Web Services de apoio a aplicações voltadas ao
trabalho em grupo

Victor Hugo Vieira

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito: 14/02/2006

Assinatura:_____

Web Services de apoio a aplicações voltadas ao trabalho em grupo

Victor Hugo Vieira

Orientador: *Profa. Dra. Renata Pontin de Mattos Fortes*

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências de Computação e Matemática Computacional.

USP - São Carlos

Fevereiro/2006

Dedicatória

Dedico aos meus pais, Sebastião e Cleide, pelo incondicional amor, apoio e incentivo, e
por sempre acreditarem em mim.

Agradecimentos

Agradeço a Deus, por estar sempre comigo.

À minha orientadora, Renata Pontin, pelos ensinamentos, dedicação e apoio que me foram concedidos.

Ao meu irmão, Cesar, pelo constante incentivo aos estudos e a esta carreira acadêmica.

À minha namorada, Greice, por todo amor, amizade e confiança a mim dedicados, e por estar sempre presente.

Aos amigos Luizão, Sergio, Carlos, Rivail, Bruno, André, Silvana, Daniel e Tanaka, pela ajuda e companheirismo.

A todos os funcionários e professores do ICMC, por tudo que me ensinaram e por estarem sempre dispostos a ajudar.

Agradeço também ao CNPq, pelo suporte financeiro, essencial a este trabalho.

Resumo

O desenvolvimento de trabalhos em grupo tornou-se uma tarefa cada vez mais comum nos dias atuais. Com isso, houve também um crescimento na utilização e no desenvolvimento de aplicações computacionais que implementam os processos de apoio à cooperação entre as pessoas, as quais possibilitam um considerável ganho de produtividade nas tarefas realizadas pelo grupo. A integração de várias dessas aplicações pode resultar em um acréscimo de funcionalidades, aumentando as possibilidades de uso e a flexibilidade para o usuário. Com o objetivo de agregar funcionalidades e melhorar os potenciais usos dos sistemas de *software*, aplicações integradas possuem, em geral, mais funcionalidades e maiores possibilidades de uso do que aplicações isoladas.

Assim, foi desenvolvido o WS4CSCW (*Web Services for Computer Supported Cooperative Work*), um *Web Service* que tem como objetivo facilitar a integração de ferramentas e sistemas CSCW. O WS4CSCW permite o gerenciamento de usuários, grupos, recursos, permissões e papéis de usuários e grupos, disponibilizando funcionalidades para aplicações voltadas ao trabalho em grupo, bem como para facilitar a integração entre essas aplicações, permitindo um co-funcionamento mais direto, consistente e coeso.

Abstract

Nowadays, the achievement of tasks made in groups has become a habitual concern by people. In this way, a rise of using and developing computer programs that support cooperative work process has been noted. The programs aim to provide productivity improvements on work made by groups. Integrating the software could result in increasing functionalities, augmenting both its use possibilities and flexibility. Integrated softwares have, in general, more features and use's potential than isolated applications.

As a result, WS4CSCW (Web Services for Computer Supported Cooperative Work) – a Web Service to assist CSCW application's integration was developed. WS4CSCW allows management of users, groups, resources, permissions, and user and group roles, by making available functionalities to applications for group work, making easy the integration among these applications and providing a straighter, more cohesive and consistent cooperation.

Sumário

Dedicatória	i
Agradecimentos	iii
Resumo	v
Abstract	vii
Sumário	ix
Lista de Figuras	xiii
Lista de Abreviaturas	xv
1 Introdução	1
2 CSCW	4
2.1 Considerações Iniciais	4
2.2 Trabalho Cooperativo	5
2.3 Histórico	8
2.4 <i>Groupware</i>	9
2.4.1 Definição	10
2.4.2 Informações Distribuídas	11
2.4.3 Características	12
2.4.4 Tipos de Groupware	17
2.4.5 Interfaces em <i>groupwares</i>	22
2.5 Aplicações CSCW Distribuídas	23
2.5.1 Requisitos de Aplicações CSCW Distribuídas	23
2.5.2 Arquitetura de Aplicações CSCW Distribuídas	24
2.6 Considerações Finais	25

3	Web Services	27
3.1	Considerações Iniciais	27
3.2	Histórico	28
3.3	Definição	29
3.4	Arquitetura	30
3.5	XML-RPC	33
3.6	SOAP	33
3.7	WSDL	35
3.8	UDDI	37
3.9	Considerações Finais	38
4	WS4CSCW	40
4.1	Considerações Iniciais	40
4.2	Conceitos	41
4.3	Cenário de Uso	45
4.4	Modelagem UML	46
4.5	Serviços do WS4CSCW	49
4.5.1	Contexto	49
4.5.2	Aplicação	50
4.5.3	Usuário	51
4.5.4	Grupo	54
4.5.5	Recurso	55
4.5.6	Papel	56
4.5.7	Permissão	58
4.6	Trabalhos relacionados	65
4.7	Cenário de Integração	70
4.7.1	Considerações Iniciais	70
4.7.2	Cenário 1: iClass e No Risk Planning	70
4.7.3	Cenário 2: Projeto SAFE	73
4.8	Considerações Finais	76
5	Conclusões	77
5.1	Publicações	78

5.2	Trabalhos Futuros	79
A	Especificação WSDL	81
	Referências Bibliográficas	99

Lista de Figuras

2.1	Gerenciamento centralizado	11
2.2	Gerenciamento distribuído	12
2.3	Gerenciamento híbrido	13
2.4	Funcionamento da Colaboração, refinamento de [Fuks et al., 2002]	14
2.5	Arquitetura de aplicações CSCW distribuídas [Cosquer et al., 2000]	25
3.1	Pilha de protocolos dos <i>Web Services</i> [Cerami, 2002]	31
3.2	Arquitetura básica de <i>Web Services</i> [W3C, 2002]	32
3.3	Papéis Básicos na Arquitetura de <i>Web Services</i> [Cerami, 2002]	33
4.1	Relacionamentos do modelo RBAC [Ferraiolo and Kuhn, 1992]	42
4.2	Exemplo do modelo de permissões usado no WS4CSCW	43
4.3	Cenários de uso do WS4CSCW	46
4.4	Diagrama de Classes do WS4CSCW	47
4.5	Diagrama de Casos de Uso do WS4CSCW	48
4.6	Interface <i>iClass</i> com o botão que aciona o <i>Web Service</i>	72
4.7	Cenário de uso: Integração <i>iClass</i> - <i>No Risk Planning</i>	72
4.8	Interface requisição de agendamento de compromissos	73
4.9	Interface do <i>framework</i> SAFE	75

Lista de Abreviaturas

ACID *Atomic, Consistent, Isolated, Durable*

BSCW *Basic Support for Cooperative Work*

CASE *Computer-Aided Software Engineering*

COM *Component Object Model*

CORBA *Common Object Request Broker Architecture*

CSCW *Computer Supported Cooperative Work*

DCOM *Distributed Component Object Model*

HTTP *Hypertext Transfer Protocol*

IDL *Interface Design Language*

LEICA *Loosely-coupled Environment for Integrating Collaborative Applications*

ORB *Object Request Broker*

OSS *Open Source Software*

P2P *Peer-to-peer*

RBAC *Role-Based Access Control*

RPC *Remote Procedure Call*

SAFE *Software Engineering Available for Everyone*

SMS *Short Messaging Service*

SOAP *Simple Object Access Protocol*

TCP *Transmission Control Protocol*

UDDI *Universal Description, Discovery and Integration*

URI *Uniform Resource Identifier*

WS4CSCW *Web Service for Computer Supported Cooperative Work*

WSDL *Web Service Description Language*

WSIF *Web Service Invocation Framework*

XML *eXtensible Markup Language*

Introdução

Há algumas décadas, sistemas CSCW (*Computer Supported Cooperative Work*) concentram esforços no suporte a várias atividades voltadas ao trabalho em grupo ou mesmo individuais, mas embutidas em múltiplas equipes com alguma meta em comum, dentro de organizações bem como entre organizações.

Devido à necessidade de discussão de problemas e tomada conjunta de decisões, bem como a divisão de uma grande tarefa em tarefas menores, a realização de atividades em grupo torna-se a cada dia mais comum. Nesse cenário, as tecnologias computacionais, desde que aplicadas de forma planejada e com a utilização dos recursos adequados, podem ser vistas como um valioso recurso adicional à disposição das pessoas que realizam tarefas em grupo, já que tornam possível um efetivo registro de informações e seu posterior acesso pelo grupo.

A demanda crescente de comunicação entre organizações dispersas combinada com os avanços da tecnologia em redes despertou o desenvolvimento de um grande número de aplicações CSCW. Essas aplicações têm como objetivo a disponibilização de formas apropriadas de cooperação dentro de um grupo de usuários para executarem atividades colaborativas.

