP sobre TLS), 2 (com cache e IIOP sem TLS) e 3 (sem cache e IIOP sem TLS).

Uma outra observação que se pode fazer é que os tempos médios de resposta, tanto para autenticação, quanto para autorização, aumentam numa proporção inferior ao aumento da carga no sistema, mesmo no pior caso. Por exemplo, no experimento 3 (*Tabela 5* e *Gráfico 3*), o tempo médio de resposta para uma autorização com 700 “usuários” simultâneos foi 3,6 ($90 \div 25$) vezes maior que o tempo de respos-

ta para 50 “usuários”. Enquanto isso, a carga média aumentou 5,6 ($\{1.426 + 75\} \div \{216 + 52\}$) vezes no mesmo intervalo (*Tabela 5* e *Gráfico 1*). Para autenticação, também no experimento 3 (*Tabela 5* e *Gráfico 2*) e no mesmo intervalo, o tempo médio de resposta aumentou 4,9 vezes ($74 \div 14$). Os tempos médios de resposta absolutos nos três experimentos, abaixo dos 100 ms, também são um bom resultado, mesmo com o sistema submetendo-se a uma carga da ordem de 2.500 solicitações/min durante a medição (valor estimado com base na carga do minuto anterior à medição dos tempos de resposta).

Por fim, pode-se notar que a utilização do protocolo IIOP sobre TLS no experimento 1 (*Gráfico 2* e *Gráfico 3*) não resultou em tempos de resposta significativamente diferentes daqueles obtidos no experimento 2 (IIOP sem TLS). Isso porque o ORB da arquitetura CORBA gerencia as conexões TCP/IP com eficiência, mantendo *pools* de conexões previamente abertas, tanto no cliente, quanto no servidor. O elevado custo de abertura de conexões TCP/IP sobre TLS, portanto, fica restrito a conexão realizada na primeira chamada a operação *authenticate*. As chamadas de operações subsequentes reutilizam conexões já abertas a um custo relativo mais baixo. A carga média ligeiramente inferior no experimento 1 (*Gráfico 1*) também pode explicar esse fato. Mesmo assim, os tempos médios de resposta inferiores a 100 ms indicam que utilizar IIOP sobre TLS é viável para o nível de carga a que o serviço de autorização do MACA foi submetido.

4.3.2 Aplicação do MACA no PEP do InCor-HC.FMUSP

A implementação dos serviços de autenticação e de autorização do MACA, atualmente na versão 3.2, vem sendo usada para controlar o acesso ao PEP no InCor – HC.FMUSP, em ambiente de produção, desde julho de 2001. O InCor – HC.FMUSP é um centro de referência em cardiologia, reconhecido nacional e internacionalmente, com mais de 20 anos de experiência no desenvolvimento de sistemas de informação para área de saúde. No momento, conta com cerca de 600 leitos, 2.600 funcionários e 300 profissionais temporários (médicos residentes e estagiários pós-graduandos, como médicos, nutricionistas e psicólogos) que auxiliam a realizar por mês, aproximadamente, 300 cirurgias, 11.600 procedimentos de diagnóstico por imagem e 100.000 exames laboratoriais, dentre outras atividades.

O servidor de segurança do MACA recebe, por mês, uma média de 42.768 solicitações de autenticação de usuário e 442.368 solicitações de autorização de acesso. O servidor OpenLDAP da instituição mantém um total de 2.036 contas de usuá-

rios e 78% delas têm uma média de 1,3 papéis associados. Há 66 papéis configurados e, desse total, 86% possuem uma média de 30,3 usuários vinculados. Em cada papel há, em média, 3,1 autorizações diretamente vinculadas e 26,5 autorizações herdadas. Das 205 autorizações definidas, 79% são positivas, 3% são negativas, 18% possuem regras e 99% são fracas (1% fortes). As autorizações estabelecem uma política de acesso para 83 objetos com diferentes operações associadas. Do total, 47 objetos representam componentes do PEP do InCor – HC.FMUSP e o restante denota as funcionalidades administrativas do MACA.

A versão 2.0 do servidor de segurança, a primeira empregada em ambiente de produção, foi inicialmente utilizada pelos componentes do PEP apenas para fins de autenticação de usuário, sem recorrer ao serviço de autorização do MACA, que, então, não possuía autorizações contextuais, mas apenas autorizações estáticas positivas e negativas. Os únicos clientes do serviço de autorização eram as próprias ferramentas administrativas do MACA. Inicialmente, realizou-se um cadastramento de todos os usuários do PEP na instituição mediante requerimento formal (ver Anexo B na p. 176) do interessado, por escrito, aprovado pelo diretor da unidade de lotação. O MACA AD *Web*, usado na criação das contas, foi integrado à base de dados do recursos humanos do InCor – HC.FMUSP para validar os dados funcionais fornecidos pelo usuário, particularmente o número de matrícula, o cargo exercido e a área meio⁹ de lotação.

O cargo exercido determinava o papel vinculado ao usuário durante o cadastramento. Isso porque a hierarquia de papéis (*Figura 23*) foi originalmente concebida com base nos 409 cargos existentes na instituição. A estrutura linear dos cargos foi estruturada numa hierarquia de papéis que levou em conta as relações de responsabilidade e de autoridade existentes, visando uma hierarquia inicial simples, mas passível de evolução. Buscou-se a definição de papéis mais genéricos, que pudessem ser especializados em papéis de maior responsabilidade em função das necessidades da organização, de uma maior experiência no uso do MACA e da tecnologia do CABP. Por exemplo, no primeiro nível da árvore (*Figura 23*), houve a separação entre os papéis dos profissionais do InCor – HC.FMUSP daqueles papéis dos usuários que não exerciam uma atividade de responsabilidade institucional, como o próprio paciente e os estudantes. Abaixo dos profissionais, houve a distinção entre os usuários que exerciam funções de assistência direta ao paciente (papéis *Médico* e *Paramédico*) e aqueles que executavam funções de atividades meio, em geral, administrativas

⁹ No InCor – HC.FMUSP adota-se a nomenclatura “área meio” no lugar de unidade organizacional.

Figura 23 – Hierarquia dos papéis não administrativos atualmente definidos no MACA para o InCor-HC.FMUSP.

(papel *Administrador Executivo*). O mesmo enfoque foi adotado até definir-se papéis *concretos*, descrevendo funções reais na instituição, aos quais usuários podiam ser diretamente vinculados.

Os 409 cargos reduziram-se aos atuais 66 papéis por dois motivos principais. Primeiro, muitos dos cargos existentes (*e.g.*, vigia, porteiro, roupeiro) relacionam-se a funções que não necessitam acesso a sistemas informatizados, em especial ao PEP. Segundo, há um número considerável de cargos que denotam as mesmas funções, mas que são diferenciados em “júnior”, “master”, “pleno” e “sênior” em função de classificação salarial. Tais cargos foram mapeados para um mesmo papel na hierarquia. Note-se que esse não é o caso de cargos numa mesma profissão (*e.g.*, enfermagem, farmácia, nutrição), em que há real distinção de autoridade e responsabilidade. Por exemplo, as relações de subordinação entre o auxiliar de enfermagem e o enfermeiro, o auxiliar de farmácia e o farmacêutico, o faturista e o analista de faturamento estão expressas na hierarquia (*Figura 23*). Cabe ressaltar que a atual hierarquia de papéis do InCor – HC.FMUSP não está concluída, tampouco é completa. A construção de uma hierarquia de papéis é um processo contínuo que evolui de acordo com as necessidades da organização e com as circunstâncias encontradas no ambiente em que se insere.

A entrada em funcionamento, em fevereiro de 2002, da versão 3.0 do servidor de segurança, suportando as autorizações contextuais do MACA, permitiu que os componentes do PEP utilizassem o serviço de autorização contextual. Um conjunto de 20 aplicações legadas, implementadas no ambiente de programação Magic (Magic Software Enterprises, 2000), foram modificadas para obter permissões de acesso do serviço de autorização do MACA, em adição à autenticação de usuários. As aplicações para prescrição médica, programação cirúrgica, registro de procedimentos e diário de enfermagem são alguns exemplos do conjunto. Porém, o acesso é permitido ou proibido a cada aplicação como um todo. O controle de acesso detalhado as suas funções internas ainda é realizado de forma independente nas aplicações, sem utilizar o serviço de autorização do MACA. De qualquer forma, o bloqueio ou a liberação do acesso à esses componentes do PEP são estabelecidos de forma unificada, utilizando-se de um ponto de controle único, baseado nos papéis desempenhados pelos usuários na instituição.

O serviço de autorização contextual do MACA também passou a ser usado no controle do acesso às funcionalidades do PEP disponibilizadas na intranet do InCor – HC.FMUSP via *web browser*. O PEP na *web* foi implementado em Ja-

va/CORBA, na plataforma Java 2 SE, usando JSP (Sun Microsystems, 2003b) e o J-ORB (Brose & Noffke, 2002). Ele integra informações clínicas provenientes das diversas aplicações que compõem o PEP (legado em Magic) e imagens médicas dos pacientes gerenciadas pelo PACS (*Picture Archiving and Communication System*) da instituição. As informações clínicas e as imagens médicas são recuperadas para integração num serviço Java/CORBA (Fiales *et al.*, 2001; Moreno *et al.*, 2002) baseado na especificação do CORBA *Healthcare*. Um visualizador de alto desempenho (Santos & Furuie, 2004), também desenvolvido em Java no InCor – HC.FMUSP, exibe para os usuários do PEP, numa *Applet* Java, os exames com imagens e os respectivos laudos, recuperados dos serviços CORBA.

O controle de acesso às informações e as imagens no PEP na *web* é realizado de forma detalhada e interna nos serviços CORBA implementados em Java. Ou seja, ele é imposto de maneira transparente para o visualizador de imagens e para os *scripts* em JSP que criam páginas *web* dinamicamente, que precisam, tão somente, de uma credencial válida do usuário, repassada aos serviços CORBA. O PEP na *web* atual é usado exclusivamente para consulta, e nele pode-se controlar o acesso aos dados demográficos do paciente, ao conteúdo de laudos, a documentos digitalizados, a imagens médicas, dentre outros tipos de informação. Uma nova geração de serviços disponibilizados para o PEP na *web* está em fase final de desenvolvimento no InCor – HC.FMUSP para substituir as aplicações legadas existentes em Magic. Em breve, o PEP na *web* oferecerá não apenas acesso para consulta, mas também para criação de informações por meio dos novos serviços desenvolvidos (*e. g.* prescrição médica, laudos, registro de procedimentos). A implementação desses serviços já embute um controle de acesso detalhado, permitindo o estabelecimento de uma política de acesso unificada para o PEP da instituição, baseada apenas nos serviços de autenticação de usuários e de autorização do MACA.

A elaboração e a efetivação da política de acesso para os serviços recém desenvolvidos para o PEP na *web* proporcionarão novos desafios que ajudarão a aprimorar o MACA, a sua implementação, assim como a conhecer melhor o processo de utilização/concepção de contextos incorporados à tecnologia de controle de acesso baseado em papéis. Não obstante, a experiência acumulada na definição e na implementação da política de acesso do atual PEP (aplicações Magic e PEP para consulta na *web*) resultou em importantes lições, particularmente em relação a hierarquia de papéis construída (*Figura 23*), e que serão úteis no estabelecimento da política dos novos serviços.

O primeiro aspecto a destacar é a importância da distinção entre os papéis estritamente relacionados à assistência ao paciente (atividade fim), daqueles que não são diretamente relacionados (atividade meio). Para os primeiros, a definição da política de acesso é mais simples porque as funções que executam, a autoridade e a responsabilidade de cada um deles são bem conhecidas, são tradicionais no cuidado ao paciente, em geral, determinadas por normas legais. É o caso dos papéis descendentes de *Médico* e *Paramédico* (Figura 23). Por exemplo, sabe-se que a emissão de prescrições é uma prerrogativa dos médicos (papel *Médico* e seus descendentes) e que, em geral, os paramédicos podem apenas consultá-las, com exceção dos enfermeiros e dos farmacêuticos. Esses, embora sejam impedidos de prescrever, podem executar operações de alteração, como registrar o horário efetivo em que a prescrição foi executada, ou ordenar a entrega de uma determinada quantidade da droga prescrita para um paciente, respectivamente.

A definição da política de acesso é mais problemática para os papéis que oferecem apoio administrativo, não relacionados diretamente às funções assistenciais. Esse é o caso de alguns dos descendentes de *Técnico Administrativo* (Figura 23). Papéis como *Auxiliar Administrativo*, *Escrivão* e *Secretário* congregam uma quantidade razoável de usuários no InCor – HC.FMUSP, cerca de 175, cujas funções variam de acordo com a unidade organizacional onde estão lotados. Por exemplo, apenas os auxiliares administrativos da área meio Coordenação de Enfermagem precisam ter acesso ao diário de enfermagem. Já as funções de leitura aos agendamentos de consultas e cirurgias só são permitidos aos auxiliares administrativos lotados em áreas meio assistenciais, como o Serviço de Hipertensão —é importante notar que a área meio do usuário é assistencial, mas não as funções que ele exerce. A solução foi incluir para esses papéis autorizações com regras contextuais baseadas no nome ou no tipo (assistencial/não assistencial) da área meio de um usuário. A necessidade identificada no InCor – HC.FMUSP de utilizar a estrutura organizacional para estabelecer políticas de acesso reforça a importância das autorizações contextuais do MACA. Sem elas, a solução seria mesclar a estrutura organizacional com a estrutura de papéis, com a conseqüente criação de um número excessivo de papéis (ver último parágrafo da seção 3.5 na p. 99). Já a definição das políticas para papéis não assistenciais, mas que têm uma função bem conhecida, como auxiliares e analistas de registros de saúde, ocorreu sem dificuldades.

A administração da política de acesso ao PEP ainda é realizada por usuários do Serviço de Informática do InCor – HC.FMUSP com os papéis administrativos

Administrador de Contas e *Administrador de Autorizações*. O primeiro é responsável exclusivamente pelo gerenciamento (criar, alterar, consultar e excluir) das contas de usuários. Entretanto, fica terminantemente proibido de criar, vincular/desvincular ou remover papéis. A idéia é evitar que um administrador de contas mal intencionado possa criar contas fraudulentas com privilégios excessivos ou atribuir papéis para si mesmo. Também é impedido de gerenciar recursos, papéis e autorizações. O segundo papel, *Administrador de Autorização*, pode gerenciar recursos, papéis e autorizações, mas está terminantemente proibido de gerenciar contas de usuários, exceto para consulta. Porém, é sua responsabilidade atribuir/remover papéis, desde que o usuário em questão não seja ele mesmo. O objetivo foi distribuir as responsabilidades de criação de contas e de atribuição de privilégios para usuários distintos. A própria política de acesso do MACA também não pode ser alterada pelo administrador de autorização.

A meta é, no futuro, transferir a responsabilidade pela gerência de contas e pela vinculação/desvinculação de papéis à área de recursos humanos da instituição e à comissão de ensino e pesquisa. Para isso, será criado um novo papel administrativo, o *Designador de Papéis*, que assumirá as funções de vinculação/desvinculação de papéis do *Administrador de Autorização*. Os administradores da área de recursos humanos serão responsáveis pelas contas dos usuários que são empregados do InCor – HC.FMUSP, enquanto aqueles da comissão de ensino e pesquisa serão responsáveis pelas contas de estagiários e estudantes, num esquema similar àquele adotado no exemplo visto na seção 3.5. A gerência de autorizações e papéis continuará a ser realizada de forma centralizada na área de informática do InCor – HC.FMUSP pelos administradores de autorização.

4.3.3 Aplicação do MACA no DATASUS/MS

A versão 3.2 da implementação do MACA foi considerada pelo DATASUS/MS para possível aplicação no âmbito do sistema nacional de informações em saúde. O DATASUS/MS é o departamento do Ministério da Saúde responsável por prover “*os órgãos do SUS (Sistema Único de Saúde) de sistemas de informação e suporte de informática necessários ao processo de planejamento, operação e controle do Sistema Único de Saúde, através da manutenção de bases de dados nacionais, apoio e consultoria na implantação de sistemas e coordenação das atividades de informática inerentes ao funcionamento integrado dos mesmos*” (DATASUS, 2003).