Assim, ambientes colaborativos atuais consistem de várias aplicações, trabalhando em conjunto, mas de forma independente, promovendo contribuições das vantagens de cada uma delas. Dessa forma, ao permitir a integração dessas aplicações, poderíamos incentivar benefícios significativos aos usuários. Um ambiente de colaboração integrado permitiria que as diferentes funcionalidades das aplicações existentes fossem combinadas e controladas, aumentando consideravelmente as possibilidades de uso e funcionalidades.

Atividades colaborativas geralmente envolvem muitas pessoas com diferentes objetivos, tarefas, necessidades e permissões. Por isso, elas dificilmente são suportadas por uma única tecnologia. As tarefas colaborativas são complexas e novas funcionalidades são disponibilizadas ao usuário, quando várias aplicações trabalham em conjunto, ao invés de

um única aplicação complexa que resolve tudo sozinha. Assim, alguns sistemas CSCW tentam combinar diferentes funcionalidades para suportar colaboração em um simples ambiente.

Ultimamente, observam-se grandes esforços na integração de aplicações e componentes de *software* [Flores et al., 2005]. Com o objetivo de agregar funcionalidades e melhorar os potenciais usos dos sistemas de *software*, aplicações integradas possuem, em geral, mais funcionalidades e maiores possibilidades de uso do que aplicações isoladas.

Há algum tempo, o foco das pesquisas de muitos sistemas CSCW era o suporte para computação *web*, ou seja, suportar o acesso através dos *browsers*. Atualmente, esse enfoque tem se voltado para a área de serviços de *software*, não importando a localização desses serviços ou quais aplicações são usadas para manipulá-los.

Sistemas CSCW têm potencial para oferecer e consumir esses serviços nos mais variados cenários, já que uma característica comum nos mais diversos sistemas CSCW refere-se ao controle e gerenciamento de usuários, grupos de usuários, recursos e permissões.

Web services disponibilizam os padrões, protocolos e tecnologias requeridos para atingir o objetivo de prover serviços, permitindo que aplicações se comuniquem perfeitamente usando a infra-estrutura da Internet, mesmo que essas aplicações tenham sido construídas em diferentes plataformas de *hardware* e *software*. *Web services* permitem o suporte a aplicações CSCW para possibilitar a interoperabilidade entre várias aplicações e sistemas CSCW.

Considerando-se as características das aplicações CSCW, foram constatadas algumas necessidades e particularidades que são comuns na maioria dessas aplicações, como a necessidade do gerenciamento dos usuários e grupos de usuários da aplicação, gerenciamento dos recursos a que estes usuários têm acesso, bem como das permissões sobre esses recursos, ou seja, o que cada usuário pode ou não fazer no contexto dos recursos envolvidos no processo colaborativo. Além disso, quando várias aplicações CSCW são integradas, é possível que essas informações fiquem redundantes ou mesmo inconsistentes, já que cada aplicação deve armazenar internamente e com sua própria estrutura de dados essas informações.

Assim, foi desenvolvido o WS4CSCW (*Web Services for Computer Supported Cooperative Work*), um *Web Service* que tem como objetivo facilitar a integração de ferramentas e sistemas CSCW. O WS4CSCW permite o gerenciamento de usuários, grupos, recursos,

permissões e papéis de usuários e grupos, disponibilizando funcionalidades para aplicações voltadas ao trabalho em grupo, bem como para facilitar a integração entre essas aplicações, permitindo um 'co-funcionamento' mais direto, consistente e coeso.

Esta dissertação está organizada da seguinte forma: no Capítulo 2 são apresentados os conceitos gerais de CSCW, discutindo as principais características das aplicações CSCW. No Capítulo 3 são discutidos os conceitos de *Web Services* e seus mecanismos e protocolos para comunicação, descrição e execução dos serviços. No Capítulo 4 é apresentado o WS4CSCW, sua modelagem, serviços disponibilizados, estudos de caso realizados e trabalhos relacionados. No Capítulo 5 são discutidas as contribuições do trabalho e sugestões de trabalhos futuros.

2.1 Considerações Iniciais

Embora nem sempre perceptível, o desenvolvimento de trabalhos em grupo tornou-se uma tarefa cada vez mais comum nos dias atuais. Todo trabalho em que mais de uma pessoa participa, seja por meio de opiniões, tomada de decisões ou apenas atendendo a ordens, pode ser considerado um trabalho em grupo. No entanto, um trabalho em grupo só pode ser considerado um trabalho cooperativo se a relação entre os membros do grupo não for vertical, onde alguns decidem e outros executam o que foi estabelecido [Porto et al., 1994].

No contexto do trabalho cooperativo, a relação entre os membros do grupo deve ser horizontal, isto é, as decisões são tomadas e os problemas são resolvidos pelo grupo, depois de uma discussão entre os seus membros. Isso não significa que em um ambiente de trabalho cooperativo também não possa existir uma hierarquia, mas ela deve ser flexível [Porto et al., 1994].

Segundo Schmidt [Schmidt, 1999], são três as formas de cooperação humana:

1. Cooperação aumentativa: caracteriza-se por agrupar pessoas que combinam suas habilidades pessoais com o objetivo de superar as capacidades físicas e intelectuais, de um único indivíduo;
2. Cooperação de integração: processo por meio do qual as pessoas focalizam suas habilidades específicas no esforço para alcançar um resultado que nenhum deles poderia alcançar isoladamente;
3. Cooperação por debate: processo aplicado a um grupo com o objetivo de chegar a decisões por ponderação de opiniões pessoais de cada membro ou por definição de metas específicas em função de metas gerais.

CSCW (*Computer Supported Cooperative Work*) é definida como a área que estuda e aperfeiçoa conceitos e fundamentos relacionados com trabalho cooperativo apoiado por computador. O termo *groupware* refere-se à tecnologia, gerada pela pesquisa em CSCW, que dá suporte às atividades realizadas por pessoas organizadas em grupos que podem variar em composição, tamanho e local de trabalho.

Este capítulo tem como objetivo descrever os conceitos relacionados a CSCW e a sistemas de *groupware*, de forma a definir as características e funcionalidades necessárias para auxiliar aplicações CSCW, bem como facilitar a integração de tais aplicações. Esse estudo serviu de base para o projeto e implementação do *Web Service* desenvolvido, conforme descrito no Capítulo 4. Este capítulo está organizado da seguinte maneira: na Seção 2.2 são descritas as características do trabalho cooperativo. Na Seção 2.3 é apresentado um histórico do surgimento e evolução dos sistemas CSCW. Na Seção 2.4 são discutidas as características, definições, classificações e interfaces dos *groupwares*. E, finalmente, na Seção 2.5 são caracterizados os requisitos e a arquitetura de aplicações CSCW distribuídas.

2.2 Trabalho Cooperativo

Um dos principais objetivos do trabalho cooperativo refere-se à forma de solucionar problemas através da divisão destes em partes que possam ser resolvidas por grupos, com o intuito de diminuir o tempo e os custos envolvidos. Ou seja, um problema pode ser mais fácil e rapidamente resolvido desmembrando-o em problemas menores e, então, alocando essas partes a diversos grupos. Uma analogia a esta estratégia pode ser feita considerando-se um dos conceitos da computação paralela, o qual propõe a utilização de um algoritmo que “quebra” um problema grande em várias partes, que podem então ser concorrentemente executadas em diversos processadores [K. and Briggs, 1984] ou a abordagem de Engenharia de Software, de “dividir-para-conquistar”, também conhecida por modularização [Mayrhauser, 2000].

A utilização de ferramentas computacionais é fundamental para que haja uma efetiva troca de idéias entre os membros dos grupos de trabalho cooperativo, pois visa prover um aumento na produtividade. Neste caso, o computador torna-se uma forma prática e indispensável de aumentar o desempenho dos indivíduos envolvidos nas tarefas alocadas

[Schmidt, 1999].

Os mecanismos de controle de um sistema CSCW devem levar em consideração o papel de cada membro do grupo de modo a permitir o estabelecimento de diferentes formas de acesso. Desta forma, a atribuição de papéis aos indivíduos é uma importante característica dos sistemas de suporte ao trabalho em grupo. A seguir, estão listados alguns requisitos para sistemas CSCW [Brooke, 1993].

- Facilitar a colaboração entre os indivíduos, ao invés de impor práticas que causem mudanças radicais na forma de trabalho;
- Permitir a recuperação de informações armazenadas em Base de Dados;
- Permitir acesso aos dados, independente da localização dos usuários;
- Gerenciar o controle de acesso quando vários participantes tentam modificar os mesmos dados ao mesmo tempo;
- Reconhecer que mudanças são freqüentes neste contexto, e que, devido a isso, devem ser capazes de permitir a redefinição de procedimentos e processos, além de disseminar essas mudanças entre os usuários;
- Privilegiar a construção de aplicações menores e inter-relacionadas, ao invés de monolíticas;
- Garantir que as informações usadas no trabalho cooperativo sejam disseminadas entre a equipe.

Muitos fatores têm colaborado para o notável crescimento do trabalho cooperativo em grupo, dentre os quais destacam-se [Schmidt, 1999]:

- Os problemas tornam-se cada vez mais complexos e há necessidade de troca de idéias e de informações para o alcance de soluções;
- A evidência de maior dimensão destes problemas;
- A necessidade de maior especialização dos profissionais envolvidos;
- As soluções que requerem diferentes competências.

A combinação desses fatores resultou em uma mudança de atitude dos profissionais atuais, tornando ainda mais necessários os benefícios fornecidos pelo trabalho cooperativo em grupo, como troca de idéias e informações com profissionais mais experientes e em outras áreas [Gouveia, 2002].

No entanto, o trabalho cooperativo deve ser feito de forma planejada, principalmente quando envolve tarefas complexas. O planejamento visa garantir que o objetivo final seja alcançado a partir de tarefas individuais, gerenciando a natureza distribuída do trabalho. Em CSCW, o planejamento é realizado pelo trabalho de articulação, evitando que os participantes se envolvam em tarefas conflitantes ou repetitivas [Carstensen et al., 1999].

Ao ambiente computacional que implementa os processos de apoio à cooperação e, assim, possibilita o trabalho, a produção em conjunto e a troca de informações, denomina-se *groupware* ou CSCW [Borges et al., 1995]. Segundo [Fluckiger, 1995], CSCW concentra o desenvolvimento de sistemas baseados em computador para dar suporte e melhorar o trabalho de grupos de usuários com interesses, objetivos ou atividades comuns, bem como o entendimento dos efeitos do uso desse tipo de sistema.

Outro tópico também incluso nos conceitos de CSCW refere-se à variedade de formas de se trabalhar em grupo e, dentro desta variedade, dois conjuntos de usuários foram identificados [Hawryszkiewicz, 1998]:

1. Grupos de usuários com, no máximo, dez pessoas trabalhando no desenvolvimento de algum artefato. Este processo é feito com o auxílio de um aplicativo síncrono (todos utilizam o aplicativo no mesmo instante);
2. Grandes grupos constituídos por dezenas de pessoas de diferentes comunidades e com diferentes objetivos, cujo trabalho deve ser coordenado em direção às metas de uma empresa. Este trabalho é auxiliado por um aplicativo de uso assíncrono (não é necessário o uso do sistema simultaneamente por todos os membros).

Quando membros trabalham em grupo, existe a possibilidade de se produzir melhores resultados do que se os membros atuassem individualmente. Isso se deve a um eventual complemento de capacidades, de conhecimentos e de esforços individuais, e a interação entre pessoas com entendimentos, pontos de vista e habilidades diferentes [Fucks et al., 2002].

Um dos grandes desafios do CSCW é habilitar pessoas a trabalharem efetivamente em grupos que são mediados por computadores, considerando que este trabalho pode

ser dificultado pela falta de intimidade no uso de computadores [Mills, 1999]. Projetos na área de CSCW podem ser caracterizados por três grandes desafios a serem vencidos: comunicação, coordenação e colaboração [Chang et al., 2001]. Os aplicativos baseados em CSCW devem considerar esses desafios para prover as funcionalidades necessárias ao trabalho cooperativo.

No entanto, apesar das vantagens descritas, trabalhar de forma cooperativa demanda um esforço adicional para a coordenação de seus membros. Sem coordenação, boa parte dos esforços de comunicação não será aproveitada na cooperação. Para que o grupo possa trabalhar em conjunto de forma satisfatória e síncrona, é necessário que os compromissos assumidos nas conversações entre os participantes sejam realizados durante a cooperação. A coordenação deve evitar conflitos inter-pessoais que possam, eventualmente, prejudicar o grupo. Esta coordenação organiza o grupo a fim de evitar que esforços de comunicação e cooperação sejam perdidos e que as tarefas e processos sejam realizados na ordem e tempo corretos e cumprindo os objetivos e restrições previamente definidos [Raposo et al., 2001].

2.3 Histórico

Em meados dos anos 70, a crescente preocupação com o crescimento da produtividade nas organizações, onde a maior parte do trabalho era feita em grupo, deu origem a uma área de pesquisa chamada Automação de Escritório (OA - *Office Automation*) [Nitzke et al., 1999].

Os primeiros esforços nessa área buscavam integrar e transformar aplicações mono-usuário, como processadores de texto e planilhas eletrônicas, com o objetivo de permitir o acesso simultâneo de um grupo de usuários. Com isso, atenção maior foi dada ao desenvolvimento de aplicações de uso cooperativo. A evolução das redes de computadores, oferecendo atraentes possibilidades de comunicação e interação entre as máquinas e, conseqüentemente, entre as pessoas, tem gradativamente redirecionado a visão do uso dos computadores em geral [Moreira, 1998]. Nos últimos tempos, houve um aumento significativo do uso de computadores para promover o trabalho cooperativo entre as pessoas.

A sigla CSCW foi publicamente lançada em 1986, como título de uma conferência organizada pela *Association Computing Machinery* (ACM). A maioria dos trabalhos científicos na área de CSCW surgiu a partir desta primeira conferência. A terceira edição, em 1989,

foi a primeira conferência européia sobre o tema. Outras conferências de periodicidade irregular, além de diversas conferências sobre outros temas, têm dedicado crescente espaço à área de CSCW [Nitzke et al., 1999].

Uma vez que o suporte à comunicação está cada vez mais consolidado através do uso de redes, a maioria das atenções foi dirigida para o desenvolvimento de aplicações que possibilitem o trabalho e a produção em conjunto, e que acrescentem esforços às pesquisas na área de CSCW [Moreira, 1998].

Com a disseminação e popularização das aplicações voltadas ao trabalho em grupo, uma questão importante refere-se à privacidade das informações. Questões relacionadas a segurança e privacidade também têm sido foco das pesquisas envolvendo aplicações CSCW. É importante disponibilizar e compartilhar recursos entre os membros de determinado grupo, mas também é imprescindível restringir e esconder o que precisa ser privado [Patil and Kobsa, 2005], garantindo que somente os membros de um grupo possuam acesso a informações referentes àquele grupo e que, mesmo dentro de um grupo, as informações e acessos também possam ser restringidos. Pesquisadores têm relatado que a computação ubíqua é uma importante base para a criação de ambientes CSCW futuros [Chang et al., 2001], permitindo que os sistemas CSCW sejam beneficiados com uma tecnologia integrada ao ambiente de forma transparente, sem sobrecarregar a interface usuário-computador.

2.4 Groupware

Essa seção apresenta os conceitos de *groupware* e suas características. Denomina-se *groupware* o ambiente computacional que implementa os processos de apoio à cooperação entre pessoas, e, assim, possibilita o trabalho, a produção em conjunto e a troca de informações, ou seja, uma ferramenta que dá suporte ao trabalho realizado por um grupo de pessoas e fornece benefícios como resultado desta cooperação.

O trabalho em grupo permite a interação entre diferentes pessoas de diferentes organizações obtendo assim os seguintes resultados: eliminação das distâncias, aproximação das pessoas, redução do tempo gasto, aumento da produtividade e diminuição de custos.

2.4.1 Definição

O termo *groupware* geralmente é usado quase como sinônimo de CSCW, porém alguns autores identificam uma tendência diferenciada no emprego destes termos. Enquanto CSCW é usado para designar a pesquisa na área do trabalho em grupo e como os computadores podem apoiá-lo, *groupware* tem sido usado para designar o uso apropriado e integrado de um conjunto de ferramentas e de tecnologias, de informação e comunicações (*hardware* e *software*), de modo a desenvolver o trabalho em grupo dentro e fora das organizações e universidades [Passos and Bernardo, 1998].

Portanto, sistemas de correio eletrônico, suporte a decisão, teleconferências, suporte a reuniões e editores de texto cooperativos são exemplos de *groupware*, já que promovem a comunicação entre os membros de um grupo de trabalho, e que contribuem com isso para que o resultado seja maior que a soma das contribuições individuais de cada membro do grupo [Moreira, 1998].

Um crescente esforço tem sido notado no desenvolvimento e utilização de sistemas de *groupware*. Dentre as razões que justificam tal crescimento, destacam-se [UsabilityFirst, 2002]:

- Tornar o trabalho em grupo mais eficiente em tempo e custo;
- Facilitar a comunicação, tendo-a mais rápida, clara e persuasiva;
- Favorecer a comunicação em lugares onde não seria possível de outra forma;
- Favorecer o trabalho à distância;
- Reduzir os custos com viagens;
- Reunir múltiplos pontos de vista e opiniões;
- Formar grupos de interesses comuns em lugares onde não seria possível reunir um número suficiente de pessoas “face a face”;
- Facilitar a solução de problemas de grupo;
- Favorecer novas formas de comunicação, como por exemplo, intercâmbios anônimos ou interações.