O DATASUS/MS decidiu inicialmente utilizar os serviços de autenticação de usuários e de autorização de acesso do MACA no sistema agente do cadastro na-

cional de estabelecimento de saúde (CNES). O CNES mantém informações cadastrais de todos os estabelecimentos de saúde do País que prestam serviços assistenciais pelo Sistema Único de Saúde. O sistema agente CNES permitiria que os responsáveis por cada estabelecimento pudessem, via *web*, obter suas informações cadastrais, bem como recuperar as tabelas e meta informações que compõem o cadastro em formatos padronizados pelo DATASUS/MS para uso em seus próprios sistemas de informação. O sistema agente CNES deveria também permitir o acesso aos cadastros para os gestores do SUS nas esferas federal, estadual e municipal.

O objetivo do DATASUS/MS era utilizar o MACA num projeto piloto a fim de se familiarizar com a tecnologia para aplicação posterior em outros sistemas. Atendendo os requisitos definidos pelo DATASUS/MS para o sistema agente CNES, o MACA foi configurado com uma política de acesso para o cadastro que considerava a hierarquia existente entre as esferas federal, estadual, municipal e de estabelecimento de saúde. Foram definidos dois papéis principais: *Operador* e *Gestor*. Operadores eram usuários padrão com acesso para consulta às informações do CNES, enquanto gestores tinham os mesmos privilégios dos operadores, podendo, em adição, consultar informações confidenciais de um estabelecimento. Porém, a abrangência geográfica dos cadastros que eles poderiam consultar dependeria da esfera em que eles estariam situados. O cadastro de cada usuário possuiria um atributo com o nível de abrangência e, nos casos dos níveis de estado, município e estabelecimento, haveria um outro atributo indicando os respectivos identificadores desses. Portanto, cada usuário estaria situado numa esfera administrativa específica, de modo similar a lotação numa unidade organizacional, como visto no exemplo da seção 3.5. Um usuário situado em uma determinada esfera só poderia consultar os cadastros dessa esfera. Por exemplo, um gestor situado na cidade de “Paraty” só teria acesso aos cadastros de estabelecimentos localizados nesse município, incluindo as informações confidenciais. Por outro lado, um operador do InCor – HC.FMUSP só poderia consultar o cadastro do próprio hospital, mas sem acesso às informações confidenciais. Já um gestor de nível federal poderia acessar todos os cadastros.

Também foram criados papéis administrativos descentralizando hierarquicamente a gerência de contas pelas diversas esferas em que os usuários do sistema agente CNES se situavam, em moldes similares ao exemplo ilustrado na *Figura 6*. Por exemplo, um administrador de contas do InCor – HC.FMUSP só poderia gerenciar as contas de usuários da sua instituição, enquanto um administrador de contas de nível federal poderia gerenciar todas as contas, mas o de nível estadual só gerenciaria as contas dos usuários de seu estado.

O DATASUS/MS recebeu uma versão do MACA configurada com essa política para utilização pelo sistema agente CNES. Ele previa um total de 20.000 contas de usuários de todo o País e um máximo de 500 usuários simultâneos. Requeria tempos médios máximos de resposta, em momentos de pico, de 500 e 300 ms para solicitações de autenticação e de autorização de acesso, respectivamente. Entretanto, o desenvolvimento do sistema agente CNES foi suspenso por prazo indeterminado e examina-se um outro projeto piloto para utilização do MACA.

4.4 Considerações finais

A arquitetura em que a implementação do MACA foi integrada, baseada em padrões de processamento aberto e distribuído, visou satisfazer dois importantes requisitos para o controle de acesso do PEP em grandes organizações de saúde: 1) a possibilidade dos diferentes componentes do PEP, muitas vezes legados, utilizarem os serviços de autenticação de usuários e de autorização de acesso a partir de plataformas heterogêneas; 2) a capacidade de administrar e de impor políticas de acesso e de autenticação de modo unificado e coerente, a despeito da natureza distribuída e heterogênea do PEP. A atual versão da implementação do MACA ainda apresenta um bom desempenho para as demandas de acesso estimadas, com a vantagem de basear-se em componentes de software sem custos de licenciamento. Por fim, a utilização rotineira do MACA no controle de acesso ao PEP do InCor-HC.FMUSP evidenciam a exeqüibilidade do modelo, da sua implementação e da sua aplicação prática em casos reais.

“A tarefa não é contemplar o que ninguém ainda contemplou, mas meditar como ninguém ainda meditou sobre o que todo mundo tem diante dos olhos.”

Arthur Schopenhauer, 1818

O MACA é discutido sob três perspectivas. A primeira, procura defender a sua validade em atender o problema do controle de acesso ao PEP (seção 5.1). A segunda perspectiva (seção 5.2) compara o MACA com os trabalhos relacionados na seção 2.5, levando-se em conta as características indicadas na *Tabela 1* (p. 38). Por fim, são realizadas considerações acerca das limitações do modelo e de possíveis extensões visando o seu aperfeiçoamento (seção 5.3).

5.1 O MACA e o controle de acesso ao PEP

Observou-se, na seção 2.4 (p. 20), que o modelo de referência para o CABP proposto pelo NIST possui características gerais que favorecem a solução do problema de controlar o acesso ao PEP, particularmente em organizações de saúde de grande porte, que realizam atendimentos em níveis secundário e terciário. Portanto, cabe agora responder quais são as contribuições das características específicas do MACA para a solução do problema de controlar o acesso ao prontuário eletrônico do paciente.

5.1.1 A contribuição das autorizações contextuais do MACA

A possibilidade de incluir regras relacionando informações contextuais nas autorizações do modelo de referência do CABP confere flexibilidade e poder expressivo para o MACA definir elaboradas políticas de acesso. A flexibilidade decorre da utilização

das regras de autorização, que permitem um controle de acesso preciso, de acordo com o direito e a necessidade do usuário realizar uma tarefa, mas sem custos gerenciais excessivos. Isso porque lógicas de controle de acesso mais complexas, tradicionalmente embutidas nas aplicações protegidas, podem agora ficar isoladas das aplicações, embora mantendo o contexto necessário para conceder ou não uma autorização. Assim, mudanças na lógica de autorização não implicam em mudanças nas aplicações. Por outro lado, a capacidade de se criar contextos com alto nível de abstração concede ao MACA poder para expressar formalmente políticas de acesso organizacionais comumente definidas em linguagem natural.

Ademais, a implementação de contextos utilizando interfaces CORBA permite relacionar informações contextuais provenientes de sistemas legados, existentes em plataformas distribuídas e heterogêneas, uma realidade em organizações de saúde reais, com tradição na utilização de PEPs. Portanto, as autorizações contextuais do MACA são uma importante contribuição para implementação de políticas de acesso para o PEP e, também, para as políticas administrativas dessas organizações.

5.1.1.1 Políticas de acesso para o PEP

O objetivo principal de uma política de acesso para o PEP é preservar a privacidade do paciente, só liberando para terceiros o acesso a informações e serviços estritamente necessários para prover cuidados médicos. O modo como esse objetivo é alcançado depende das características organizacionais, da determinação do paciente e de resoluções legais, implicando, portanto, em políticas de acesso *ad hoc*, diferenciadas em cada organização. Por exemplo, a política especificada na *Figura 5* (seção 3.4, p. 80) ilustra uma possível política de acesso para o PEP. Considera o contexto e as circunstâncias de um hospital específico visando satisfazer o objetivo principal anterior. Entretanto, as autorizações contextuais do MACA permitem a definição de outras políticas, que se ajustam aos diferentes ambientes e culturas encontradas nas organizações de saúde.

Dependendo da organização, varia o conceito de lugar onde o usuário acessa uma informação. O lugar pode representar uma localização “virtual”, como a lotação do usuário no organograma institucional. O “Serviço de Hemodinâmica” e o “Serviço de Neuropediatria” (*Figura 7*, p. 94) são exemplos de localizações num hospital. Uma política utilizando contextos apropriados poderia estabelecer que os auxiliares administrativos dessas áreas (que prestam assistência médica direta ao paciente) têm mais privilégios de acesso ao PEP que aqueles lotados na “Unidade Central de Atendimento ao Cliente” (setor não assistencial), por exemplo. Em outros casos, como o

ilustrado na *Figura 5*, considera-se o lugar fisicamente, identificado por um domínio ou endereço de rede. Note-se que, com a utilização de dispositivos móveis, a localização de um usuário poderia ser efetuada dinamicamente, com os privilégios de acesso variando de acordo com sua localização efetiva no hospital. As autorizações contextuais do MACA também permitiriam especificar políticas combinando as localizações “virtuais” e físicas para liberar acesso ao PEP.

Uma outra política poderia utilizar o status do fluxo de trabalho (*workflow*) dos processos organizacionais de um hospital para incrementar a segurança de acesso ao PEP. Por exemplo, o acesso para visualização da imagem de um exame só seria autorizado após a assinatura digital do respectivo laudo pelo médico responsável. Malamateniou *et al.* (1998) propuseram uma solução para o controle de acesso baseada no fluxo de trabalho dos processos organizacionais relacionados ao PEP. O modelo de autorização contextual do MACA é capaz de suportar uma solução similar. Um contexto de fluxo de trabalho poderia ser implementado para obter informações sobre o status do andamento dos processos organizacionais. Os processos seriam representados por recursos (objetos e operações), para os quais se criariam autorizações com regras estabelecendo as condições de acesso baseadas no contexto de fluxo de trabalho.

Existem estabelecimentos de saúde que requerem outros tipos de relacionamentos entre paciente e usuário para basear sua política de acesso, além dos mostrados no exemplo da *Figura 5*. É o caso dos estabelecimentos que mantêm informações sobre a saúde mental do paciente. As políticas de acesso a essas informações são fortemente dependentes dos relacionamentos entre o paciente e o seu psiquiatra, o seu médico particular e os membros da equipe multiprofissional que auxiliam na assistência médica (Varadharajan & Calvelli, 1996). Ademais, em geral, as políticas proibem o paciente de acessar os próprios registros psiquiátricos, embora ele tenha o direito de conceder ou negar esse acesso a terceiros, parentes, por exemplo (Varadharajan & Calvelli, 1996). As autorizações contextuais do MACA são capazes de expressar tais políticas, desde que haja os contextos apropriados denotando esses relacionamentos.

A disponibilidade de políticas de privacidade também é uma meta importante a ser alcançada pelas organizações que dispõem de um PEP. Um exemplo é a política de privacidade proposta por Schoenberg & Safran (2000), onde o paciente pode determinar o nível de sigilo de cada tipo de dado do seu prontuário eletrônico. O modelo de autorização do MACA é capaz de implementar uma solução similar com

a criação de um contexto de privacidade. O contexto forneceria os níveis de sigilo atribuídos pelo paciente a tipos de dados do PEP, como status de HIV ou contagem de células CD4. Cada papel teria um nível de sigilo previamente estabelecido pelos formuladores da política. Regras de autorização especificariam que um papel tem acesso a um tipo de dado sempre que o seu nível de sigilo fosse igual ou superior ao nível estabelecido pelo paciente para esse dado. Os tipos de dados seriam representados como objetos e operações. Um sistema seria fornecido para os pacientes especificarem suas preferências de liberação de acesso.

5.1.1.2 Políticas administrativas

A capacidade de executar políticas administrativas elaboradas é indispensável para viabilizar a utilização efetiva do MACA em grandes organizações de saúde. Conforme ilustrado na seção 3.5, o MACA pode ser configurado para combinar diferentes políticas administrativas a fim de atender demandas comuns às organizações, bem como suas idiossincrasias. Suporta desde as políticas amplamente descentralizadas e autônomas, até as políticas centralizadas, ou um misto das duas. Por exemplo, permite a administração autônoma e descentralizada, nas unidades organizacionais, das relações usuário-papel, mas combinada a uma gerência unificada da hierarquia de papéis e das relações papel-autorização, conforme ilustrado na política da *Figura 6* (p. 93). Porém, o que o MACA traz de inovador é a capacidade de executar diversas políticas administrativas para o CABP com a criação de contextos e a configuração de regras de autorização apropriadas. Ou seja, as possíveis políticas administrativas para o MACA não estão definidas *a priori*, tampouco estão embutidas no próprio modelo. Com isso, as características mais relevantes dos principais modelos administrativos recentemente propostos para o CABP podem ser suportadas.

É o caso dos modelos administrativos para o CABP baseados em hierarquias organizacionais propostos por Kern *et al.* (2003), Oh & Sandhu (2002) e Perwaiz & Sommerville (2001). Eles se caracterizam por separar a autoridade administrativa —das unidades de uma estrutura organizacional— dos papéis aos quais usuários ou objetos estão associados. A independência entre a estrutura organizacional e a hierarquia de papéis resolveu um dos maiores problemas encontrados no modelo seminal para administração do CABP, o ARBAC97 (Sandhu *et al.*, 1999), que é o forte acoplamento entre a autoridade administrativa e as associações de usuários e objetos (via autorizações) com papéis (Kern *et al.*, 2003; Oh & Sandhu, 2002). A criação do contexto `organogramaCtx` com as funções de escopo (*e. g.*,

LotaçãoAbaixoOuIgualA) e a definição das regras de autorização, no exemplo da *Figura 6*, permitiu ao MACA distribuir privilégios administrativos de acordo com a pertinência de um usuário a uma unidade de uma estrutura hierárquica, mas com independência da hierarquia de papéis. Outros contextos e regras habilitariam o MACA a utilizar estruturas organizacionais distintas para determinar a autoridade administrativa sobre usuários e objetos. Por exemplo, os membros do projeto de uma organização, possivelmente pertencente a papéis e a unidades organizacionais distintas, poderiam ter privilégios administrativos apenas sobre as autorizações associadas aos objetos desenvolvidos pelo projeto. Já a abrangência da autoridade sobre usuários e seus relacionamentos seria determinada pelo organograma organizacional.

O MACA também pode ser configurado para suportar a gerência da hierarquia de papéis com base em *escopos administrativos*, um conceito introduzido por Crampton & Loizou (2002). Um escopo administrativo define um subconjunto de papéis hierarquicamente relacionados sobre os quais um papel administrativo tem autoridade. Por exemplo, poderia-se especificar que um papel administrativo (*e. g.*, *Administrador de Papéis Assistenciais*) somente gerenciaria (criar, ler, alterar, excluir) os papéis que fossem exclusivamente descendentes (direta ou indiretamente) do papel *Profissional de Saúde* (v. *Figura 3* (a), p. 31). Como o papel *Diretor Clínico* também é descendente de *Diretor*, que não descende de *Profissional de Saúde*, então o escopo administrativo do papel *Administrador de Papéis Assistenciais* ficaria restrito aos papéis *Profissional de Saúde*, *Médico*, *Paramédico* e *Auxiliar de Enfermagem*. Caso a relação de herança *Diretor Clínico* $\succeq$ *Diretor* fosse removida por um outro administrador, o escopo administrativo anterior passaria dinamicamente a incluí-lo. Esse dinamismo dos escopos administrativos pode ser definido no MACA com regras utilizando as funções de escopo sobre papéis (*e. g.*, *PapelAbaixoOuIgualA*, *PapelAcimaOuIgualA*) e outras definições disponíveis no contexto *macaCtx*.

5.1.2 A contribuição da separação de responsabilidades independente da ativação automática de papéis

A facilidade de uso é um atributo de qualidade essencial para utilização efetiva de um sistema de PEP por usuários finais. O PEP deve fornecer mecanismos e interfaces simples e diretas, orientadas aos problemas atuais e pregressos do paciente (Marin, 2003). Os serviços de segurança favorecem esse atributo na medida em que não difi-

cultem o uso do PEP e mantenham a sua eficácia. Nesse aspecto, o MACA oferece uma contribuição inovadora quando torna a ativação automática de papéis independente da separação de responsabilidades. No MACA, dois ou mais papéis com restrições de separação de responsabilidades, dinâmica ou estática, podem estar ativos simultaneamente. Porém, as eventuais ações que representam risco de cometimento de erros ou fraudes, motivadoras da separação de responsabilidades, ficam indisponíveis. Assim, a ativação automática de papéis torna-se independente da SR (estática ou dinâmica) e vice-versa. Conseqüentemente, os sistemas de PEP podem utilizar o CABP do MACA de forma transparente, não ostensiva para o usuário final, embora suportando o conceito de separação de responsabilidades.