2.4.2 Informações Distribuídas

Uma questão importante na criação de *groupwares* refere-se ao compartilhamento de informações. À medida que informações são manipuladas por grupos distribuídos, a distribuição e o compartilhamento de informações devem ser considerados e tratados. Deve-se garantir, ainda, que essas informações sejam fornecidas aos usuários de forma rápida e confiável.

Para tratar esse gerenciamento de informações distribuídas, há três abordagens distintas: gerenciamento centralizado, gerenciamento distribuído e gerenciamento híbrido, conforme descritos a seguir.

Gerenciamento Centralizado: No gerenciamento centralizado, como exibido na Figura 2.1, apenas o servidor central controla as informações e o trabalho dos usuários do *groupware*. A vantagem resultante deste tipo de gerenciamento é que o sincronismo das informações entre os diversos usuários é facilitado, já que essas informações estão localizadas em um único lugar. No entanto, esse gerenciamento torna o sistema vulnerável a falhas e lentidão do servidor central, provavelmente sobrecarregado em caso de muitos usuários concomitantemente, devido ao grande número de requisições.

Figura 2.1: Gerenciamento centralizado

Gerenciamento Distribuído: O gerenciamento distribuído, exibido na Figura 2.2, mantém uma cópia do *groupware* na máquina de cada usuário. Os *groupwares* de cada usuário são sincronizados através da comunicação direta entre si. Nesse gerenciamento, cada estação local é responsável pelo seu usuário e pela troca de todas as

informações necessárias com as outras estações distribuídas. Deste modo, a possibilidade de falha ou sobrecarga do servidor central é eliminada. Entretanto, pode haver dificuldades na coordenação, sincronismo e consistência das informações. Pesquisas relatam esforços em se prover gerenciamento distribuído. Em [Ma et al., 2003] são relatadas as características e dificuldades encontradas no desenvolvimento de um sistema *groupware* P2P ¹ com arquitetura distribuída de gerenciamento.

Figura 2.2: Gerenciamento distribuído

Gerenciamento Híbrido: O gerenciamento híbrido caracteriza-se por combinações alternativas dos dois modelos referidos anteriormente. Um gerenciamento híbrido típico é a utilização do servidor central apenas para efeito de sincronização, com todas as outras atividades sendo executadas localmente. Cada mudança significativa é reportada ao servidor central, e então redistribuída aos outros usuários. Esse tipo de arquitetura é ilustrado na Figura 2.3.

2.4.3 Características

A seguir são descritas algumas características de sistemas *groupware* [Passos and Bernardo, 1998].

- **Comunicação:** a comunicação envolve pessoas que se relacionam, discutem e desenvolvem novas idéias. Esta comunicação auxilia as pessoas a compartilharem suas informações pessoais. Facilidades de comunicação são essenciais para um sistema cooperativo, uma vez que o trabalho em grupo segue uma sequência de conversação;

¹*Peer-to-peer*: rede virtual, onde cada estação possui capacidades e responsabilidades equivalentes.

Figura 2.3: Gerenciamento híbrido

- **Coordenação:** envolve equipes de projeto, planejamento e agendamento de compromissos de grupos. A característica de coordenação ajuda as pessoas a conhecerem os seus papéis dentro dos grupos e faz com que haja uma interação com os papéis dos demais membros. A coordenação auxilia o trabalho cooperativo na gerência e no acompanhamento do trabalho do grupo e das atividades de cada componente de forma individual. A coordenação também é responsável por gerenciar as permissões que cada grupo e membros dos grupos têm para cada recurso, permitindo e restringindo acessos;
- **Memória Corporativa:** é um conceito muito importante numa economia baseada em conhecimento. Isso deve-se ao fato de que é imprescindível conhecer o que foi feito anteriormente para que se possa seguir em frente. Memória corporativa refere-se a todos os documentos, registros e conhecimentos elaborados ou referenciados durante o processo de desenvolvimento de uma atividade ou projeto em grupo;
- **Cooperação:** envolve atividades eletrônicas dos membros do grupo na Memória Corporativa. Os membros de um grupo cooperam para produzir, manipular, organizar e refinar informações, como documentos, planilhas, textos, gráficos, etc. A cooperação ajuda e estimula as pessoas a trabalharem juntas sobre um mesmo objetivo;

Além dessas características descritas, [Dourish and Belloti, 1992] definem também a percepção, ou contexto, como característica importante de sistemas de *groupware*:

- Contexto: informação relevante sobre as entidades que participam da interação entre usuários e entre estes e o computador, incluindo os próprios usuários e o computador. No trabalho em grupo, refere-se ao aspecto da percepção de um indivíduo sobre o trabalho do restante do grupo. Quando um elemento do grupo está trabalhando em sua estação, como e quando tornar visível para ele as atividades que estão sendo realizadas pelos outros, principalmente aquelas que interferem com a sua? Esta percepção do sistema disponibiliza um contexto, que pode afetar as atividades individuais de cada membro do grupo e do grupo como um todo. Os elementos de percepção transmitem as mudanças de planos, contribuindo para gerar o novo entendimento. Além disto, podem informar aos participantes os aspectos temporais e espaciais de suas ações e facilitar a sincronização das tarefas individuais [Gerosa et al., 2003]

Figura 2.4: Funcionamento da Colaboração, refinamento de [Fuks et al., 2002]

O contexto refere-se à capacidade de tornar os membros do grupo e suas atividades visíveis para os outros membros do grupo e melhorar o entendimento das atividades do grupo e da interação dos participantes. Com isso, é possível disponibilizar uma cooperação mais eficiente. Informações referentes ao contexto referem-se a presença (Há alguém no sistema?), identidade (Quem é determinado indivíduo?), localização (Onde está determinado indivíduo?), ação (O que um determinado indivíduo está fazendo?), etc [Nova et al., 2003].

Em sistemas distribuídos, tradicionalmente, diferentes formas de transparência têm sido utilizadas para esconder informações dos usuários. Entretanto, na área CSCW, há a necessidade de certas informações para uma cooperação efetiva. [Ramduny and Dix, 1997] definem quatro tipos de informações relativas ao contexto de um grupo no trabalho cooperativo:

1. Ciência da presença dos membros do grupo e as respectivas disponibilidades para o trabalho em grupo;
2. Ciência dos resultados das ações do grupo, isto é, que mudanças ocorreram;
3. Ciência de como as mudanças ocorreram;
4. Ciência dos estados dos canais de comunicação.

No diagrama da Figura 2.4 são exibidas as características descritas e o relacionamento entre elas. O diagrama é um refinamento de [Fuks et al., 2002], incluindo também a Memória Corporativa, que é gerenciada pela Coordenação e disponibiliza elementos para a Cooperação, permitindo que os usuários trabalhem cooperativamente, sem que um iniba ou prejudique o trabalho do outro, e que os elementos comuns sejam gerenciados pela Coordenação, controlando os papéis e permissões de acesso de cada usuário.

A Comunicação é caracterizada por permitir que os membros do grupo procurem um entendimento comum, compartilhem idéias, discutam pontos conflitantes e tomem decisões, desta forma auxiliando o processo de cooperação. A Comunicação também pode gerar compromissos, como agendamento de futuras reuniões ou *deadlines* de projetos. Esses compromissos devem ser gerenciados pela Coordenação, conforme mostra a Figura 2.4.

A Cooperação caracteriza-se pelas atividades conjuntas dos membros do grupo em recursos da Memória Corporativa. Quando os membros do grupo trabalham em recursos

compartilhados (armazenados e recuperados da Memória Corporativa), esses recursos devem ser gerenciados de modo a se evitar inconsistências e redundâncias. Essa tarefa fica a cargo da coordenação, que pode disponibilizar ferramentas de gerenciamento de artefatos, como registro e controle de versões, controle e permissões de acesso, etc. A Cooperação também gera elementos para o contexto do sistema, como, por exemplo, que usuários estão trabalhando em conjunto naquele momento e o que está sendo produzido.

As informações resultantes dos processos cooperativos podem ser importantes e deve ser possível acessá-las posteriormente. Uma boa estratégia é preservá-las de forma estruturada, catalogada e categorizada, de forma a facilitar a busca em momentos nos quais essas informações sejam necessárias. Este tipo de conhecimento pode ser definido como conhecimento formal. No entanto, o conhecimento informal, ou seja, idéias, conversas, pontos de vista, fatos, questões, discussões e decisões que ocorrem durante o processo cooperativo e acabam por defini-lo, é difícil de ser capturado. Pesquisas têm sido dedicadas ao armazenamento destes conhecimentos [Dey and Abowd, 2000, Dey, 2001]. Esse conhecimento informal permite recuperar o histórico de uma discussão e o contexto em que as decisões foram tomadas, por exemplo.

Para garantir o cumprimento de compromissos em grupo e a realização do trabalho cooperativo através da união dos trabalhos individuais, é necessária a Coordenação das atividades. Esta Coordenação organiza o grupo e evita que esforços de Comunicação e Cooperação sejam perdidos e que as tarefas sejam realizadas na ordem e no tempo corretos e cumprindo as restrições e objetivos definidos [Raposo et al., 2001].

Eventualmente, podem ocorrer conflitos devido a problemas de comunicação ou de percepção, ou por diferenças na interpretação da situação ou de interesse [Putnam and Poole, 1987]. A Coordenação deve tratar os conflitos que prejudiquem o grupo, como competição, desorientação, problemas de hierarquia, difusão de responsabilidade. A Coordenação também deve tratar as permissões envolvidas no sistema, como que usuários e grupos têm acesso a quais recursos.

Para permitir a cooperação, são necessárias algumas informações sobre o que está acontecendo a cada momento. Estas informações são fornecidas através de elementos de percepção que capturam e armazenam as informações coletadas durante a interação entre os participantes e o funcionamento do sistema. A percepção em si é relativa ao ser humano, enquanto os elementos de percepção estão relacionados à interface do ambiente

[Fuks et al., 2002].

[Dey and Abowd, 2000] definem contexto como qualquer informação relevante sobre as entidades que participam da interação entre usuário e computador, incluindo os próprios usuário e computador. Com o uso de informações de contexto, as aplicações podem disponibilizar benefícios aos grupos e usuários na realização de suas tarefas. Por exemplo, estando ciente do horário atual, aplicações como agendas ou calendários poderiam enviar avisos a seus usuários ou a todos os membros de determinado grupo; estando ciente da identificação de usuários (a qual grupo o usuário pertence), aplicações poderiam personalizar sua aparência ou interface, disponibilizando apenas as funcionalidades a que o grupo tem permissão. Ainda, diferentes tipos de informações de contexto podem ser utilizados em conjunto. Por exemplo, aplicações cientes de tempo e de identidade podem expandir seus serviços de histórico e de busca de dados. Com o objetivo de disponibilizar armazenamento e recuperação de informações de contexto, o *Context Kernel* [Arruda, 2002] é um *Web Service* que permite às aplicações gerenciarem suas informações de contexto.

Um *groupware* pode ser composto por um conjunto de ferramentas colaborativas, possibilitando a interação entre múltiplos usuários. Como os processos de trabalho entre os indivíduos são muito específicos e evolutivos, a tecnologia de *groupware* deve prover flexibilidade suficiente para ser adaptada às necessidades de cada grupo e à evolução dos processos de trabalho. O *groupware* deve prover ao grupo a possibilidade de montar seu contexto de trabalho, selecionar e configurar um conjunto de ferramentas colaborativas específicas para suas necessidades [Fuks et al., 2002].

2.4.4 Tipos de Groupware

Nesta seção são descritos alguns tipos de *groupware* com o intuito de ilustração da ampla variedade de possibilidades de sistemas que suportam e apóiam o trabalho cooperativo [Brink, 1998].

Workflow

Os sistemas *workflow* permitem que documentos sejam distribuídos através das organizações. Um exemplo simples de uma aplicação *workflow* é um relatório de despesas em uma organização: um empregado escreve um relatório de despesas e o envia a seu chefe. Uma cópia do relatório é arquivada e então é enviada ao gerente para aprovação. O gerente

recebe o original, aprova-o eletronicamente, a despesa é registrada no grupo e enviada ao departamento de contabilidade para o pagamento. Os sistemas de *workflow* podem fornecer características como distribuição, desenvolvimento de formulários e base para diferentes papéis e privilégios.

Edição Colaborativa

Edição colaborativa proporciona a possibilidade de edição de documentos por mais de uma pessoa. Os sistemas cooperativos de escrita suportam trabalhos que acontecem ao mesmo momento e trabalhos que acontecem em momentos diferentes. Podem ser usados por um grupo para compor e editar documentos, gráficos ou textos conjuntamente. Geralmente, há uma área de trabalho comum onde todos atuam e podem visualizar o que é feito. Há editores síncronos, que possibilitam a um usuário editar uma frase enquanto outro usuário está atualizando a frase seguinte, sendo possível que ambos visualizem, em tempo real, o que cada um está fazendo. Existem também editores assíncronos, para que os usuários modifiquem o documento de forma individual, e posteriormente visualizem as alterações dos outros membros do grupo.

Lousas Compartilhadas

As lousas compartilhadas permitem que duas ou mais pessoas visualizem e desenhem em uma mesma tela mesmo estando em locais diferentes. A maioria das lousas compartilhadas é projetada para a conversação informal, mas podem também servir para comunicações estruturadas mais sofisticadas, como projeto gráfico cooperativo, publicações ou aplicações de engenharia.

Sistemas de Comunicação Síncrona

Esses sistemas caracterizam-se pela necessidade da presença temporal dos usuários, independente da localização geográfica dos mesmos. Alguns exemplos de ferramentas de comunicação síncrona são: sistemas de bate-papo; sistemas de mensagens instantâneas; e sistemas de videoconferência.

- Bate-papo (*chat*): Permitem a muitas pessoas escreverem mensagens em tempo real em um espaço público, permitindo a interação em tempo real através da troca de mensagens em forma de textos e gráficos. Alguns sistemas implementam a troca de

símbolos que representam expressões faciais (*emoticons* ou *smileys*), com o objetivo de tornar a comunicação mais expressiva.

- Sistemas de mensagens instantâneas: geralmente permitem a comunicação direta e reservada entre dois ou mais usuários. Empresas possuem sistemas de mensagens instantâneas com o objetivo de alavancar suas estratégias de gestão do conhecimento através da melhoria da comunicação interna. Alguns exemplos desse tipo de *groupware* são: MSN, ICQ, IRC e AIM
- Videoconferência: A videoconferência é essencialmente um sistema de telefone com um componente visual adicional. O vídeo é vantajoso quando a informação visual está sendo discutida, mas não pode fornecer o benefício substancial na maioria de casos onde os telefones convencionais são adequados. Vídeo suporta conversação entre os membros de um *groupware* e oferece uma visão remota das atividades que estão sendo desenvolvidas pelos membros do grupo.

Sistemas de Comunicação Assíncrona

São sistemas nos quais não é necessária a presença dos usuários ao mesmo tempo: a utilização pode ser feita nos horários que houver conveniência. Dentre os sistemas de comunicação assíncrona, destacam-se: *e-mail*, listas de discussão, fóruns e *blogs*.

- *E-mail*: É a aplicação mais comum dos tipos de *groupware*. Através dele, as pessoas podem trocar idéias e informações e, com isso, conseguem um crescimento conjunto nas áreas abordadas. Sistemas básicos de *e-mail* incluem tipicamente características interessantes: repassar mensagens, arquivar mensagens, criar grupos de usuários e anexar arquivos.
- *Newsgroups* e Listas de Usuários: Os *newsgroups* e as listas são similares aos sistemas do *e-mail*, exceto pelo fato de utilizarem mensagens entre grupos de usuários. As listas são muito usadas para troca de mensagens entre pequenos grupos, de forma assíncrona. As mensagens são enviadas por um participante ao endereço do servidor, que realiza a distribuição da mensagem para todos os usuários cadastrados naquela lista. Podem ser criados grupos fechados, abertos ou semi-abertos de participantes. Também é possível a existência de um moderador para filtragem ou avaliação da pertinência das mensagens enviadas.

- Fóruns: É um recurso bastante utilizado por comunidades de usuários com temas e interesses em comum. Geralmente os usuários podem acessar uma área, ler e publicar mensagens e fazer comentários sobre as demais. As mensagens são organizadas em tópicos, com o objetivo de estruturar melhor o conteúdo publicado.
- *Blogs*: são páginas *web* atualizadas freqüentemente, composta por pequenos parágrafos dispostos de forma cronológica. Os temas e conteúdos são diversificados, como diários, piadas, *links*, notícias, fotografias e aplicações de ensino a distância.

Agendas para Grupos

As agendas para grupos permitem marcar compromissos, gerenciar projetos e coordenar ações e compromissos entre muitas pessoas através da visualização do seu calendário de atividades pessoais. Uma característica importante é encontrar horários livres comuns entre os membros do grupo, para que não haja conflitos entre compromissos. A ferramenta *No Risk Planning*, utilizada como estudo de caso do WS4CSCW, descrito na Seção 4.7.2, é um exemplo de agenda para grupos. Em [Fortes et al., 2004a] é realizado um estudo do processo de engenharia e um estudo comparativo entre várias agendas *web* para grupos.