Os modelos para CABP levantados na literatura, incluindo os relacionados na *Tabela 1* (p. 40), requerem a ativação explícita de papéis pelo usuário. A exceção é o mecanismo de ativação automática de papéis para o CABP do NIST proposto por Obelheiro *et al.* (2001). Entretanto, tal mecanismo apresenta desvantagens em relação ao mecanismo do MACA por ser dependente da separação de responsabilidades dinâmica preconizada no modelo do NIST. Segundo os autores, um papel não pode ser ativado (automaticamente) se tiver restrições de SRD em relação a quaisquer papéis ativos, a fim de não violar as restrições de SRD existentes. No caso, se o usuário precisasse desempenhar as funções do papel que não pode ser ativado, ele teria que desativar explicitamente os papéis que têm restrições de SRE para poder ativá-lo, também de forma explícita. A ativação/desativação automática é dificultada porque, em geral, o mecanismo de ativação não tem como saber qual é o papel que o usuário quer ativar. Por exemplo, suponha-se uma restrição de SRD entre os papéis *Médico Assistente* e *Médico Auditor*; tal como estabelecida pelo CABP do NIST (v. p. 35), e que eles autorizam o acesso a pelo menos uma operação/objeto em comum (*e. g.*, executar o “Serviço de Prescrição Médica”). Quando um usuário detendo ambos os papéis tenta executar a operação em comum, o mecanismo de ativação automática deve escolher apenas um dos papéis para ativar, que pode ser aquele que o usuário não quer desempenhar. Ou seja, o usuário pode, no futuro, desejar executar uma operação que só é autorizada pelo papel que não foi ativado e, nesse caso, ou o acesso é negado, ou o usuário deve intervir no processo de ativação/desativação dos papéis. E, acresça-se, ele deve conhecer os conceitos da SRD e do processo de ativação de papéis do CABP. Portanto, a necessidade de intervenção do usuário final no processo de ativação de papéis é um fator que desfavorece o uso do CABP em sistemas de PEP, mas que não está presente no MACA.

5.2 O MACA e os trabalhos relacionados

As subseções seguintes comparam o MACA com os modelos para CABP revisados na seção 2.5 (p. 38), tomando por base as características relacionadas na *Tabela 1*.

5.2.1 CABP hierárquico

Em relação aos modelos revisados na seção 2.5 (*Tabela 1*), o MACA é o único que adota a hierarquia de papéis restrita à estrutura de árvore invertida. Duas razões principais justificam essa decisão. Primeiro, hierarquias estruturadas em árvores podem ser diretamente representadas no serviço de diretórios LDAP (Yeong *et al.*, 1995) (v. seção 4.2.1, p. 109), que se caracteriza por ter implementações eficientes (*e. g.*, o OpenLDAP (OpenLDAP Foundation, 2003)), otimizadas para consultas intensivas. Segundo, árvores proporcionam interfaces com o usuário mais fáceis de usar e mais familiares do que hierarquias gerais, que se estruturam como grafos acíclicos ordenados. A visualização de grafos com centenas ou milhares de nós é trabalhosa, sua complexidade ultrapassa nossa capacidade em estabelecer padrões de relacionamentos, além de exigir avançadas técnicas e ferramentas de software (Gordin, 2002; Larkin & Simon, 1987).

Embora a hierarquia geral de papéis seja útil em representar funções como as do *Diretor Clínico* (*Figura 3* (a), p. 31), a hierarquia em árvore invertida é capaz de representar tais funções pela introdução de papéis adicionais. Por exemplo, o papel *Diretor Clínico* poderia ser dividido em dois papéis, um tendo *Diretor* como único ascendente e o outro tendo *Médico* como único ascendente. O usuário que exercesse a função de diretor clínico teria estes dois papéis associados.

Os modelos TCM (seção 2.5.4, p. 44) e DRIVE-RBAC (seção 2.5.5, p. 45) estendem o conceito de hierarquia de papéis para incluir usuários e categorias de usuários, por exemplo, para representar unidades organizacionais. O MACA se distingue por ser capaz de incorporar essas características na definição de políticas de acesso (v. exemplo da seção 3.5, p. 92), mas de forma independente da hierarquia de papéis e sem modificar conceitos fundamentais (usuários e papéis) do modelo de referência do CABP do NIST.

Por outro lado, a hierarquia restrita do MACA oferece vantagens em relação aos modelos que não hierarquizam os papéis. Bacon & Moody (2002) justificam, para o modelo OASIS (seção 2.5.8, p. 48), que as hierarquias são impraticáveis em ambientes altamente distribuídos, que não têm administração global e centralizada. Po-

rém, no âmbito de cada domínio administrativo, a hierarquia de papéis é viável e vantajosa. Nas interações interdomínios, a estrutura hierárquica não precisa ser considerada, o importante é haver uma padronização dos papéis chave. Por exemplo, suponha-se os domínios administrativos distintos dos hospitais A e B e que há um padrão estabelecendo uma designação para os papéis de médico, suas especialidades e áreas de atuação. Considerando que o hospital A tem a hierarquia de papéis da *Figura 3 (a)*, um usuário desse hospital poderia receber uma designação (conceito do modelo OASIS) para o papel *Médico* e apresentá-la ao hospital B a fim de acessar o prontuário de um paciente internado em A, que foi transferido de B. Observe-se que o hospital B poderia ter uma hierarquia de papéis e uma política de acesso distintas de A. O indispensável é ter o papel padrão *Médico* também definido.

5.2.2 CABP restrito

A separação de responsabilidades disponível no MACA se diferencia daquelas encontradas nos modelos revisados (*Tabela 1*) e do próprio modelo de referência para CABP do NIST por basear-se nas operações e objetos das aplicações e nos conflitos existentes entre autorizações (v. seção 3.4.3.4, p. 91). O MACA adota esse enfoque porque a separação de responsabilidades é uma política inerentemente orientada a aplicação, que é projetada considerando-se a partição de tarefas sensíveis que levam a conflitos de interesses ou a erros (Gligor *et al.*, 1998). As políticas de separação de responsabilidades são representadas por meio de pares de autorizações fortemente conflitantes associadas às tarefas sensíveis. Quando tais pares de autorizações são associadas a papéis distintos, a política de SR fica estabelecida entre eles pela indisponibilidade de tais autorizações (positivas) para um mesmo usuário detendo estes papéis.

Nos modelos relacionados na *Tabela 1*, a separação de responsabilidades é orientada a papéis. O C-TMAC (seção 2.5.2, p. 42) e o XORBAC (seção 2.5.7, p. 47) suportam a SRE com a definição de papéis mutuamente exclusivos, isto é, papéis sem usuários em comum. O XORBAC ainda permite a definição de autorizações mutuamente exclusivas, ou seja, autorizações que não podem estar associadas a um mesmo papel. Porém, o modelo não determina uma resolução específica para as situações em que um mesmo usuário esteja associado a papéis com autorizações mutuamente exclusivas. Caso se adote a política padrão do modelo, o usuário possui tais autorizações disponíveis, o que torna essa modalidade de SRE pouco eficaz. Por outro lado, a definição desses papéis mutuamente exclusivos resolveria o problema,

pois um mesmo usuário não poderia compartilhá-las. Ora, mas, assim, as autorizações mutuamente exclusivas seriam desnecessárias.

O modelo E-RBAC (seção 2.5.1, p. 41) suporta a SRD com a definição de papéis ambientais mutuamente exclusivos, isto é, papéis que não podem estar ativos simultaneamente. Assim, autorizações distintas associadas a tais papéis ambientais não ficam disponíveis (para quaisquer usuários) ao mesmo tempo, o que caracteriza a SRD. O OASIS (seção 2.5.8, p. 48) adota uma estratégia similar para suportar a SRD, com a utilização das regras para ativação de papéis, a fim de estabelecer que dois ou mais papéis não podem estar ativos simultaneamente. Por exemplo, podem-se definir regras determinando que o papel *Médico Assistente* somente pode ser ativado por um usuário qualquer quando o papel *Médico Auditor* não estiver ativo, e vice-versa. Assim, as autorizações distintas associadas aos papéis nunca ficam disponíveis ao mesmo tempo para o usuário. Os autores do OASIS reconhecem a necessidade de uma definição mais explícita da SRD no modelo.

O modelo GTRABC (seção 2.5.9, p. 49) segue um enfoque semelhante, porém, generaliza o uso de restrições (temporais) no estabelecimento das relações do modelo. Portanto, além de se poder impor restrições de SRD para ativação simultânea de papéis conflitantes, é possível estabelecer restrições de SRE para atribuição de usuários e de permissões a esses papéis. Por exemplo, poderia-se definir que, se um usuário está associado ao papel *Médico Auditor*, ele não pode estar associado ao papel *Médico Assistente*, e vice-versa. Joshi *et al.* (2003) mostram que, devido a generalidade e a flexibilidade, o GTRBAC é capaz de suportar 38 diferentes formas de separação de responsabilidades. Embora com importância teórica, o problema é que as numerosas formas de SR introduzem mais complexidade ao modelo, gerando desafios no desenvolvimento de ferramentas para administração e verificação das políticas de SR, o que dificulta a sua utilização efetiva em ambientes reais.

5.2.3 Autorizações

O MACA suporta autorizações positivas e negativas assim como os modelos E-RBAC (seção 2.5.1, p. 41), TCM (seção 2.5.4, p. 44) e RBPEAC (seção 2.5.6, p. 46). No MACA, a possibilidade de negar uma autorização é importante para revogar autorizações que concedem o acesso e também para explicitar proibições. Embora a proposta de padrão para CABP do NIST não impeça a existência de autorizações negativas, Sandhu *et al.* (2000) advertem que o seu uso pode ser confuso, especialmente na presença de hierarquias. O principal problema é qual política adotar para a resolução de conflitos entre autorizações.

Os trabalhos que apresentam o E-RBAC (Covington *et al.*, 2002; Covington *et al.*, 2001) omitem a eventual política de resolução de conflitos adotada, embora o modelo permita autorizações positivas e negativas. Já o modelo TCM estabelece uma elaborada política de resolução de conflitos baseada no tipo de entidade a qual a autorização está associada (identidade, coleções de identidades, papéis ou coleções de papéis) e no tipo do objeto protegido pela autorização (coleção de itens de dados ou coleção de tipos de dados), que determinam uma ordem de precedência na avaliação de autorizações. Havendo duas ou mais autorizações conflitantes, prevalecerá aquela avaliada primeiro. Por fim, o RBPEAC adota uma política mais simples: havendo conflitos, prevalece a autorização que concede o acesso.

Um aspecto na herança de autorizações no modelo TCM merece consideração, antes da discussão do enfoque adotado pelo MACA para resolução de conflitos entre autorizações. No TCM, uma diretiva determina se uma autorização é herdada pelos descendentes ou pelos ascendentes de um papel. A possibilidade de um ascendente herdar autorizações de descendentes acarreta anomalias no conceito existente de CABP porque permite que um usuário possua privilégios de um papel ao qual ele não está associado, direta ou indiretamente. Por exemplo, uma autorização definida no papel *Médico* (v. *Figura 3 (a)* na p. 31) poderia ser herdada pelo papel *Profissional de Saúde*. No caso, os usuários associados a *Profissional de Saúde* teriam essa autorização disponível, mesmo sem ser membro do papel *Médico*. Note-se que, na herança tradicional (ascendente para descendente), tal problema não ocorre porque um usuário associado a um papel sempre é membro (indireto) dos papéis ascendentes desse e, portanto, tem o direito de possuir as autorizações herdadas.

O MACA adota um enfoque diferente dos modelos E-RBAC, TCM e RBPEAC para resolver os conflitos entre autorizações. Classifica as autorizações em fortes e fracas para que se possa particionar conflitos que representam conflitos de interesses potenciais daqueles que denotam apenas diferenças de autoridade e de responsabilidade entre papéis. Isso permitiu estabelecer a seguinte política de resolução de conflitos:

- Quando duas autorizações fracas conflitam, prevalece a aquela que concede o acesso. O objetivo é permitir que o usuário desempenhe as ações necessárias ao cumprimento das suas funções organizacionais, visando atender o princípio da necessidade saber/fazer;
- Quando duas autorizações fortes conflitam, prevalece aquela que proíbe o acesso. O objetivo é impedir que o usuário desempenhe ações que o

levem a cometer fraudes ou erros, para suportar políticas de separação de responsabilidades.

Os conceitos de autorizações fortes e fracas utilizados no MACA, originalmente introduzidos por Rabitti *et al.* (1991), foram posteriormente refinados por Bertino *et al.* (1999). Entretanto, a forma como o MACA os emprega apresenta importantes diferenças em relação ao seu uso nesses modelos de autorização. O MACA os utiliza para distinguir os conflitos que representam conflitos de interesses potenciais daqueles que denotam conflitos casuais entre papéis. A distinção dos conflitos permite utilizar diferentes critérios para sua resolução: 1) para conflitos fracos, o critério adotado foi atender o princípio da necessidade saber/fazer; e 2) para conflitos fortes, o critério adotado foi suportar políticas de separação de responsabilidades. Bertino *et al.* (1999) e Rabitti *et al.* (1991) utilizam outros critérios, levando a diferentes políticas para resolução de conflitos entre autorizações.

O modelo proposto por Bertino *et al.* (1999) implementa políticas de acesso discricionárias (v. seção 2.2, p. 17), onde autorizações (positivas ou negativas, fortes ou fracas) podem associar-se a grupos de usuários (estruturados em hierarquias gerais) e também diretamente a usuários individuais. Nele, a existência de conflitos fortes entre autorizações é categoricamente proibida. Não é possível modificar um estado do modelo que resulte em conflitos fortes. Por exemplo, um mesmo usuário não pode ser membro de dois ou mais grupos que tenham autorizações fortes conflitantes entre si. Os autores justificam que, na hipótese de haver conflitos fortes, qualquer política de resolução implicaria na desobediência a uma das autorizações. Ou seja, o modelo adota, por princípio, a observância estrita de uma autorização forte. O MACA admite conflitos fortes em estados do modelo, para representar conflitos de interesses latentes, porque a sua política de resolução considera a desobediência a uma autorização positiva forte conflitante uma razão superior à possibilidade de cometimento de fraude ou erro devido a poderes excessivos concentrados num único usuário. Em relação aos conflitos fracos, o modelo de Bertino *et al.* (1999) é mais permissivo, admitindo-os. A política para resolução de conflitos fracos segue dois critérios distintos: 1) a autorização conflitante, positiva ou negativa, diretamente associada a um usuário tem precedência sobre as autorizações conflitantes equivalentes definidas em grupos nos quais o usuário é membro (enfoque similar ao seguido no modelo TCM); e 2) nos casos em que o critério anterior não se aplica, prevalece a autorização conflitante fraca que nega o acesso. O modelo de Bertino *et al.* (1999) adota, no segundo critério, uma política oposta a do MACA porque grupos não repre-

sentam funções organizacionais, portanto, não há a justificativa da *necessidade* para conceder o acesso. Logo, pelo princípio da necessidade de saber/fazer, ele é negado.