Área de trabalho compartilhada (*Shared Workspace*)

Fornece funcionalidades básicas para cooperação de grupos e permite que usuários armazenem vários tipos de arquivos, com o objetivo de organizar tarefas ou atividades e ter acesso às ações dos membros do seus grupo. Inclui funcionalidades para armazenamento de documentos, gerenciamento de versões, administração de membros e grupos, e edição colaborativa.

- Quickplace: é uma área de trabalho para colaboração em equipe na *web*. Tem como objetivo auxiliar os usuários a planejar reuniões, gerenciar tarefas, comunicar-se em tempo real, apresentar idéias, criar e editar documentos [IBM Corporation, 2005]. Possui as seguintes funcionalidades: sistemas de busca e gerenciamento de conteúdo; integração com banco de dados; sistema de administração de equipe; compartilhamento de arquivos; co-autoria na criação de documentos; divulgação de documentos na *web*; e *chat*. Os usuários possuem diferentes níveis de acesso, podendo ser leitores, colaboradores ou gerentes.

- BSCW² (*Basic Support for Cooperative Work*): permite o trabalho cooperativo via rede. Tem como objetivo ser acessível a partir de *browsers* convencionais, sem a necessidade de instalação de ferramentas adicionais nos clientes. O BSCW fornece funcionalidades básicas para cooperação de grupos e utiliza a *web* como infra-estrutura de comunicação. Possui recursos para armazenamento de vários tipos de documentos, gerenciamento de versões, administração de membros e grupos e edição colaborativa de documentos, podendo conter vários tipos de objetos (documentos, imagens ou *links*) [Bentley et al., 1997]. Os objetos são organizados em hierarquias de pastas. Membros de um determinado grupo podem postar objetos no espaço compartilhado ou transferir objetos deste espaço para o seu sistema local [Klockner, 2000]. Em [Appelt, 2001], baseando-se no histórico de utilização do BSCW, são analisadas quais funcionalidades os usuários mais utilizam, dentre elas: autenticação, fóruns de discussão, buscas, conversão de formatos de documentos, controle de versões e suporte a vários idiomas.

Sistemas de Suporte à Decisão

Os sistemas de suporte a decisão são projetados para facilitar a tomada de decisões por grupos. Fornecem ferramentas para *brainstorming*, críticas a idéias, colocação de pesos e probabilidades sobre eventos e alternativas e suporte a votação. Tais sistemas permitem decisões mais racionais porque fazem com que os membros do grupo pensem mais sobre o assunto em questão. Têm como objetivo aumentar a produtividade, a velocidade do processo decisório e a qualidade do resultado de uma reunião. Para isso, utilizam mecanismos para votação, geração de idéias e identificação de alternativas, permitindo a captura do raciocínio e da argumentação gerados durante as reuniões.

Sistemas de apoio a Reuniões

Permitem a realização de trabalhos cooperativos de forma distribuída e, posteriormente, seu prosseguimento na sala de reuniões, no caso da necessidade de um encontro “face a face”. Ferramentas de suporte a reuniões geralmente apóiam o planejamento de reuniões e a criação de agendas. Elas fornecem os meios necessários para que o grupo chegue a um consenso, o que resulta em um maior comprometimento com as decisões tomadas. Outra

²Disponível online em <http://bscw.fit.fraunhofer.de/index.html>

funcionalidade deste tipo de *groupware* é a geração automática de relatórios das reuniões realizadas.

Sistemas para Gerenciamento Eletrônico de Documentos (GED)

Auxilia o processo da manipulação diária de documentos de uma organização, tornando mais ágil e fácil a sua recuperação e agilizando os processos operacionais e de tomada de decisão.

2.4.5 Interfaces em *groupwares*

Considerando-se o desenvolvimento de *groupwares*, uma importante questão refere-se ao projeto de interfaces usuário-computador. Essa interface deve usar metáforas claras, com a finalidade de proporcionar ao usuário um conjunto de expectativas associadas ao seu ambiente computacional, permitindo um aprendizado mais rápido e intuitivo do usuário.

Segundo [Shneidermann, 1998], são necessárias algumas diretivas no projeto de interfaces, entre elas:

- Manter a consistência: umas das regras mais freqüentemente violadas no projeto de interfaces, pode causar frustrações na utilização do sistema. Uma interface deve manter seqüências de ações semelhantes para tarefas semelhantes. Além disso, as terminologias utilizadas devem ser padronizadas em menus, botões e telas de ajuda. Da mesma forma, devem-se padronizar também cores e fontes. Mensagens devem ser exibidas de forma clara e compreensível;
- Prevenção de erros: interfaces não devem induzir o usuário ao erro. Caso ocorram erros, o tratamento deve ser o mais natural possível, sem penalizar o usuário com sinais sonoros ou com a necessidade de reiniciar um longo processo;
- Uso de atalhos: deve haver a possibilidade de automatizar tarefas que se tornam repetitivas, permitindo, por exemplo, uma longa seqüência de ações com comandos reduzidos;
- Agrupamento de tarefas semelhantes: com o objetivo de facilitar a localização das opções de diálogo, é aconselhável manter agrupadas funções com características ou alvos semelhantes;

- Reduzir a memorização: uma interface não deve exigir do usuário a memorização de uma longa lista de comandos. Por exemplo, deve-se escolher textos de menus que favoreçam o reconhecimento de suas funções no momento da sua visualização.
- Possibilidade de desfazer ações: ações devem ser reversíveis. Com essa característica, o usuário sente-se mais à vontade no uso da interface, concentrando-se na tarefa a ser realizada. Além disso, encoraja o usuário a explorar melhor a ferramenta e enriqueça o seu aprendizado acerca das opções desconhecidas do sistema.

Interfaces de sistemas de *groupware* são baseadas em sistemas de interfaces para um usuário, já que cada estação é utilizada por um único usuário. No entanto, a interface também deve considerar elementos específicos do trabalho cooperativo, como, por exemplo, a memória de grupo. Pesquisas nesta área apontam uma necessidade de mecanismos de interação que considerem a memória e a percepção do grupo [Borges and Pino, 2001] e [Greenberg, 1997].

2.5 Aplicações CSCW Distribuídas

O enorme crescimento notado nos últimos tempos das redes interconectadas e, mais especificamente, da Internet tem resultado no crescimento da infra-estrutura com grande potencial para apoiar o desenvolvimento e uso de aplicações distribuídas. Aplicações CSCW distribuídas podem usar esta infra-estrutura disponível para reunir e combinar funcionalidades, através da integração de aplicações, e prover sistemas *groupware* mais funcionais, abrangentes e flexíveis. Além disso, a integração também permite um reaproveitamento de funcionalidades, já que uma aplicação pode requisitar funções e tarefas de outra aplicação que já as faça, ao invés de implementá-las.

2.5.1 Requisitos de Aplicações CSCW Distribuídas

Para haver um bom funcionamento da integração e co-funcionamento de aplicações CSCW distribuídas, são definidos alguns requisitos básicos [Cosquer et al., 2000].

- Necessidade de gerência de dados compartilhados com controle de acesso flexível: a maioria das plataformas distribuídas (base de dados, sistemas de arquivos, etc) são construídas com o objetivo de tornar essa distribuição transparente ao usuário. Essa

transparência se opõe ao conceito natural de percepção das aplicações CSCW. Os sistemas de controle de acesso devem considerar e tratar esses conflitos de definições no gerenciamento de acessos. Alguns exemplos de mecanismos que têm se mostrado úteis para solucionar esses conflitos são citados a seguir.

- Papéis para estruturação de hierarquias;
 - Mecanismos de reserva;
 - Políticas de acesso.
-
- Necessidade de comunicação distribuída eficiente: melhora a resposta da ferramenta devido à característica distribuída do sistema;
 - Necessidade de gerenciamento de audiência: muito pouco esforço tem sido gasto no registro e representação dos relacionamentos dinâmicos dos usuários. Esse requisito deve tratar o registro e sincronização dos usuários: quem pode fazer o quê, como, etc. e manter o registro do estado dos usuários: quem está *online*, *offline*, etc;
 - Troca de mensagens entre aplicações e usuários;
 - Identificação das aplicações e recursos distribuídos;
 - Endereçamento de aplicações e recursos;
 - Mobilidade de aplicações.

2.5.2 Arquitetura de Aplicações CSCW Distribuídas

Aplicações distribuídas contam com uma variedade de serviços distribuídos e ambientes de desenvolvimento de aplicações conhecido como *middleware*, que é definido como a camada de *software* que permite a comunicação de diferentes componentes de aplicações distribuídas através de infra-estruturas de redes e sistemas operacionais [Gribble and Brewer, 1997, Emmerich, 2000].