No modelo proposto por Rabitti *et al.* (1991), autorizações (positivas ou negativas, fortes ou fracas) não podem associar-se diretamente a usuários, mas apenas a papéis (estruturados em hierarquias gerais). Porém, embora adote a nomenclatura “papel”, os papéis têm a finalidade de agrupar usuários e não autorizações de funções organizacionais específicas. Nesse aspecto, não se enquadram no atual conceito de papéis, mas sim no de grupos. Ademais, o modelo não incorpora características básicas do CABP do NIST, como os conceitos de sessão e de ativação/desativação de papéis. O modelo não permite a existência de conflitos (fortes ou fracos) entre autorizações associadas a papéis hierarquicamente relacionados. Por exemplo, os papéis *Administrador* (v. *Figura 3* (a), p. 31) e *Profissional de Saúde* não poderiam ter autorizações conflitantes entre si porque implicaria na existências dessas autorizações no papel *Diretor Clínico*, por herança. Entretanto, não haveria impedimentos para conflitos entre as autorizações dos papéis *Analista de Registro de Saúde* e *Paramédico*, pois não possuem descendentes em comum. Porém, um usuário pode pertencer a papéis com conflitos entre si (*e. g.*, *Analista de Registro de Saúde* e *Paramédico*). O critério de resolução de conflitos é baseado no conceito da autorização mais específica, com a precedência das autorizações positivas sobre as negativas. Isso permite que uma autorização positiva de um papel (*e. g.*, *Analista de Registro de Saúde*) tenha precedência sobre uma autorização negativa forte e equivalente existente num outro papel (*e. g.*, *Paramédico*). Assim, um usuário detendo tais papéis teria acesso a um recurso para o qual há uma autorização forte e negativa. A desvantagem desse enfoque é que ele não considera o fato de um conflito ser forte ou fraco para resolvê-los. Ou seja, para fins de resolução de conflitos, autorizações fortes e fracas são irrelevantes. São usadas apenas para indicar se uma autorização pode (fraca) ou não (forte) ser redefinida num papel descendente. O MACA, por outro lado, utiliza essa informação para conciliar o princípio da necessidade saber/fazer com políticas de separação de responsabilidades.

5.2.4 Regras contextuais

O MACA aplica as regras contextuais nas autorizações para determinar dinamicamente se elas são positivas ou negativas. Dos modelos relacionados na *Tabela 1*, o RBPEAC (seção 2.5.6, p. 46) é o único que adota enfoque similar. Os modelos XORBAC (seção 2.5.7, p. 47), OASIS (seção 2.5.8, p. 48) e GTRBAC (seção 2.5.9, p. 49) também permitem situar uma regra numa autorização, mas não para decidir se ela é

positiva ou negativa, e sim para estabelecer as condições em que uma autorização está associada a um papel, ou seja, se ela é aplicável. A principal vantagem de aplicar regras nas autorizações é a possibilidade de se definir políticas de acesso detalhadas, dependentes de parâmetros obtidos do ambiente existente nas operações dos objetos protegidos, no momento de sua execução. O modelo E-RBAC (seção 2.5.1, p. 41), indiretamente, utiliza regras para determinar se uma autorização é aplicável ou não. Papéis ambientais podem ser vistos como abstrações de regras e atuam como condições para aplicabilidade de uma autorização, mas apresentam a limitação de não serem parametrizados.

O modelo C-TMAC (seção 2.5.2, p. 42) associa regras contextuais exclusivamente a equipes. A função das regras é filtrar as autorizações disponíveis para um usuário, de acordo com os papéis e as equipes que ele ativou. O problema do C-TMAC é que ele limita-se a proteger recursos que são bancos de dados relacionais. Isso porque as autorizações são definidas como visões ou consultas SQL a bancos de dados e as regras são cláusulas *where* da linguagem SQL. Ademais, no C-TMAC, as autorizações de um usuário são a combinação das autorizações associadas a todos os papéis correntes ativados pelos membros das equipes ativadas, das quais ele é membro, filtradas pelas regras das equipes. Conseqüentemente, o usuário pode exercer privilégios de um papel ao qual ele não está associado, violando um princípio básico do CABP.

O modelo CS-RBAC (seção 2.5.3, p. 43) associa regras contextuais exclusivamente a papéis. Quando a regra é avaliada para verdadeiro, as autorizações associadas ao papel ficam disponíveis para o usuário. O problema no CS-RBAC é que só existem dois contextos, de usuário e de objeto, que disponibilizam apenas valores simples, equivalentes às variáveis contextuais do MACA, não havendo conjuntos ou funções contextuais. Embora os valores dos contextos de usuário e de objeto sejam dependentes do usuário corrente e da aplicação (objeto protegido), respectivamente, a avaliação das regras não o são. Assim, não é possível variar a regra a ser avaliada dependendo da aplicação que o usuário tenta executar. A regra de um papel é sempre global às autorizações e, por conseguinte, às aplicações. Isso limita a implementação de políticas de acesso detalhadas, ao nível das operações executadas nos objetos protegidos.

O modelo DRIVE-RBAC (seção 2.5.5, p. 45) associa regras (eventos contextuais) exclusivamente a papéis a fim de determinar as condições para sua ativação/desativação em sessões de um usuário, o que limita a implementação de políti-

cas de acesso detalhadas ao nível de aplicação. Outra importante limitação do DRIVE-RBAC é que tais regras possuem um formato fixo, baseadas também em variáveis contextuais fixas, que determinam um intervalo de tempo, uma periodicidade, um local, uma circunstância, uma prioridade para ativação/desativação de um papel. Isso dificulta a implementação de políticas de acesso dependentes da situação do paciente (*e. g.*, está internado ou com consulta agendada) ou de sua relação com o usuário (*e. g.*, é seu médico particular), por exemplo.

5.2.5 Implementação e aplicação ao PEP

O MACA foi desenvolvido para plataforma Java 2 SE, com o ORB JacORB, utilizando o servidor OpenLDAP para armazenar as informações persistentes do modelo, como os papéis e sua hierarquia, as autorizações contextuais, a representação das operações e dos objetos, as contas de usuários e os relacionamentos usuário-papel e papel-autorização (v. seção 4.2 na p. 109 para informações adicionais). A implementação do MACA se diferencia de outros modelos para CABP levantados na literatura, inclusive aqueles presentes na *Tabela 1*, por implementar contextos por meio de interfaces CORBA, que permitem considerar informações provenientes de sistemas legados em plataforma distribuídas e heterogêneas. Os modelos revisados (*Tabela 1*) implementam contextos em linguagem de *scripts* de sistemas de gerência de bancos de dados relacionais (C-TMAC, OASIS e GTRBAC), que possuem baixa interoperabilidade para ambientes que não seguem o padrão SQL, ou permitem a sua implementação numa única linguagem de programação (E-RBAC, RBPEAC e XORBAC), ou ainda os definem de forma fixa e limitada (CS-RBAC e DRIVE-RBAC).

A implementação do MACA ainda se distingue dos modelos revisados por representar/armazenar os dados persistentes num serviço de diretórios LDAP. Os modelos C-TMAC, TCM, OASIS e GTRBAC utilizam sistemas de gerência de bancos de dados relacionais. Os autores dos demais modelos não explicitam onde os dados persistentes são representados/armazenados.

O LDAP é mais vantajoso que os SGBDs relacionais para representar/armazenar modelos de CABP, que requerem estruturas de dados hierárquicas, com flexibilidade para inclusão/exclusão de atributos de entidades (*e. g.*, usuários, papéis), onde a frequência de operações de leitura é elevada quando comparada à frequência de operações de escrita. Os SGBDs relacionais não implementam estruturas de dados hierárquicas diretamente e não possuem flexibilidade para inclusão/exclusão de atributos. Por exemplo, numa tabela representando contas de usuá-

rio, a adição de um novo atributo para uma única conta implica na definição de uma coluna com todos os valores nulos, exceto o valor do atributo em questão. Ademais, os SGBDs relacionais são projetados para suportar eficientemente o processamento transacional complexo, requerido em sistemas tradicionais, onde o número de operações de leitura e escrita são equiparáveis. Por outro lado, o LDAP suporta diretamente estruturas de dados hierárquicas, com flexibilidade para inclusão/exclusão de atributos (v. seção 4.2.1, p. 109), e suas implementações são otimizadas para privilegiar operações de leitura intensivas, em detrimento das operações de escrita.

Com exceção do XORBAC, os modelos revisados, em sua maioria, estão implementados em protótipos e foram utilizados experimentalmente em estudos de caso para o PEP. No entanto, não há relatos de uso efetivo em ambientes de produção, particularmente em organizações de saúde. Por sua vez, a implementação do MACA, atualmente na versão 3.2, está em uso efetivo para controlar o acesso ao PEP no InCor – HC.FMUSP, em ambiente de produção, desde julho de 2001 (v. seção 4.3.2, p. 142). Em adição, o MACA se caracteriza por ser produto de software estável, mas em evolução, apto a ser utilizado em outras organizações. Ainda dispõe de manuais (v. item Manuais na p. 195) para instalação e configuração, para definição de políticas de acesso e para utilização dos seus serviços por programas clientes.

5.3 Limitações do MACA e extensões futuras

Embora o MACA ofereça contribuições para solucionar o problema do controle de acesso ao prontuário eletrônico do paciente no âmbito de organizações de saúde, ele apresenta limitações que merecem ser investigadas visando o desenvolvimento de soluções futuras.

A utilização efetiva do MACA em ambientes reais será facilitada com a disponibilidade de uma ferramenta para auxiliar na detecção de conflitos e na verificação da implementação de políticas de alto nível, em moldes similares àqueles propostos por Lupu & Sloman (1999). Essa ferramenta deveria ser capaz de ajudar a responder questões como “existe alguém que pode ler prescrições de um paciente que não está internado?” ou “existe alguém que pode criar prescrições e, concomitantemente, tem permissão para glosá-las?”. Ela seria útil para detectar possíveis inconsistências ou violações de políticas previamente estabelecidas, incluindo o não cumprimento de políticas de separação de responsabilidades. Isso porque a capacidade de se definir políticas (técnicas) de acesso elaboradas é de pouca valia se não houver

meios eficazes que auxiliem a evidenciar a sua validade, que é a satisfação das políticas de acesso de alto nível estabelecidas pela organização.

A utilização de informações contextuais, particularmente do ambiente externo ao serviço do MACA (parâmetros de regras e contextos externos), dá margem para erros que podem causar fluxos indesejáveis de informação e também possibilita ataques visando influenciar, maliciosamente, a decisão de acesso. É recomendável investigar soluções para o problema, como a proposta por Belokosztolszki *et al.* (2003), que disciplina o fluxo da informação com base em meta-dados semânticos sobre as informações contextuais. McDaniel (2003) observa que, embora uma arquitetura para políticas baseadas em contextos preveja a comunicação com os contextos remotos via TLS, para assegurar confidencialidade, integridade e autenticidade das partes, outras condições de segurança vão depender das implementações dos contextos e da origem das informações. Segundo o autor, a avaliação das vulnerabilidades e a proposta de soluções devem ser feitas caso a caso.

Segundo Zhang *et al.* (2002), existem situações em que um médico precisa compartilhar, com um colega, o acesso exclusivo que ele tem a uma informação sensível no PEP de um paciente sob seus cuidados. Para isso ele precisa ter o poder de delegar parte de sua autoridade para outrem atuar em seu nome, em geral, por um tempo limitado e sob circunstâncias específicas. Essa flexibilidade também é importante em situações de emergência, onde o próprio médico pode conceder, de imediato, parte de seus poderes para auxiliares, sem necessidade de recorrer à intermediação de administradores. No MACA, o poder que um usuário (com papéis administrativos) tem de conceder privilégios, com a atribuição de papéis, não se constitui em delegação verdadeira. A delegação se caracteriza pela capacidade de um usuário atribuir um subconjunto dos próprios privilégios. A introdução de mecanismos de delegação no MACA é um tema relevante, que merece investigação futura.

A utilização de regras contextuais exclusivamente em autorizações dificulta a implementação de políticas de acesso que não são dependentes de aplicações. Por exemplo, uma política poderia estabelecer que usuários no papel de *Auxiliar Administrativo* somente exerceriam seus privilégios no horário de expediente. No MACA, para essa política ser implementada, deveria-se incluir uma regra bloqueando o acesso fora do horário de expediente em cada autorização que permite a execução de aplicações, mesmo sendo as regras independentes da aplicação. A incorporação, ao MACA, de regras para especificar condições para ativação/desativação de papéis facilitaria a configuração de políticas desse tipo. Para o exemplo anterior,

bastaria incluir uma regra proibindo a ativação do papel *Auxiliar Administrativo* no horário do expediente. Portanto, como o papel não poderia ser ativado, os usuários não exerceriam os privilégios de *Auxiliar Administrativo* fora do horário de expediente. Regras também poderiam ser empregadas para impor restrições no relacionamento usuário-papel, estabelecendo, por exemplo, o prazo de validade de uma associação.

Uma outra limitação do MACA é a atuação pontual de uma autorização sobre um objeto protegido, desconsiderando que os objetos podem estar estruturados numa hierarquia. O MACA deveria permitir que uma autorização abrangesse todo um ramo de uma árvore de objetos. Por exemplo, uma autorização proibindo o acesso ao *Serviço de Informações Cadastrais* (Figura 5 (b)) deveria, quando indicado, também abranger os objetos subordinados. A inexistência dessa facilidade dificulta a administração da política de acesso porque obriga a redefinição de um mesmo tipo de autorização nos objetos subordinados.

Por fim, a ausência de mecanismos de modularização na linguagem de expressões lógicas complica a gerência de políticas recorrentes. Atualmente, se uma mesma expressão precisa ser definida em duas ou mais autorizações, ela é reescrita em cada uma delas. A introdução de funções nomeadas, compondo uma biblioteca de regras, minimizaria este problema, pois bastaria chamar a função, em vez de reescrever toda a expressão.

Conclusão

“(...) o poder de bem julgar e de distinguir o verdadeiro do falso, que é propriamente o que se chama o bom senso ou a razão é naturalmente igual em todos os homens; e assim, a diversidade de nossas opiniões não resulta de serem umas mais razoáveis que outras, mas, somente de conduzirmos nossos pensamentos por diversas vias, e de não considerarmos as mesmas coisas. Porque não basta ter o espírito bom, o principal é aplicá-lo bem.”

René Descartes, 1637

Este trabalho propôs o MACA, um modelo de autorização adequado às exigências de controle de acesso para o prontuário eletrônico em grandes organizações de saúde, capaz de conciliar a permissão de acesso ao PEP motivada pela necessidade de cuidado ao paciente, com a proibição de acesso em situações indevidas. Isso é possível porque o MACA permite incluir regras relacionando informações contextuais nas autorizações, viabilizando um controle de acesso preciso, de acordo com o direito e a necessidade do usuário realizar uma tarefa. Ademais, a utilização rotineira de sua implementação no controle de acesso ao PEP do InCor-HC.FMUSP evidencia a exequibilidade do MACA, da sua implementação e da sua aplicação prática em casos reais. As principais contribuições do MACA são relacionadas a seguir:

- Extensão do modelo de referência proposto pelo NIST para o CABP, com a inclusão de regras de autorização, relacionando informações existentes em contextos para decidir um acesso. Contextos estão disponíveis via interfaces CORBA, que podem ser implementadas em plata-

formas distribuídas e em linguagens heterogêneas. Isso confere ao MACA flexibilidade e poder expressivo para estabelecer políticas de acesso para o PEP e políticas administrativas para o CABP que se adaptam à diversidade ambiental e cultural das organizações de saúde;

- Utilização do CABP pelos sistemas componentes do PEP de forma transparente e não ostensiva para o usuário final, consequência da ativação automática de papéis independente da separação de responsabilidades. Como resultado, consegue-se um PEP mais fácil de usar quando comparado com a exigência da ativação explícita de papéis pelo usuário final em outros modelos de CABP;
- Conciliação do princípio da necessidade de saber/fazer com o conceito de separação de responsabilidades na política de resolução de conflitos entre autorizações. Com isso, diferenciam-se os conflitos que ocorrem casualmente entre papéis, quando o acesso é concedido por necessidade, daqueles que se configuram em conflitos de interesses reais, quando o acesso é proibido para se reduzir as chances de cometimento de fraude ou erros;
- Disponibilidade de uma implementação do MACA como um produto de software estável, integrado a uma arquitetura baseada em padrões de processamento aberto e distribuído, apto para utilização em organizações reais. Como consequência, dois importantes requisitos para o controle de acesso ao PEP em grandes organizações de saúde são atendidos:
 - 1) a possibilidade dos diferentes componentes do PEP, muitas vezes legados, utilizarem os serviços de autenticação de usuários e de autorização de acesso a partir de plataformas heterogêneas;
 - 2) a capacidade de administrar e de impor políticas de acesso e de autenticação de modo unificado e coerente, a despeito da natureza distribuída e heterogênea do PEP.

A implementação do MACA ainda tem a vantagem de ser um software livre, disponibilizado sob licença GNU GPL, de basear-se em componentes de software sem custos de licenciamento e de apresentar um bom desempenho para as demandas de acesso estimadas.

A

Linguagem de expressões lógicas: sintaxe formal

Este anexo especifica a gramática que define a sintaxe formal da linguagem de expressões lógicas usadas nas regras de autorização (v. seção 3.2). A gramática segue o formato do *SableCC* (Gagnon & Hendren, 1998; SableCC, 2003), um *framework* orientado a objetos em Java para construção de compiladores.

```
Package ContextExpressionLanguage;

/*****
 * Helpers
 *****/
Helpers
    unicode_input_character = [0..0xffff];

    ht = 0x0009;
    lf = 0x000a;
    ff = 0x000c;
    cr = 0x000d;
    sp = ' ';

    line_terminator = lf | cr | cr lf;
    input_character = [unicode_input_character - [cr + lf]];
    not_star = [input_character - '*'] | line_terminator;
    not_star_not_slash = [input_character - ['*' + '/']] | line_terminator;

    unicode_letter =
        [0x0041..0x005a] | [0x0061..0x007a] | [0x00aa..0x00ba] | [0x00b5..0x00b5] |
        [0x00ba..0x00ba] | [0x00c0..0x00d6] | [0x00d8..0x00f6] | [0x00f8..0x01f5] |
        [0x01fa..0x0217] | [0x0250..0x02a8] | [0x02b0..0x02b8] | [0x02bb..0x02c1] |
```

[0x02d0..0x02d1]		[0x02e0..0x02e4]		[0x037a..0x037a]		[0x0386..0x0386]	
[0x0388..0x038a]		[0x038c..0x038c]		[0x038e..0x03a1]		[0x03a3..0x03ce]	
[0x03d0..0x03d6]		[0x03da..0x03da]		[0x03dc..0x03dc]		[0x03de..0x03de]	
[0x03e0..0x03e0]		[0x03e2..0x03f3]		[0x0401..0x040c]		[0x040e..0x044f]	
[0x0451..0x045c]		[0x045e..0x0481]		[0x0490..0x04c4]		[0x04c7..0x04c8]	
[0x04cb..0x04cc]		[0x04d0..0x04eb]		[0x04ee..0x04f5]		[0x04f8..0x04f9]	
[0x0531..0x0556]		[0x0559..0x0559]		[0x0561..0x0587]		[0x05d0..0x05ea]	
[0x05f0..0x05f2]		[0x0621..0x063a]		[0x0640..0x064a]		[0x0671..0x06b7]	
[0x06ba..0x06be]		[0x06c0..0x06ce]		[0x06d0..0x06d3]		[0x06d5..0x06d5]	
[0x06e5..0x06e6]		[0x0905..0x0939]		[0x093d..0x093d]		[0x0958..0x0961]	
[0x0985..0x098c]		[0x098f..0x0990]		[0x0993..0x09a8]		[0x09aa..0x09b0]	
[0x09b2..0x09b2]		[0x09b6..0x09b9]		[0x09dc..0x09dd]		[0x09df..0x09e1]	
[0x09f0..0x09f1]		[0x0a05..0x0a0a]		[0x0a0f..0x0a10]		[0x0a13..0x0a28]	
[0x0a2a..0x0a30]		[0x0a32..0x0a33]		[0x0a35..0x0a36]		[0x0a38..0x0a39]	
[0x0a59..0x0a5c]		[0x0a5e..0x0a5e]		[0x0a72..0x0a74]		[0x0a85..0x0a8b]	
[0x0a8d..0x0a8d]		[0x0a8f..0x0a91]		[0x0a93..0x0aa8]		[0x0aaa..0x0ab0]	
[0x0ab2..0x0ab3]		[0x0ab5..0x0ab9]		[0x0abd..0x0abd]		[0x0ae0..0x0ae0]	
[0x0b05..0x0b0c]		[0x0b0f..0x0b10]		[0x0b13..0x0b28]		[0x0b2a..0x0b30]	
[0x0b32..0x0b33]		[0x0b36..0x0b39]		[0x0b3d..0x0b3d]		[0x0b5c..0x0b5d]	
[0x0b5f..0x0b61]		[0x0b85..0x0b8a]		[0x0b8e..0x0b90]		[0x0b92..0x0b95]	
[0x0b99..0x0b9a]		[0x0b9c..0x0b9c]		[0x0b9e..0x0b9f]		[0x0ba3..0x0ba4]	
[0x0ba8..0x0baa]		[0x0bae..0x0bb5]		[0x0bb7..0x0bb9]		[0x0c05..0x0c0c]	
[0x0c0e..0x0c10]		[0x0c12..0x0c28]		[0x0c2a..0x0c33]		[0x0c35..0x0c39]	
[0x0c60..0x0c61]		[0x0c85..0x0c8c]		[0x0c8e..0x0c90]		[0x0c92..0x0ca8]	
[0x0caa..0x0cb3]		[0x0cb5..0x0cb9]		[0x0cde..0x0cde]		[0x0ce0..0x0ce1]	
[0x0d05..0x0d0c]		[0x0d0e..0x0d10]		[0x0d12..0x0d28]		[0x0d2a..0x0d39]	
[0x0d60..0x0d61]		[0x0e01..0x0e2e]		[0x0e30..0x0e30]		[0x0e32..0x0e33]	
[0x0e40..0x0e46]		[0x0e81..0x0e82]		[0x0e84..0x0e84]		[0x0e87..0x0e88]	
[0x0e8a..0x0e8a]		[0x0e8d..0x0e8d]		[0x0e94..0x0e97]		[0x0e99..0x0e9f]	
[0x0ea1..0x0ea3]		[0x0ea5..0x0ea5]		[0x0ea7..0x0ea7]		[0x0eaa..0x0eab]	
[0x0ead..0x0eae]		[0x0eb0..0x0eb0]		[0x0eb2..0x0eb3]		[0x0ebd..0x0ebd]	
[0x0ec0..0x0ec4]		[0x0ec6..0x0ec6]		[0x0edc..0x0edd]		[0x0f40..0x0f47]	
[0x0f49..0x0f69]		[0x10a0..0x10c5]		[0x10d0..0x10f6]		[0x1100..0x1159]	
[0x115f..0x11a2]		[0x11a8..0x11f9]		[0x1e00..0x1e9b]		[0x1ea0..0x1ef9]	
[0x1f00..0x1f15]		[0x1f18..0x1f1d]		[0x1f20..0x1f45]		[0x1f48..0x1f4d]	
[0x1f50..0x1f57]		[0x1f59..0x1f59]		[0x1f5b..0x1f5b]		[0x1f5d..0x1f5d]	
[0x1f5f..0x1f7d]		[0x1f80..0x1fb4]		[0x1fb6..0x1fbc]		[0x1fbe..0x1fbe]	
[0x1fc2..0x1fc4]		[0x1fc6..0x1fcc]		[0x1fd0..0x1fd3]		[0x1fd6..0x1fdb]	
[0x1fe0..0x1fec]		[0x1ff2..0x1ff4]		[0x1ff6..0x1ffc]		[0x207f..0x207f]	
[0x2102..0x2102]		[0x2107..0x2107]		[0x210a..0x2113]		[0x2115..0x2115]	
[0x2118..0x211d]		[0x2124..0x2124]		[0x2126..0x2126]		[0x2128..0x2128]	
[0x212a..0x2131]		[0x2133..0x2138]		[0x3005..0x3005]		[0x3031..0x3035]	
[0x3041..0x3094]		[0x309b..0x309e]		[0x30a1..0x30fa]		[0x30fc..0x30fe]	
[0x3105..0x312c]		[0x3131..0x318e]		[0x4e00..0x9fa5]		[0xac00..0xd7a3]	
[0xf900..0xfa2d]		[0xfb00..0xfb06]		[0xfb13..0xfb17]		[0xfb1f..0xfb28]	
[0xfb2a..0xfb36]		[0xfb38..0xfb3c]		[0xfb3e..0xfb3e]		[0xfb40..0xfb41]	
[0xfb43..0xfb44]		[0xfb46..0xfbb1]		[0xfbd3..0xfd3d]		[0xfd50..0xfd8f]	
[0xfd92..0xfdc7]		[0xfdf0..0xfdfb]		[0xfe70..0xfe72]		[0xfe74..0xfe74]	
[0xfe76..0xfefc]		[0xff21..0xff3a]		[0xff41..0xff5a]		[0xff66..0xffbe]	
[0xffc2..0xffc7]		[0xffca..0xffcf]		[0xffd2..0xffd7]		[0xffda..0xffdc];	

```

unicode_digit =
    [0x0030..0x0039] | [0x0660..0x0669] | [0x06f0..0x06f9] | [0x0966..0x096f] |
    [0x09e6..0x09ef] | [0x0a66..0x0a6f] | [0x0ae6..0x0aef] | [0x0b66..0x0b6f] |
    [0x0be7..0x0bef] | [0x0c66..0x0c6f] | [0x0ce6..0x0cef] | [0x0d66..0x0d6f] |
    [0x0e50..0x0e59] | [0x0ed0..0x0ed9] | [0x0f20..0x0f29] | [0xff10..0xff19];

letter = unicode_letter | '_';
digit  = unicode_digit;

string_character = [input_character - ['"' + '\']];

/*****
* Tokens
*****/
Tokens

blank          = sp* | cr | lf;
traditional_comment = '/' not_star+ '*'
                    (not_star_not_slash not_star* '*' +) '/' ;
end_of_line_comment = '/' input_character* line_terminator?;

true           = 'true';
false          = 'false';
l_parenthese   = '(';
r_parenthese   = ')';
l_brace        = '{';
r_brace        = '}';

expression_abstraction = 'exp-abs';

plus  = '+';
minus = '-';
star  = '*';
div   = '/';
mod   = '%';

lt  = '<';
gt  = '>';
not = '!';
eq  = '=';
lteq = '<=';
gteq = '>=';
neq  = '!=';
and  = '&';
or   = '|';
in   = 'in';
set  = '..';

qualifier = '.';

```

```

comma      = ',';

identifier = letter (letter | digit)*;
number     = digit+;
text       = '"' string_character* '"';

/*****
* Ignored Tokens
*****/
Ignored Tokens

blank,
traditional_comment,
end_of_line_comment;

/*****
* Productions
*****/
Productions

expression =
    {logical_expression}
    logical_expression |
    {expression_application}
    expression_application;

expression_application =
    expression_declaration l_parenthese argument_list? r_parenthese;

argument_list =
    {expression}
    expression |
    {argument_list}
    argument_list comma expression;

expression_declaration =
    expression_header expression_body;

expression_header =
    expression_abstraction l_parenthese formal_parameter_list? r_parenthese;

formal_parameter_list =
    {formal_parameter}
    formal_parameter |
    {formal_parameter_list}
    formal_parameter_list comma formal_parameter;

formal_parameter =
    identifier;

```

```

expression_body =
    l_brace logical_expression r_brace;

function_application =
    {identifier}
    identifier l_parenthese argument_list? r_parenthese |
    {qualified_identifier}
    [left]:identifier qualifier [right]:identifier l_parenthese
    argument_list? r_parenthese;

logical_expression =
    {and_expression}
    and_expression |
    {or}
    [left]:logical_expression or [right]:and_expression;

and_expression =
    {equality_expression}
    equality_expression |
    {and}
    [left]:and_expression and [right]:equality_expression;

equality_expression =
    {relational_expression}
    relational_expression |
    {eq}
    [left]:equality_expression eq [right]:relational_expression |
    {neq}
    [left]:equality_expression neq [right]:relational_expression;

relational_expression =
    {additive_expression}
    additive_expression |
    {in}
    [left]:additive_expression in [right]:set_expression |
    {lteq}
    [left]:additive_expression lteq [right]:additive_expression |
    {gteq}
    [left]:additive_expression gteq [right]:additive_expression |
    {lt}
    [left]:additive_expression lt [right]:additive_expression |
    {gt}
    [left]:additive_expression gt [right]:additive_expression;

set_expression =
    {set_literal}
    [left]:additive_expression set [right]:additive_expression |
    {set_identifier}
    identifier |
    {qualified_set_identifier}

```



```

    [left]:identifier qualifier [right]:identifier |
{set_function_application}
    function_application;

additive_expression =
    {multiplicative_expression}
        multiplicative_expression |
    {plus}
        [left]:additive_expression plus [right]:multiplicative_expression |
    {minus}
        [left]:additive_expression minus [right]:multiplicative_expression;

multiplicative_expression =
    {unary_expression}
        unary_expression |
    {star}
        [left]:multiplicative_expression star [right]:unary_expression |
    {div}
        [left]:multiplicative_expression div [right]:unary_expression |
    {mod}
        [left]:multiplicative_expression mod [right]:unary_expression;

unary_expression =
    {value}
        value |
    {not}
        not unary_expression;

value =
    {true}
        true |
    {false}
        false |
    {number}
        number |
    {text}
        text |
    {identifier}
        identifier |
    {qualified_identifier}
        [left]:identifier qualifier [right]:identifier |
    {function_application}
        function_application |
    {expression_declaration}
        expression_declaration |
    {expression}
        l_parenthese expression r_parenthese;

```

Modelo do formulário de solicitação de acesso a sistemas informatizados

(frente)

SOLICITAÇÃO ACESSO A SISTEMAS INFORMATIZADOS Instituto do Coração - HCFMUSP
No. Solicitação:																																									
Nome Completo do Solicitante			Matrícula HC / FZ / FMUSP																																						
Área Meio		Cargo/Função																																							
Ramal		E-mail (se tiver)																																							
Serviços Solicitados:																																									
☐ E-MAIL ☐ Acesso ao Prontuário Web ☐ Sistema de Laudos ☐ Laudos Radioisótopos ☐ Internação – InCor ☐ Registro de Procedimentos ☐ Guia Única – Consultório ☐ Guia Única - CACI ☐ Relatórios de Produção ☐ Prescrição ☐ Resumo Clínico ☐ Programação Cirúrgica ☐ Sistema de Cirurgia ☐ Prontuário ☐ Prontuário – Histórico ☐ Diário de Enfermagem ☐ Lista de Espera (Fila Internação) ☐ Faturamento – FAT		☐ Sistema de Ponto ☐ Agenda de Exames ☐ SIFH ☐ NOMECO Web <table border="1"> <tr> <td>ERP Fundação Zerbini</td> <td>C</td> <td>A</td> <td>S</td> </tr> <tr> <td>GL (Contabilidade)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>FA (Imobilizado)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>AP (Contas a pagar)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>AR (Contas a receber)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>CASH (Tesouraria)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>RI (Rec. Integrado)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>INV (Materiais)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> <tr> <td>PO (Compras)</td> <td>☐</td> <td>☐</td> <td>☐</td> </tr> </table> ☐ ORGANIZAÇÕES : ☐ Mumps(Especificar) :				ERP Fundação Zerbini	C	A	S	GL (Contabilidade)	☐	☐	☐	FA (Imobilizado)	☐	☐	☐	AP (Contas a pagar)	☐	☐	☐	AR (Contas a receber)	☐	☐	☐	CASH (Tesouraria)	☐	☐	☐	RI (Rec. Integrado)	☐	☐	☐	INV (Materiais)	☐	☐	☐	PO (Compras)	☐	☐	☐
ERP Fundação Zerbini	C	A	S																																						
GL (Contabilidade)	☐	☐	☐																																						
FA (Imobilizado)	☐	☐	☐																																						
AP (Contas a pagar)	☐	☐	☐																																						
AR (Contas a receber)	☐	☐	☐																																						
CASH (Tesouraria)	☐	☐	☐																																						
RI (Rec. Integrado)	☐	☐	☐																																						
INV (Materiais)	☐	☐	☐																																						
PO (Compras)	☐	☐	☐																																						
		C= Consulta / A = Atualização / S= Setup																																							
Termo de Compromisso e Responsabilidade																																									
Declaro que tenho conhecimento e concordo integralmente com a Ordem Conjunta de Serviço No. 06/98 de 25/09/1998, que dispõe sobre o fornecimento de senhas para acesso lógico às informações, que é também aplicável aos sistemas desenvolvidos/ instalados no InCor, sob responsabilidade do Serviço de Informática.																																									
Reconheço a minha senha (password) como minha assinatura digital e comprometo-me a:																																									
1. Não ceder a outra pessoa qualquer a senha de acesso (password) aos sistemas e computadores do InCor e do Complexo H.C. 2. Cuidar para a integridade, confidencialidade e disponibilidade dos dados e informações e sistemas, devendo comunicar por escrito à chefia imediata quaisquer irregularidades, desvios ou falhas identificadas 3. Não usar os sistemas para trabalho de natureza particular 4. Solicitar cancelamento de minha conta (login) quando cessar o vínculo com o InCor-HCFMUSP ou com o uso de um sistema em particular																																									
Data da Solicitação		Assinatura do Solicitante		Assinatura do Diretor da área																																					