Baseado na definição da arquitetura de *middleware*, [Cosquer et al., 2000] definiu a arquitetura da Figura 2.5 para aplicações CSCW distribuídas.

Na base da pilha das camadas da Figura 2.5 encontra-se a infra-estrutura distribuída, composta pelo sistema operacional e *software* de rede, que fornece os processos de trans-

Figura 2.5: Arquitetura de aplicações CSCW distribuídas [Cosquer et al., 2000]

ferência não confiável de dados. No topo da pilha estão situadas as aplicações CSCW. A camada intermediária é composta por duas sub-camadas:

Serviços *middleware* gerais (camada 2.1 da Figura 2.5): disponibilizam transferência confiável de dados, mobilidade de código, transações, etc., e auxilia os desenvolvedores a lidarem com problemas de falhas e concorrência.

Serviços *middleware* específicos (camada 2.2): implementam serviços específicos para suporte às aplicações, como suporte e gerenciamento de sessões.

A vantagem desta arquitetura, baseada em camadas, é separar mecanismos básicos (geralmente embutidos nos *middlewares* genéricos) das políticas específicas dos domínios das aplicações. As políticas específicas das aplicações CSCW são embutidas nas plataformas de suporte e contam com o auxílio da sub-camada de serviços *middleware* gerais.

2.6 Considerações Finais

Neste capítulo foram descritos os conceitos e características de CSCW, enfatizando a área de sistemas *groupware*. Também foram descritos os conceitos relacionados a aplicações CSCW distribuídas. Os conceitos principais e que recorrentemente são mencionados na

literatura são de que existe a necessidade de identificar níveis de comunicação, colaboração [Fuks et al., 2002], coordenação [Raposo et al., 2001], contexto [Rosa et al., 2003] e memória corporativa [Passos and Bernardo, 1998].

Para a integração de aplicações CSCW, neste trabalho investigamos o suporte que permita o gerenciamento de grupos e usuários, de forma a facilitar e organizar o trabalho cooperativo entre as aplicações envolvidas, e gerenciar os recursos da memória corporativa e as permissões atribuídas a cada usuário ou grupo, permitindo que o trabalho cooperativo tenha uma coordenação efetiva e consistente.

Web Services

3.1 Considerações Iniciais

Devido à crescente necessidade de comunicação e interoperabilidade de aplicações, foi notado um considerável crescimento no uso de *Web Services* na *World Wide Web*. O mundo está cheio de dados e serviços úteis. No entanto, a maioria destes dados e serviços estão isolados e não permitem acesso por outros dados e aplicações. *Web Services* permitem que computadores se comuniquem, tornando os dados e serviços acessíveis, seja através de uma rede interna ou da *Web*. *Web Services* permitem uma solução simples para o problema de heterogeneidade causada por diferenças entre tecnologias de rede, sistemas operacionais, dispositivos computacionais, linguagens de programação, mecanismos de comunicação e estruturação dos dados.

Web Services permitem que aplicações possam inter-operar por meio de padrões *Web* [Martin et al., 2003]. Projetados e construídos baseando-se em padrões abertos, *Web Services* possibilitam que funcionalidades de aplicações sejam exportadas para outras aplicações [Ray and Kulchenko, 2002].

Bibliotecas pré-compiladas são úteis na linguagem em que elas foram compiladas. A utilização de *Web Services* permite o reaproveitamento de código entre módulos de sistemas desenvolvidos em diferentes linguagens. Isso é feito baseando-se em requisições cliente-servidor, nas quais um servidor (*Web Service*) atende requisições de clientes para a funcionalidade a qual se propõe a disponibilizar, e retorna uma resposta ao cliente.

Apesar de existirem tecnologias que possibilitam o acesso remoto a serviços distribuídos, os *Web Services* apresentam algumas vantagens sobre tais tecnologias. De acordo com [Stal, 2002], tecnologias como CORBA [Object Management Group, 2003] e Java RMI [Microsystems, 2002], por exemplo, são complexas e precisam de um alto poder de comunicação, além de apresentarem baixa escalabilidade e baixa capacidade de interoperabilidade. *Web Services* disponibilizam interoperabilidade, simplicidade e flexibilidade com

um baixo custo de processamento, ao utilizar mecanismos como o protocolo HTTP (*Hypertext Transfer Protocol*) [Fielding et al., 1999] para transporte e a linguagem XML para representação e padronização de suas informações [Kleijnen and Raju, 2003]. Esta seção descreve as características e conceitos relacionados a *Web Services* e apresenta as principais tecnologias relacionadas.

3.2 Histórico

Assim como a maioria dos padrões referentes a sistemas computacionais e protocolos de comunicação, *Web Services* são resultado de uma longa série de evoluções e iniciativas em computação distribuída. Portanto, não é um padrão que foi disponibilizado e recomendado instantâneamente.

O primeiro passo pode ser situado no projeto dos sistemas RPC's (*Sun Remote Procedure Call*), que permitem a interação de um cliente e um servidor através do modelo Requisição-Processamento-Resposta. Uma dificuldade nesta abordagem é a representação binária dos dados, que variam dependendo da linguagem de programação, *hardware* e sistema operacional. A Sun Microsystems [Microsystems, 2004b] resolveu esse problema especificando como codificar valores em parâmetros - um formato binário padrão - e servidores passaram a representar valores em XML (*Extensible Markup Language*) [W3C, 2004a].

O COM (*Component Object Model*) representou um segundo passo para a evolução dos *Web Services* como são padronizados hoje. Uma iniciativa da Microsoft [Corporation, 2006], COM foi baseado em independência de linguagem, interoperabilidade e componentes reutilizáveis. Enquanto RPC's baseavam-se apenas em chamadas de procedimentos, COM baseia-se em objetos e chamadas de métodos [Ray and Kulchenko, 2002]. O modelo COM foi então estendido para DCOM (*Distributed Component Object Model*), melhorando algumas especificações de interfaces.

Paralelamente, surgiu a CORBA (*Common Object Request Broker Architecture*) [Object Management Group, 2003], que é baseada em objetos, possui em seu núcleo um ORB (*Object Request Broker*), que é responsável por receber as requisições e despachá-las para o servidor apropriado. Para descrever as funções que um serviço oferece é utilizada a IDL (*Interface Design Language*). Baseando-se na IDL, um código-fonte é construído

para manipular a interação com o ORB.

A insatisfação dos programadores com vários aspectos das tecnologias descritas anteriormente levaram-nos a idealizar a combinação dos melhores aspectos dessas tecnologias (independência de plataforma, alto nível de abstração, interoperabilidade entre diferentes linguagens) e deixar de lado as desvantagens (complexidade dos sistemas ORB, sistemas proprietários, IDLs confusas) [Ray and Kulchenko, 2002].

Isso motivou o desenvolvimento e padronização da tecnologia de *Web Services*. Por utilizarem XML para representação de dados, *Web Services* permitem a construção de aplicações que não são restritas a *hardwares* ou linguagens de programação específicas. Além disso, são baseados em padrões abertos e não possuem a complexidade dos ORBs, *sockets* e tratamento da negociação de conectividade, já que estão embutidos no topo do protocolo HTTP.

Nos dias atuais, os *Web Services* têm sido usados como ferramenta nos mais variados campos de aplicação da computação. Com o notável crescimento da comercialização e utilização de dispositivos móveis, arquiteturas para acesso aos *Web Services* têm sido propostas e implementadas considerando as atuais restrições dos dispositivos móveis, como memória e processamento. Aplicações têm sido propostas [Rendon et al., 2005] considerando todos os tipos de dispositivos móveis, dos mais restritos aos mais tecnologicamente evoluídos, usando *Java 2 Platform Micro Edition* [Microsystems, 2004a] e SMS (*Short Messaging Service*). Por exemplo, SMS permite que o usuário estabeleça uma comunicação bi-direcional para enviar e receber mensagens formatadas de um servidor genérico qualquer [Lai, 2004]. Isso demonstra a abrangência dos possíveis usos e soluções providos pelos *Web Services*.

3.3 Definição

Web Services são sistemas de *software* identificados por uma URI (*Uniform Resource Identifier*) e definidos por um documento de descrição baseado na linguagem XML. Essa descrição tem como objetivo possibilitar que outros sistemas possam identificar o sistema descrito e interagir com o mesmo segundo sua definição, utilizando mensagens baseadas em XML conduzidas por protocolos Internet.

Web Services fornecem uma solução simples para resolver o problema de heterogenei-

dade causada por diferenças entre tecnologias de rede, dispositivos, sistemas operacionais, linguagens de programação, mecanismos de comunicação e estruturação dos dados.