(verso)

ORDEM CONJUNTA DE SERVIÇO Nº 06/98*Dispõe sobre o fornecimento de senhas para acesso lógico às informações em saúde, no HCFMUSP*

O Diretor Clínico e o Superintendente do Hospital Das Clínicas da Faculdade de Medicina da Universidade de São Paulo – HCFMUSP, no uso de suas atribuições, com aprovação do Conselho Deliberativo, em sessão de 8/9/98, considerando que:

- os dados, informações e sistemas informatizados de saúde devem ser protegidos contra ações intencionais ou acidentais que impliquem perda, destruição, inserção, cópia, acesso e alteração indevidos, em conformidade com o princípio da confidencialidade integridade e disponibilidade;
- devem ser adotadas medidas de segurança proporcionais aos riscos existentes e à magnitude dos danos potenciais;
- as necessidades de segurança devem ser avaliadas com relação a **confidencialidade integridade e disponibilidade**;
- acesso aos sistemas de informações de saúde deve ser controlado e estar disponível apenas às pessoas devidamente autorizadas;
- os usuários devem ser permanentemente conscientizados sobre os aspectos de segurança e formas de proteção dos recursos e informações sobre sua responsabilidade.

RESOLVEM expedir a presente **Ordem Conjunta de Serviço**, objetivando disciplinar o fornecimento de **senhas** para o acesso lógico aos Sistemas Informatizados de Saúde, no Complexo HCFMUSP conforme a seguir articulado.

I - DISPOSIÇÕES PRELIMINARES

Artigo 1º - Para os efeitos desta Ordem Conjunta de Serviço entende-se por:

- I. **Usuário**: pessoa física cadastrada e habilitada nos Sistemas para acesso a informações em saúde;
- II. **Habilitação**: procedimento que permite ao usuário cadastrado acessar sistemas informatizados de Saúde;
- III. **Acesso lógico**: operação de atualização e consulta de dados e informações em um sistema;
- IV. **Perfil**: subconjunto de transações de um Sistema, que define a abrangência de atuação do usuário;
- V. **Transação**: um programa executável do Sistema;
- VI. **Confidencialidade**: princípio de segurança que estabelece restrições ao acesso e à utilização da informação;
- VII. **Integridade**: princípio de segurança que trata da confiabilidade da informação;
- VIII. **Disponibilidade**: princípio de segurança que trata da entrega tempestiva da informação a usuários e processos autorizados;
- IX. **Acesso imotivado**: aquele realizado para fins estranhos às tarefas do usuário;

II – ACESSO LÓGICO AS INFORMAÇÕES EM SAÚDE

Artigo 2º - O acesso do usuário aos Sistemas informatizados de Saúde será feito mediante o uso privativo de senha pessoal e intransferível, sendo que a sua outorga não confere direito de acesso imotivado.

Artigo 3º - O acesso aos Sistemas Informatizados de Saúde deve ser sempre motivado por necessidade de serviço, com anuência expressa do superior hierárquico em linha direta.

Artigo 4º - Cabe aos Dirigentes das Unidades do HCFMUSP submeter aos respectivos Diretores Executivos ou de Departamentos, o pedido de concessão de SENHA aos seus subordinados, por absoluta necessidade de serviço.

Parágrafo Único - O acesso motivado compreende o conjunto de transações inerentes aos perfis estabelecidos pelo Centro de Informações e Análise para as Unidades do HCFMUSP.

Artigo 5º - Compete aos Diretores Executivos ou de Departamentos autorizar as concessões de Senha e enviar os pedidos ao Centro de Informações e Análise para providências de cadastramento e habilitação.

Artigo 6º - Os casos excepcionais, não enquadrados no disposto do Artigo 4º, mas que justifiquem a concessão de SENHAS, deverão ser submetidos ao Superintendente para análise e autorização.

Artigo 7º - Caberá ao Centro de Informações e Análise, definir as condições para adoção, manutenção e controle do acesso lógico aos Sistemas Informatizados de Saúde, visando:

- I. proteger as informações dos Sistemas Informatizados de Saúde

contra o uso não autorizado;

II. auxiliar na detecção de violações de segurança;

III. assegurar recuperação nas situações de falha;

IV. permitir contabilização individual do uso do Sistema;

V. preservar os dados relativos às transações realizadas nos Sistemas, com a identificação do usuário, local, data e horário de acesso.

Artigo 8º - Os usuários disporão da SENHA após assinatura do Termo de Responsabilidade, conforme modelo anexo.

Artigo 9º - Os usuários ao acessarem os Sistemas serão informados, na tela inicial, de suas responsabilidades quanto ao uso de Sistemas e informações, bem como sobre as penalidades aplicáveis quanto ao uso indevido de suas senhas.

III – RESPONSABILIDADES INSTITUCIONAIS E FUNCIONAIS

Artigo 10º - Constitui-se em responsabilidade de todos os usuários cuidar da integridade, confidencialidade e disponibilidade dos dados, informações e Sistemas, devendo comunicar por escrito à chefia imediata quaisquer irregularidades, desvios ou falhas identificadas.

§ 1º - O acesso à informação não garante direito sobre a mesma nem confere autoridade para liberar acesso a outras pessoas.

§ 2º - Os usuários e os cadastradores devem manter suas senhas de acesso secretas, não podendo deixar qualquer sistema em condições de ser acessado por terceiros.

Artigo 11º - É de responsabilidade da chefia imediata iniciar ação corretiva apropriada para corrigir os desvios com relação às normas desta Ordem Conjunta de Serviço ou procedimentos de segurança dentro de sua área de atuação, comunicando o fato ao Diretor Executivo ou Departamento.

Artigo 12º - O descumprimento das disposições desta Ordem Conjunta de Serviço caracterizará infração funcional, a ser apurada em processo administrativo disciplinar, sem prejuízo da responsabilidade penal e civil.

Artigo 13º - O acesso imotivado do usuário aos Sistemas Informatizados do HCFMUSP constitui, sem prejuízo da responsabilidade civil e penal, infração funcional de falta de zelo e dedicação às atribuições do cargo ou função-atividade e descumprimento de normas legais ou regulamentares.

Artigo 14º - Ressalvadas as hipóteses de requisições legalmente autorizadas, constitui infração funcional a revelação de segredo do qual se apropriou em razão do cargo ou função-atividade, e crime tipificado no Código Penal, a divulgação, a terceiros, de informações dos Sistemas Informatizados protegidas pelo sigilo, sujeitando o infrator à penalidade de demissão.

Artigo 15º - Sem prejuízo da responsabilidade penal e civil e de outras infrações disciplinares, constitui falta de zelo e dedicação às atribuições do cargo ou função-atividade o descumprimento de normas legais e regulamentares, não proceder servidor com o devido cuidado na guarda e utilização da senha ou emprestá-la a outro servidor, ainda que habilitado.

IV – DISPOSIÇÕES FINAIS

Artigo 16º - Cabe ao Centro de Informações e Análises gerenciar o processo de implantação e de aplicação das normas desta Ordem Conjunta de Serviço, expedir instruções complementares, bem como realizar periodicamente auditoria de segurança nos ambientes operacionais e nos Sistemas de Informação em Saúde no HCFMUSP.

Artigo 17º - Os contratos de prestação de serviço relacionados aos sistemas informatizados de saúde no HCFMUSP devem conter cláusulas que viabilizem a adoção e manutenção das normas de segurança. Esta Ordem Conjunta de Serviço entra em vigor na data de sua publicação, revogando-se as disposições em contrário. São Paulo, 25 /09/ 1.998

C

Glossário de definições matemáticas da notação Z

Este anexo reproduz algumas das definições matemáticas da notação Z usadas nas especificações dos capítulos 2 e 3. A referência completa à notação é encontrada em Spivey (1998).

Conjunto vazio

$$\emptyset[X] == \{x : X \mid \text{false}\}$$

Conjunto potência

$$\mathbb{P} T == \{S \mid S \subseteq T\}$$

Relação binária

$$X \leftrightarrow Y == \mathbb{P}(X \times Y)$$

Maplet

$[X, Y]$	
$_ \mapsto _ : X \times Y \rightarrow X \times Y$	
$\forall x : X, y : Y \bullet x \mapsto y = (x, y)$	

Domínio e faixa de uma relação

[X, Y]
$\text{dom} : (X \leftrightarrow Y) \rightarrow \mathbb{P}X$
$\text{ran} : (X \leftrightarrow Y) \rightarrow \mathbb{P}Y$
$\forall R: X \leftrightarrow Y \bullet$ $\text{dom } R = \{x: X, y: Y \mid x \underline{R} y \bullet x\} \wedge$ $\text{ran } R = \{x: X, y: Y \mid x \underline{R} y \bullet y\}$

Função parcial

$$X \rightarrow Y == \{f: X \leftrightarrow Y \mid (\forall x: X, y_1, y_2: Y \bullet (x \mapsto y_1) \in f \wedge (x \mapsto y_2) \in f \Rightarrow y_1 = y_2)\}$$

Função total

$$X \rightarrow Y == \{f: X \rightarrow Y \mid \text{dom } f = X\}$$

Funções para projeção de pares ordenados

[X, Y]
$\text{first}: X \times Y \rightarrow X$
$\text{second}: X \times Y \rightarrow Y$
$\forall x: X, y: Y \bullet$ $\text{first}(x, y) = x \wedge \text{second}(x, y) = y$

Relação de ordem parcial

[X]
$\succeq : X \leftrightarrow X$
$\forall a: X \bullet a \succeq a \in X$ $\forall a \succeq b, b \succeq a: X \leftrightarrow X \bullet a = b$ $\forall a \succeq b, c \succeq d: X \leftrightarrow X \mid b = c \bullet a \succeq d \in X \leftrightarrow X$

União e intersecção generalizada de conjuntos

[X]
$\cup, \cap : \mathbb{P}(\mathbb{P}X) \rightarrow \mathbb{P}X$
$\forall A: \mathbb{P}(\mathbb{P}X) \bullet$ $\cup A = \{x: X \mid (\exists S: A \bullet x \in S)\} \wedge$ $\cap A = \{x: X \mid (\forall S: A \bullet x \in S)\}$

Operador relacional para anti-restrição de domínio

$[X, Y]$
$_ \triangleleft _ : \mathbb{P}X \times (X \leftrightarrow Y) \rightarrow (X \leftrightarrow Y)$
$\forall S : \mathbb{P}X; R : X \leftrightarrow Y \bullet$ $S \triangleleft R = \{x : X; y : Y \mid x \notin S \wedge x \underline{R} y \bullet x \mapsto y\}$

Operador relacional de sobreposição

$[X, Y]$
$_ \oplus _ : (X \leftrightarrow Y) \times (X \leftrightarrow Y) \rightarrow (X \leftrightarrow Y)$
$\forall Q, R : X \leftrightarrow Y \bullet$ $Q \oplus R = ((\text{dom } R) \triangleleft Q) \cup R$

Referências

ADATIA, R.; ARNI, F.; GABHART, K.; GRIFFIN, J.; JURIC, M. B.; LOTT, J.; MCALLISTER, T.; MULDER, A.; NAGARAJAN, N.; O'CONNOR, D.; OSBORNE, T.; SARANG, P. G.; TOST, A.; YOUNG, D.; BERRY, C. A. *Professional EJB*. Wrox, 2001. 1257 p.

AL-SALQAN, Y. Y. Security and confidentiality in healthcare informatics. In: SEVENTH IEEE INTERNATIONAL WORKSHOP ON ENABLING TECHNOLOGIES: INFRASTRUCTURE FOR COLLABORATIVE ENTERPRISES, 1998. *Proceedings...* p. 371-375.

ANDERSON, R. J. A security policy model for clinical information systems. In: 17TH IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 1996. *Proceedings...* p. 30-43.

APACHE SOFTWARE FOUNDATION. *The Apache Jakarta project: Apache Tomcat*. 2003. Disponível em: <<http://jakarta.apache.org/tomcat/>>. Acesso em: 18 nov. 2003.

ARNOLD, K.; GOSLING, J.; HOLMES, D. *The Java™ programming language*. 3. ed. Addison-Wesley, 2000, 704 p.

BACON, J.; LLOYD, M.; MOODY, K. Translating role-based access control policy within context. In: INTERNATIONAL WORKSHOP ON POLICIES FOR DISTRIBUTED SYSTEMS AND NETWORKS. *Lecture Notes in Computer Science*, v. 1995, Springer-Verlag, p. 107-119, jan. 2001.

BACON, J.; MOODY, K. Toward open, secure, widely distributed services. *Communications of the ACM*, v. 45, n. 6, p. 59-64, jun. 2002.

BACON, J.; MOODY, K.; YAO, W. A model of OASIS role-based access control and its support for active security. *ACM Transactions on Information and System Security*, v. 5, p. 492-540, nov. 2002.

_____. Access control and trust in the use of widely distributed services. *Software – Practice and Experience*, v. 33, p. 375-394, fev. 2003.

BELOKOSZTOLSZKI, A.; EYERS, D. M.; MOODY, K. Policy contexts: controlling information flow in parameterised RBAC. In: FOURTH INTERNATIONAL WORKSHOP ON POLICIES FOR DISTRIBUTED SYSTEMS AND NETWORKS, 2003. *Proceedings...*

BERNSTEIN, P. A. Middleware: a model for distributed system services. *Communications of the ACM*, v. 39, n. 2, p. 86-96, fev. 1997.

BERTINO, E.; BONATTI, E.; FERRARI, E. TRBAC: a temporal role-based access control model. *ACM Transactions on Information and System Security*, v. 4, p. 191-233, ago. 2001.

BERTINO, E.; JAJORDIA, S.; SAMARATI, P. A flexible authorization mechanism for relational data management systems. *ACM Transactions on Information Systems*, v. 17, n. 2, p. 101-140, abr. 1999.

BEZNOSOV, K. *Engineering access control for distributed enterprise applications*. 2000. 215 p. Tese Doutorado – Florida International University, Miami, Florida, EUA.

BEZNOSOV, K.; DENG, Y. A framework for implementing role-based access control Using CORBA Security Service. In: 4th ACM WORKSHOP ON ROLE-BASED ACCESS, 1999. *Proceedings...* p. 19-30.

BEZNOSOV, K.; DENG, Y.; BLAKLEY, B.; BURT, C.; BARKLEY, J. A resource access decision service for CORBA-based distributed systems. In: 15TH ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, 1999. *Proceedings...* p. 310-319.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. Interactions diagrams. In: _____. *The unified modeling language user guide*. Addison-Wesley, 1999, cap. 18, p. 243-256.

BORLAND SOFTWARE CORPORATION. *Visibroker developers guide: version 5.2.1*. Scotts Valley, CA, 2003, 596 p.

BRASIL. Constituição (1988). *Constituição da República Federativa do Brasil*. São Paulo, SP: Saraiva, 2001.

BREWER, D. F. C.; NASH, M. J. The Chinese wall security policy. In: IEEE SYMPOSIUM ON SECURITY AND POLICY, 1989. *Proceedings...* p. 206-214.

BRINKLEY, D. L.; SCHELL, R. R. Concepts and terminology for computer security. In: ABRAMS, M. D.; JAJODIA, S.; PODELL, H. J. (Ed.). *Information security: an integrated collection of essays*. Los Alamitos, CA: IEEE Computer Society Press, jan. 1995. p. 40-97.

BROSE, G.; NOFFKE, N. *JacORB 1.4 programming guide*. 2002. 90 p. Disponível em: <http://www.jacorb.org/docs/ProgrammingGuide_1_4_1.pdf>. Acesso em: 14 nov. 2003.

CANADIAN INSTITUTE FOR HEALTH INFORMATION. *Privacy and confidentiality of health information at CIHI: principles and policies for the protection of health information*. 2. ed. Ottawa, ON: Canadian Institute for Health Information, 1999. 16 p.

CELKO, J. *Joe Celko's SQL for smarties: advanced SQL programming*. 2. ed. Morgan Kaufmann, 1999.

CLARK, D.; WILSON, D. A comparison of commercial and military computer security policies. In: IEEE SYMPOSIUM ON SECURITY AND POLICY, 1987. *Proceedings...* p. 184-194.

CLOCKSIN, W. F.; MELLISH, C. S. *Programming in Prolog*. 2. ed. Springer-Verlag, 1984.

CONSELHO FEDERAL DE MEDICINA (Brasil). Resolução CFM nº 1246/88. Código de Ética Médica. 3. ed. Brasília, DF: Conselho Federal de Medicina, 1996. 80 p.

CONSELHO FEDERAL DE MEDICINA (Brasil). Resolução CFM nº 1.638/2002a. Define prontuário médico e dá outras providências. Brasília, DF: Conselho Federal de Medicina, 10 jul. 2002. Disponível em: <http://www.cfm.org.br/ResolNormat/Numerico/1638_2002.htm>. Acesso em: 10 set. 2002.

CONSELHO FEDERAL DE MEDICINA (Brasil). Resolução CFM nº 1.639/2002b. Dispõe sobre as normas técnicas para uso de sistemas informatizados para guarda e manuseio do prontuário médico e dá outras providências. Brasília, DF: Conselho Federal de Medicina, 10 jul. 2002b. Disponível em: <http://www.cfm.org.br/ResolNormat/Numerico/1639_2002.htm>. Acesso em: 10 set. 2002.

COVINGTON, M. J.; FOGLA, P.; ZHAN, Z.; AHAMAD, M. A context-aware security architecture for emerging applications. In: 18TH ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, 2002. *Proceedings...* Disponível em: <<http://www.acsac.org/2002/papers/71.pdf>>. Acesso em: 31 jul. 2003.

COVINGTON, M. J.; LONG, W.; SRINIVASAN, S.; DEY, A. K.; AHAMAD, M.; ABOWD, G. D. Securing context-aware applications using environment roles. In: SIXTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2001. *Proceedings...* p. 10-20.

COYNE, E. J. Role engineering. In: FIRST ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 1996. *Proceedings...*

CRAMPTON, J.; LOIZOU, G. Administrative scope and role hierarchy operations. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 145-154.

DATASUS. Página institucional de apresentação do DATASUS. Ministério da Saúde – Secretaria Executiva. 2003. Disponível em: <<http://www.datasus.gov.br/>>. Acesso em: 27 nov. 2003.

DENLEY I.; WESTON SMITH S. Privacy in clinical information systems in secondary care. *British Medical Journal*, v. 318, p. 1328-1330, mai. 1999. Seção *Information in practice*.

DEY, A. K. *Providing architectural support for building context-aware applications*. 2000. 170 p. Tese Doutorado – Georgia Institute of Technology, Atlanta, Georgia, EUA.

DIERKS, T.; ALLEN, C. *The TLS protocol version 1.0*. Internet Engineering Task Force, jan. 1999. 80 p. Disponível em: <<http://www.ietf.org/rfc/rfc2246.txt>>. Acesso em: 05 set. 2002.

EPSTEIN, P.; SANDHU, R. Towards a UML based approach to role engineering. In: FOURTH ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 1999. *Proceedings...* p. 135-143.

_____. Engineering of role/permission assignments. In: 17TH ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, 2001. *Proceedings...* p. 127-136.

FERNANDES, L. *Oracle 9i para desenvolvedores: curso completo*. Axcel Books do Brasil, 2002a.

_____. PL/SQL. In: _____. *Oracle 9i para desenvolvedores: curso completo*. Axcel Books do Brasil, 2002b.

FERNANDEZ, E. B.; HAWKINS, J. C. Determining role rights from use cases. In: SECOND ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 1997. *Proceedings...* p. 121-125.

FERRAILOLO, D. F.; BARKLEY, J. S.; KUHN, D. R. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security*, v. 2, n. 1, p. 34-64, fev. 1999.

FERRAILOLO, D. F.; GILBERT, D. M.; LYNCH, N. An examination of federal and commercial access control policy needs. In: 16TH NIST-NSA NATIONAL COMPUTER SECURITY CONFERENCE, 1993. *Proceedings...* p. 107-116.

FERRAILOLO, D. F.; SANDHU, R.; GAVRILA, S.; KUHN, D. R.; CHANDRAMOULI, R. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security*, v. 4, n. 3, p. 224-274, ago. 2001.

FIALES, V. R.; NARDON F. B.; FURUIE, S. S. Construção de um serviço de identificação de pacientes. *Revista Eletrônica de Iniciação Científica*, v. 1, n. 1, ago. 2001. Disponível em: <<http://www.sbc.org.br/reic/edicoes/2001e1/>>. Acesso em: 09 set. 2002.

FRANCISCONI, C. F.; GOLDIM, J. R. Aspectos bioéticos da confidencialidade e privacidade. In: Costa, S. I. F.; Oselka, G. e Garrafa, V. (Org.). *Iniciação à bioética*. Brasília, DF: Conselho Federal de Medicina, 1998. p. 269-284.

FRANÇA, G. V. Fundamentos legais e filosóficos do segredo médico. *Revista Electrónica de Derecho y Bioética*, n. 1, jan./mar. 2002. Disponível em: <<http://www.filosofiyderecho.com/redb/numero1.htm>>. Acesso em: 19 jul. 2002.

GAGNON, E. M.; HENDREN, L. J. SableCC, an object-oriented compiler framework. In: TECHNOLOGY OF OBJECT-ORIENTED LANGUAGES, 1998. *Proceedings...* p. 140-

154. Disponível em: <<http://www.sablecc.org/tools-98.pdf>>. Acesso em: 16 nov. 2003.

GEORGIADIS, C. K.; MAVRIDIS, I. K.; NIKOLAKOPOULOU, G.; PANGALOS, G. I. Implementing context and team based access control in healthcare intranets. *Medical Informatics and the Internet in Medicine*, v. 28, n. 3, p. 185-201, set. 2002.

GEORGIADIS, C. K.; MAVRIDIS, I.; PANGALOS, G.; THOMAS, R. Flexible team-based access control using contexts. In: SIXTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2001. *Proceedings...* p. 21-27.

GLIGOR, V. D.; GAVRILA, S. I.; FERRAILOLO, D. On the formal definition of separation-of-duty policies and their composition. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY, 1998. *Proceedings...* p. 172-183.

GORDIN, D. N. Managing graph(ical) complexity with Raisin and its category explorer. In: SECOND INTERNATIONAL SYMPOSIUM ON SMART GRAPHICS, 2002. *Proceedings...* p. 136-139.

GRIMSON, J.; GRIMSON, W.; HASSELBRING, W. The system integration challenge in health care. *Communications of the ACM*, v. 43, n. 6, p. 49-55, jun. 2000.

HARTMAN, B.; FLINN, D. J.; BEZNOSOV, K. *Enterprise security with EJB and CORBA*. Wiley, 2001, 364 p.

HEXSEL, R. A. *Software Livre*. Versão 1.0. Departamento de Informática, Universidade Federal do Paraná. 13 jan. 2003. Disponível em: http://www.inf.ufpr.br/info/techrep/RT_DINF004_2002.pdf.gz. Acesso em: 16 dez. 2003.

HORN, A. On sentences which are true of direct unions of algebras. *Journal of Symbolic Logic*, v. 16, p. 14-21, 1951.

HOWES, T. A.; SMITH, M. C.; GOOD, G. S. *Understanding and deploying LDAP directory services*, MacMillan Technical Publishing, 1999.

HUSTON, T. Security issues for implementing of e-medical records. *Communications of the ACM*, v. 44, n. 8, p. 89-94, set. 2001.

ISO Panel JTC1/SC22/WG19. Z notation: final committee draft, CD 13568.2. 24 ago. 1999. 182 p. Disponível em: <ftp://ftp.cs.york.ac.uk/hise_reports/cadiz/ZSTAN/fcd.pdf>. Acesso em: 21 ago. 2002.

ITU-T. Recommendations series X.509: information technology – open systems interconnection – the directory: authentication framework, 1993.

ITU-T. Recommendations series X.5xx: information technology – open systems interconnection – the directory, 1992.

JAIN, A.; HONG, L.; PANKANTI, S. Biometric identification. *Communications of the ACM*, v. 43, n. 2, p. 91-98, fev. 2000.

JOSHI, J. B. D.; AREF, W. G.; GHAFOR, A.; SPAFFORD, E. H. Security models for web-based applications. *Communications of the ACM*, v. 44, n. 2, p. 38-44, fev. 2001.

JOSHI, J. B. D.; BERTINO, E.; GHAFOR, A. Hybrid role hierarchy for generalized temporal role based access control model. In: 26TH ANNUAL INTERNATIONAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE, 2002. *Proceedings...* p. 951-956.

JOSHI, J. B. D.; BERTINO, E.; SHAFIQ, B.; GHAFOR, A. Dependencies and separation of duty constraints in GTRBAC. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 51-64.

KAIHARA, S. Realisation of the computerised patient record: relevance and unsolved problems. *International Journal of Medical Informatics*, v. 49, p. 1-8, mar. 1998.

KERN, A.; KUHLMANN, M.; SCHAAD, A.; MOFFETT, J. Observations on the role life-cycle in the context of enterprise security management. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 43-51.

KERN, A.; SCHAAD, A.; MOFFETT, J. An administration concept for the enterprise role-based access control model. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 3-11.

KUHLMANN, M.; SHOHAT, D.; SCHIMPF, G. Role mining – revealing business roles for security administration using data mining technology. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 179-186.

KUMAR, A.; KARNIK, N.; CHAFLE, G. Context sensitivity in role-based access control. *ACM SIGOPS Operating Systems Review*, v. 36, n. 3, p. 53-66, jul. 2002.

LAMPORT, L. Password authentication with insecure communication. *Communications of the ACM*, v. 24, n. 11, p. 770-772, nov. 1981. Seção *Technical note – operating systems*.

LARKIN, J. H.; SIMON, H. A. Why a diagram is (sometimes) worth them thousand words. *Cognitive Science*, v. 11, p. 65-99, 1987.

LEE, R.; SELIGMAN, S. *JNDI API tutorial and reference: building directory-enabled Java*. Addison-Wesley, 2000, 848 p.

LEWANDOWSKI, S. M. Frameworks for component-based client/server computing. *ACM Computing Surveys*, v. 30, n. 1, p. 3-27, mar. 1998.

LIMA, E.; REIS, E. *C# e .NET: guia do desenvolvedor*. Campus, 2002. 358 p.

LIN, A.; BROWN, R. The application of security policy to role-based access control and the common data security architecture. *Computer Communications*, v. 23, p. 1584-1593, 2000.

LINUX ONLINE, INC. *Linux Online!*. Sítio que disponibiliza acesso a documentação e códigos fontes e binário das distribuições do sistema operacional Linux. 2003 Disponível em: <<http://www.linux.org/>>. Acesso em: 13 nov. 2003.

LONGSTAFF, J.; LOCKYER, M.; NICHOLAS, J. The Tees confidentiality model: an authorisation model for identities and roles. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 125-133.

LUPU, E. C.; SLOMAN, M. Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering*, v. 25, n. 6, p. 852-869, nov./dez. 1999.

MAGIC SOFTWARE ENTERPRISES. *The Magic® guide to application partitioning & client/server*. Magic enterprise edition version 8. Magic Software Enterprise Ltd., 2000.

MALAMATENIOU, F.; VASSILACOPOULOS, G.; TSANAKAS, P. A workflow-based approach to virtual patient record security, *IEEE Transactions on Information Technology in Biomedicine*, v. 2, n. 3, p. 139-145, set. 1998.

MARIN, H. F.; MASSAD, E.; AZEVEDO-NETO, R. S. Prontuário eletrônico do paciente: definições e conceitos. In: _____ (Ed.). *O prontuário eletrônico do paciente na assistência, informação e conhecimento médico*. Washington, DC: Organização Pan-Americana de Saúde, 2003. 213 p.

McDANIEL, P. On Context in Authorization Policy. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 80-89.

MONJIAN, B. *PostgreSQL: introduction and concepts*. Addison-Wesley, 2000. 462 p.

MORENO, R. A. FIALES, V. R.; FURUIE, S. S. Clinical image access service: achieving interoperability using CORBA. In: SPIE INTERNATIONAL SYMPOSIUM ON MEDICAL IMAGING, v. 4685, 2002. *Proceedings...* p. 217-227.

MUFTIC, S.; PATEL, A.; SANDERS, P.; COLON, R.; HEIJNSDIJK, J.; PULKKINEN, U. *Security architecture for open distributed systems*. Wiley, 1993. 281 p.

MYERS, J. *Simple authentication and security layer*. Internet Engineering Task Force, out. 1997. 16 p. Disponível em: <<http://www.ietf.org/rfc/rfc2222.txt>>. Acesso em: 05 set. 2002.

NATIONAL ACADEMY OF SCIENCES. *For the record: protecting electronic health information*. Washington, DC: National Academy Press, 1997. 264 p.

NATIONAL HEALTH SERVICE – Information Authority. *Caring for information: model for the future*. Versão 1.0 para consulta. Departamento de Saúde da Grã-Bretanha, 14 out. 2002. Disponível em: <<http://www.nhsia.nhs.uk/confidentiality/pages/consultation/>>. Acesso em: 06 ago. 2003.

NEUMANN, G.; STREMBECK, M. Design and implementation of a flexible RBAC-service in an object-oriented scripting language. In: EIGHTH ACM CONFERENCE ON COMPUTER AND COMMUNICATION SECURITY, 2001. *Proceedings...* p. 58-67.

_____. A scenario-driven role engineering process for functional RBAC roles. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 33-42.

_____. An approach to engineer and enforce context constraints in an RBAC environment. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 65-79.

NEUMANN, G.; ZDUM, U. XOTcl, an object-oriented scripting language. In: SEVENTH USENIX TCL/TK CONFERENCE, 2000. *Proceedings...* Disponível em: <<http://nm.wu-wien.ac.at/research/publications/xotcl-tclconf.pdf>>. Acesso em: 12 ago. 2003.

NEWCOMER, E. *Understanding web services*. XML, WSDL, SOAP and UDDI. Addison Wesley, 2002. 368 p.

OBELHEIRO, R. R.; FRAGA, J. S.; WESTPHALL, C. M. Controle de acesso baseado em papéis para o modelo CORBA de segurança. In: 19^o SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, mai. 2001. *Anais...*

OBJECT MANAGEMENT GROUP. *Lexicon query service specification*, versão 1.0. 2000a. 182 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/2000-06-31>>. Acesso em: 19 ago. 2002.

_____. *Resource access decision facility*, 2000b. 76 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?dtc/00-08-06>>. Acesso em: 21 ago. 2002.

_____. *Clinical image access service specification*. Mar. 2001a. 108 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?dtc/2001-10-04>>. Acesso em: 07 nov. 2003.

_____. *Clinical observation access service specification*, versão 1.0. 2001b. 260 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/2001-04-06>>. Acesso em: 19 ago. 2002.

_____. *CORBA security service specification*, versão 1.7. Mar. 2001c. 434 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/01-03-08>>. Acesso em: 21 ago. 2002.

_____. *Person identification service specification*, versão 1.1. 2001d. 138 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/2001-04-04>>. Acesso em: 19 ago. 2002.

_____. *Common object request broker architecture: core specification*, versão 3.0.2. 2002a. 1166 p. Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/02-12-02>>. Acesso em: 12 nov. 2003.

_____. Fault Tolerant CORBA. In: _____. *Common object request broker architecture: core specification*, versão 3.0.2. 2002b. cap. 23, p. 23-1-106 Disponível em: <<http://www.omg.org/cgi-bin/doc?formal/02-12-02>>. Acesso em: 12 nov. 2003.

OH, S.; PARK, S. Enterprise model as a basis of administration on role-based access control. In: THIRD INTERNATIONAL SYMPOSIUM ON COOPERATIVE DATABASE SYSTEMS FOR ADVANCED APPLICATIONS, 2001. *Proceedings...* p. 150-158.

OH, S.; SANDHU, R. A model for role administration using organization structure. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 155-168.

OLIVEIRA, J. A. P. Sigilo ou segredo médico: a ética e o direito. *Revista Bioética*, Brasília, DF, v. 9, n. 2, p. 141-148, 2001. Seção Pareceres e Resoluções.

OPENLDAP FOUNDATION. *The OpenLDAP project*. Sítio que disponibiliza a documentação e códigos fontes e binário do produto. 2003 Disponível em: <<http://www.openldap.org/>>. Acesso em: 13 nov. 2003.

OSBORN, S.; SANDHU, R.; MUNAWER, Q. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Transactions on Information Systems Security*, v. 3, n. 2, p. 85-106, mai. 2000.

OUSTERHOUT, J. K. *Tcl and the Tk toolkit*. Addison-Wesley, 1994. 480 p.

PERWAIZ, N.; SOMMERVILLE, I. Structured management of role-permission relationships. In: SIXTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2001. *Proceedings...* p. 163-169.

PHILIPS, D. *The software project manager's handbook: principles that work at work*. Los Alamitos, CA: IEEE Computer Society Press, 1998. 403 p.

POTTER, B.; SINCLAIR, J.; TILL, D. *An introduction to formal specification and Z*. Prentice Hall, 1991. 304 p.

PUTMAN, J. *Architecting with RM-ODP*. Prentice Hall, 2001. 834 p.

RABITTI, F.; BERTINO, E.; KIM, W.; WOELK, D. A model of authorization for next-generation database systems. *ACM Transactions on Database Systems*, v. 16, p. 88-131, mar. 1991.

RINDFLEISCH, T. C. Privacy, information technology, and health care. *Communications of the ACM*, v. 40, n. 8, p. 93-100, ago. 1997.

RODRIGUES, R. J.; WILSON, P.; SCHANZ, S. J. *The regulation of privacy and data protection in the use of electronic health information: an international perspective and reference source on regulatory and legal issues related to person-identifiable health databases*. Washington, DC: Organização Pan-Americana de Saúde, 2001. 217 p.

ROECKLE, H.; SCHIMPF, G.; WEIDINGER, R. Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization. In: FIFTH ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 2000. *Proceedings...* p. 103-110.

- SABLECC. *SableCC*. Sítio que disponibiliza a documentação e códigos fontes e binário do produto. 2003. Disponível em: <http://www.sablecc.org/>>. Acesso em: 16 nov. 2003.
- SACARDO, D. P.; FORTES, P. A. C. Desafios para a preservação da privacidade no contexto da saúde. *Revista Bioética*, Brasília, DF, v. 8, n. 2, p. 307-322, 2000. Seção Temas em Debate.
- SANDHU, R. S. Lattice-based access control models. *IEEE Computer*, p. 9-19, nov. 1993.
- _____. Roles versus groups. In: FIRST ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 1996. *Proceedings...* p. I25-I26.
- SANDHU, R. S.; BHAMIDIPATI, V.; MUNAWER, Q. The ARBAC97 model for role-based administration of roles. *ACM Transactions on Information and System Security*, v. 2, p. 105-135, fev. 1999.
- SANDHU, R. S.; COYNE, E. J.; YOUMAN, C. E. Role-based access control models. *IEEE Computer*, p. 38-47, fev. 1996.
- SANDHU, R. S.; FERRAILOLO, D.; KUHN, R. The NIST model for role-based access control: towards a unified standard. In: FIFTH ACM WORKSHOP ON ROLE-BASED ACCESS CONTROL, 2000. *Proceedings...* p. 47-63.
- SANDHU, R. S.; MUNAWER, Q. The ARBAC99 model for administration of roles. In: 15TH ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, 1999. *Proceedings...* p. 229-238.
- SANDHU, R. S.; SAMARATI, P. Access control: principles and practice. *IEEE Communications Magazine*, p. 40-48, set. 1994.
- _____. Authentication, access control, and audit. *ACM Computing Surveys*, v. 28, n. 1, p. 241-243, mar. 1996.
- SANTOS, M.; FURUIE, S. S. High performance web viewer for cardiac images. In: SPIE INTERNATIONAL SYMPOSIUM ON MEDICAL IMAGING, v. 5371, fev. 2004. *Proceedings...*
- SCHAAD, A.; MOFFETT, J.; JACOB, J. The role-based access control system of a European bank: a case study and discussion. In: SIXTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2001. *Proceedings...* p. 3-9.
- SCHOENBERG R.; SAFRAN C. Internet based repository of medical records that retains patient confidentiality. *British Medical Journal*, v. 321, p. 1199-1203, nov. 2000. Seção *Information in practice*.
- SHELFER, K. M.; PROCACCINO, J. D. Smart card evolution. *Communications of the ACM*, v. 45, n. 7, p. 83-88, jul. 2002.
- SHIN, D.; AHN, G.-J.; CHO, S.; JIN, S. On modeling system-centric information for role engineering. In: EIGHTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2003. *Proceedings...* p. 169-178.

SMITH, E.; ELOFF, J. H. P. Security in health-care information systems – current trends. *International Journal of Medical Informatics*, v. 54, p. 39-54, abr. 1999.

SOMMERER, A. The extension mechanism. In: _____. *The Java tutorial*. Sun™ Microsystems, 2002. Disponível em: <<http://java.sun.com/docs/books/tutorial/ext/index.html>>. Acesso em: 09 set. 2002.

SOMMERVILLE, I. *Software engineering*. 5. ed. Addison-Wesley, 1996. 742 p.

SPIVEY, J. M. *The Z notation: a reference manual*. 2. ed. Prentice Hall, 1998. 158 p. Disponível em: <<http://spivey.oriel.ox.ac.uk/~mike/zrm/zrm.pdf>>. Acesso em: 02 dez. 2003.

STALLINGS, W. *Cryptography and network security: principles and practice*. 2. ed. Prentice Hall, 1999. 569 p.

SUN MICROSYSTEMS. *Java 2 platform, standard edition*. 2003a. Disponível em: <<http://java.sun.com/j2se/>>. Acesso em: 01 ago. 2003.

_____. *Java server pages*. 2003b. Disponível em: <<http://java.sun.com/products/jsp/>>. Acesso em: 05 ago. 2003.

_____. *Java servlet technology*. 2003c. Disponível em: <<http://java.sun.com/products/servlet/>>. Acesso em: 05 ago. 2003.

UNITED STATES DEPARTMENT OF HEALTH AND HUMAN SERVICES - Office of the Secretary. *Health insurance reform: security standards*. Regra final para atender a Lei HIPAA – Health Insurance Portability and Accountability Act of 1996. Washington, DC: Federal Register, v. 68, p. 8333-8381, fev. 2003.

VARADHARAJAN, V.; CALVELLI, C. An access control model and its use in representing mental health application access policy. *IEEE Transactions on Knowledge and Data Engineering*, v. 8, n. 1, p. 81-95, fev. 1996.

W3 CONSORTIUM. *Extensible markup language (XML)*. 2. ed., out. 2000. Disponível em: <<http://www.w3.org/XML/>>. Acesso em: 10 set. 2002.

WILIKENS, M.; FERITI, S.; SANNA, A.; MASERA, M. A context-related authorization and access control method based on RBAC: a case study from the health care domain. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 117-124.

YEONG, W.; HOWES T.; KILLE, S. *Lightweight directory access protocol*. Internet Engineering Task Force, mar. 1995. 22 p. Disponível em: <<http://www.ietf.org/rfc/rfc1777.txt>>. Acesso em: 21 ago. 2002.

ZHANG, L.; AHN, G.-L.; CHU, B.-T. A role-based delegation framework for healthcare information systems. In: SEVENTH ACM SYMPOSIUM ON ACCESS CONTROL MODELS AND TECHNOLOGIES, 2002. *Proceedings...* p. 125-134.

Produção bibliográfica e técnica

Artigos Publicados em Periódicos

Completo

MOTTA, G. H. M. B.; FURUIE, S. S. *A Contextual Role-Based Access Control Authorization Model for Electronic Patient Record*. IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE, v. 7, n. 3, p. 202-207, 2003. ISSN: 1089-7771.

MOTTA, G. H. M. B.; FURUIE, S. S. *Um Modelo de Autorização e Controle de Acesso para o Prontuário Eletrônico de Pacientes em Ambientes Abertos e Distribuídos*. REVISTA BRASILEIRA DE ENGENHARIA BIOMÉDICA, Rio de Janeiro, v. 17, n. 3, p. 141-150, 2001. ISSN: 1517-3151.

Resumos

MOTTA, G. H. M. B.; FURUIE, S. S.; NARDON, F. B.; GUTIERREZ, M. A. *A Role-based Access Authorization to Clinical and Image Data*. In: RSNA 2001, Chicago – IL. Radiology, The Radiological Society of North America, 2001, v. 221, n. (P), p. 761-761. ISSN: 0033-8419.

*Trabalhos Publicados em Anais de Eventos**Completo*

MOTTA, G. H. M. B.; FURUIE, S. S.; GUTIERREZ, M. A. *A Contextual Role-Based Access Control Model for Electronic Patient Record in Open Distributed Environment*. In: WORKSHOP EM SEGURANÇA (SAFETY WORKSHOP): SISTEMAS COMPUTACIONAIS EM APLICAÇÕES CRÍTICAS – LADC 2003 – 1st LATIN AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING, 2003, São Paulo – SP. Anais do Workshop em Segurança (Safety Workshop): Sistemas Computacionais em Aplicações Críticas. São Paulo: Escola Politécnica da USP, 2003. p. 57-79. ISBN: 8586686247.

FURUIE, S. S.; REBELO, M. S.; MORENO, R. A.; SANTOS, M.; BERTOZZO, N.; MOTTA, G. H. M. B.; GUTIERREZ, M. A. *Integrating Medical Images and Clinical Information: InCor's Experience*. In: XII CONGRESSO INTERNACIONAL DE COMPUTACIÓN - CIC 2003, 2003, Cidade do México. Avances en: Ciencias de la Computación. Cidade do México: Instituto Politécnico Nacional - Centro de Investigación en Computación, 2003. v. 3, p. 67-72. ISBN: 9703600980.

MOTTA, G. H. M. B.; FURUIE, S. S. *MACA: Uma Ferramenta de Autorização e Controle de Acesso para o Prontuário Eletrônico de Pacientes*. In: CBIS'2002 - VIII CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE, 2002, Natal – RN. Anais do CBIS'2002 – VIII Congresso Brasileiro de Informática em Saúde, 2002. 6 p.

FURUIE, S. S.; REBELO, M. S.; GUTIERREZ, M. A.; MORENO, R. A.; NARDON, F. B.; MOTTA, G. H. M. B.; FIGUEIREDO, J. C. B.; BERTOZZO, N.; FIALES, V. *Prontuário Eletrônico em Ambiente Distribuído e Heterogêneo: a Experiência do InCor*. In: CBIS'2002 – VIII CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE, 2002, Natal – RN. Anais do CBIS'2002 - VIII Congresso Brasileiro de Informática em Saúde, 2002. 6 p.

FIGUEIREDO, J. C. B.; FURUIE, S. S.; GUTIERREZ, M. A.; MOTTA, G. H. M. B.; MORENO, R. A. *Sistema Compacto para Armazenamento e Distribuição de Imagens Médicas em Ambientes Clínico-Hospitalares*. In: CBIS'2002 – VIII CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE, 2002, Natal – RN.

Anais do CBIS'2002 – VIII Congresso Brasileiro de Informática em Saúde, 2002. 4 p.

MOTTA, G. H. M. B.; FURUIE, S. S. *Um Modelo de Autorização Contextual para o Controle de Acesso Baseado em Papéis*. In: XX SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES: WSEG'2002 – II WORKSHOP EM SEGURANÇA DE SISTEMAS COMPUTACIONAIS, 2002, Búzios – RJ. Anais do WSeg'2002 - II Workshop em Segurança de Sistemas Computacionais, Búzios – RJ. Sociedade Brasileira de Computação, 2002, p. 136-143.

MOTTA, G. H. M. B.; FURUIE, S. S.; NARDON, F. B.; GUTIERREZ, M. A.; YAMAGUTI, M. *Autorização e Controle de Acesso para o Prontuário Eletrônico do Paciente em Ambientes Abertos e Distribuídos: uma Proposta de Modelo e Arquitetura*. In: IX SIMPÓSIO BRASILEIRO DE COMPUTAÇÃO TOLERANTE A FALHAS: WSEG'2001 - WORKSHOP EM SEGURANÇA DE SISTEMAS COMPUTACIONAIS, 2001, Florianópolis – SC. Anais do WSeg'2001 - Workshop em Segurança de Sistemas Computacionais, Florianópolis – SC, Sociedade Brasileira de Computação, 2001, p. 92-97.

MOTTA, G. H. M. B.; FURUIE, S. S.; NARDON, F. B.; GUTIERREZ, M. A. *Considerações Sobre o Controle de Acesso ao Prontuário Eletrônico do Paciente*. In: CBIS'2000 – VII CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE, 2000, São Paulo. Anais do VII Congresso Brasileiro de Informática em Saúde, 2000. 5 p.

Relatórios Técnicos

MOTTA, G. H. M. B. *Controle de Acesso Baseado em Papéis para o Prontuário Eletrônico do Paciente em Ambientes Distribuídos e Abertos*. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, jul. 2001. 23 p. (Relatório Técnico)

MOTTA, G. H. M. B. *Middleware para Autenticação e Controle de Acesso (MACA) ao Prontuário Eletrônico do Paciente: projeto lógico*. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, ago. 2000. 40 p. (Relatório Técnico)

MOTTA, G. H. M. B. *Middleware para Autenticação e Controle de Acesso (MACA) ao Prontuário Eletrônico do Paciente: análise de requisitos e especificação fun-*

cional. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, jul. 2000. 18 p. (Relatório Técnico)

MOTTA, G. H. M. B. *Estratégias para Controle de Acesso ao Prontuário Eletrônico do Paciente na Arquitetura CORBA*. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, mai. 2000. 27 p. (Relatório Técnico)

Manuais

MOTTA, G. H. M. B. *MACA*: guia de instalação e configuração. Versão 3.2. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, jun. 2003. 102 p.

MOTTA, G. H. M. B. *MACA administrativo*: manual do usuário. Versão 3.2. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, jul. 2002. 48 p.

MOTTA, G. H. M. B. *MACA cliente*: guia do programador. Versão 3.2. São Paulo, SP: Unidade de Pesquisa e Desenvolvimento, Serviço de Informática, Instituto do Coração - HC-FMUSP, jul. 2002. 70 p.

Softwares

MOTTA, G. H. M. B.; FURUIE, S. S. *MACA – Middleware para Autenticação e Controle de Acesso*. Software registrado no INPI sob número 00046880 em 15 de julho de 2002.