Um *Web Service* pode ser visto como um conceito abstrato que deve ser implementado por um agente concreto. Esse agente é a parte concreta do *software* ou *hardware* que envia e recebe mensagens. O serviço é definido como o recurso caracterizado pelo conjunto abstrato de funcionalidades que são providas [W3C, 2002].

O funcionamento dos *Web Services* é baseado em uma arquitetura composta de mecanismos capazes de enviar e receber mensagens, descrevê-los e publicar essas descrições. Essa arquitetura define a interação entre agentes de *software* através da troca de mensagens entre os que requisitam e os que fornecem serviços.

Os fornecedores de serviços têm a responsabilidade de publicar a descrição de um ou mais serviços. Já os agentes que requisitam serviços devem ser capazes de encontrar e identificar tais serviços.

3.4 Arquitetura

A arquitetura básica de funcionamento e construção dos *Web Services* baseia-se em mecanismos capazes de enviar e receber mensagens, descrever os *Web Services* e publicar essas descrições. Essa arquitetura representa e define a interação entre os agentes de *software*, como a troca de mensagens entre os que requisitam e os que fornecem serviços. Os agentes fornecedores de serviços devem publicar a descrição de um ou mais serviços e os agentes que requisitam os serviços devem ser capazes de encontrar e identificar tais serviços. Tomando-se um exemplo típico, um fornecedor de serviços descreve e publica a descrição de seus serviços para aqueles que irão requisitá-los. Os possíveis usuários dos serviços podem recuperar essa descrição e utilizá-la para identificar os aspectos de interface do *Web Service*. Após a descrição ser analisada, os serviços podem ser requisitados de forma correta, considerando-se os parâmetros de entrada e saída.

Um fator importante na comunicação baseada na arquitetura orientada a serviços refere-se à agilidade dos métodos e serviços envolvidos. Os serviços têm que ser respondidos com velocidade e flexibilidade, de modo que consigam atender a demanda dos clientes. Além disso, serviços devem ser dinâmicos, de alguma forma, devem poder se adaptar a novos requisitos [Krogdahl et al., 2005].

A estrutura básica de comunicação é definida utilizando-se a linguagem XML para classificação e estruturação das mensagens, e o protocolo HTTP para transporte. A arquitetura pode ser expandida com extensões da especificação XML, como a linguagem de definição XML Schema [W3C, 2001], e outros protocolos para transporte, como os protocolos UDP [Postel, 1980], TCP [Postel, 1981], BEEP [Rose, 2001].

A Figura 3.1 exibe a pilha de protocolos utilizados no funcionamento dos *Web Services*. No topo da pilha encontra-se a descoberta dos serviços, com o protocolo UDDI. Na segunda camada está situada a descrição dos serviços, com o protocolo WSDL. A terceira camada refere-se à troca de mensagens XML, com os protocolos XML-RPC, SOAP e XML. Na última camada está situado o transporte, com os protocolos HTTP, SMTP, FTP e BEEP.

FUNÇÃO	PROTOCOLOS
Descoberta	UDDI
Descrição	WSDL
Mensagens XML	XML-RPC, SOAP, XML
Transporte	HTTP, SMTP, FTP, BEEP

Figura 3.1: Pilha de protocolos dos *Web Services* [Cerami, 2002]

Através da descrição, é possível saber o formato das mensagens de envio e resposta, protocolos utilizados e a localização de cada serviço disponibilizado. Os detalhes de implementação ficam transparentes na descrição, permitindo que o agente requisitante possa utilizar o serviço independente da plataforma de *hardware* e da linguagem de programação na qual o serviço foi implementado.

Quando é necessário acessar algum serviço, a arquitetura básica de comunicação define que uma mensagem XML seja enviada pelo agente requisitante e recebida pelo *Web Service*. Essa mensagem XML é normalmente gerada pelo próprio agente requisitante e deve incluir elementos de cabeçalho e um corpo com os dados a serem trocados. Os elementos de cabeçalho são opcionais e contém informações como aspectos de segurança, roteamento ou existência de pontos intermediários no transporte da mensagem. O corpo da mensagem inclui o conteúdo das informações baseado na estrutura definida na descrição do *Web Service*. As mensagens são associadas com endereços únicos que identificam onde o agente deverá enviar a sua mensagem de requisição.

Figura 3.2: Arquitetura básica de *Web Services* [W3C, 2002]

Na Figura 3.2 é exibida uma arquitetura básica de *Web Services* na qual uma aplicação qualquer deseja requisitar serviços de um *Web Service*. Nessa arquitetura utiliza-se o protocolo de empacotamento SOAP e o protocolo HTTP para o transporte das mensagens. A mensagem SOAP da aplicação é inserida no corpo da mensagem de requisição, a qual é processada pelo *Web Service*. Após o processamento, uma mensagem SOAP de resposta é gerada e inserida no corpo da mensagem de resposta para ser enviada de volta à aplicação requisitante.

Considerando-se a arquitetura básica de descoberta e requisição de serviços, podem ser definidos três papéis básicos [Ceramí, 2002]:

Provedor de serviços: É o fornecedor de *Web Services*, que disponibiliza serviços para eventuais solicitações.

Solicitador de serviços: É qualquer consumidor do *Web Service*. O Solicitador utiliza um *Web Service* existente estabelecendo uma conexão e enviando uma requisição XML.

Registrador de serviços: É um diretório centralizado de serviços. O Registrador provê um local central onde desenvolvedores podem publicar novos serviços ou procurar serviços existentes. Ou seja, é uma agência central de classificação e distribuição de serviços.

A Figura 3.3 ilustra os papéis básicos de uma arquitetura de *Web Services* e a relação entre eles.

Figura 3.3: Papéis Básicos na Arquitetura de *Web Services* [Cerami, 2002]

3.5 XML-RPC

XML-RPC é um protocolo que usa mensagens XML para realizar RPC's. Requisições são codificadas em XML e enviadas via HTTP POST. Respostas XML são embutidas no corpo da resposta HTTP. Pelo fato de XML-RPC ser independente de plataforma, é possível que diversas aplicações, em diferentes plataformas, se comuniquem. Por exemplo, um cliente Java pode se comunicar, via XML-RPC, com um servidor PHP ou Perl.

Conforme exibido na Figura 3.1, dois protocolos podem ser utilizados para troca de mensagens XML, XML-RPC e SOAP. Em muitos casos, XML-RPC é mais simples e, por isso, mais fácil de se adotar. Entretanto, ao contrário do SOAP, XML-RPC não tem uma gramática correspondente de serviço de descrição. Isso impede a invocação automática de serviços XML-RPC, uma característica crucial para a integração de aplicações *just-in-time*.

3.6 SOAP

O protocolo de comunicação de um *Web Service* define a sintaxe, a semântica e a ordem das mensagens que serão trocadas entre aplicações e servidores. SOAP (*Simple Object Access Protocol*) é um protocolo de comunicação de dados extensível para troca de informações em ambientes descentralizados e distribuídos [Gudgin et al., 2001]. SOAP utiliza um protocolo de transporte e define XML como a linguagem de estruturação para suas mensagens. Além disso, SOAP define um conjunto de regras para serialização e convenção de chamadas entre sistemas de *software*.

Outras linguagens provêm funcionalidades similares ao SOAP, como CORBA, Java RMI e DCOM, mas SOAP é escrito completamente em XML e independente de plataforma

e linguagem. Por exemplo, um cliente SOAP Java rodando em Linux ou um cliente Pearl rodando em Solaris podem conectar-se a um servidor SOAP Microsoft rodando no Windows 2000 [Cerami, 2002].

A especificação SOAP define três partes principais [Gudgin et al., 2001]:

Envelope SOAP

Define as regras para encapsulamento de dados que serão transferidos entre dois computadores. Isso inclui dados específicos das aplicações, como nome de métodos, parâmetros ou valores de retorno. Pode também incluir informações sobre quem deverá processar o seu conteúdo e, em caso de falha, como codificar mensagens de erro.

Regras de codificação

Para intercambiar dados, computadores devem estabelecer regras específicas de codificação de dados. Por exemplo, dois computadores que processam preços de ações precisam de uma regra para codificar tipos de dados flutuantes. SOAP inclui seu próprio conjunto de convenções para codificação de tipos de dados, a maioria baseada em recomendações da especificação W3C XML Schema [W3C, 2001].

Convenções RPC

SOAP pode ser usada em uma variedade de sistemas de troca de mensagens, incluindo unidirecionais ou bidirecionais. Para troca bidirecional, SOAP define uma convenção simples para representação de chamadas e respostas de procedimentos remotos. Isso permite que uma aplicação cliente especifique o nome de um método remoto, inclua alguns parâmetros (se necessário), e receba a resposta do servidor.

Mensagens SOAP possuem uma estrutura que contém o elemento envelope como elemento principal e dois elementos internos: *header* e *body*. O conteúdo desses elementos internos é definido pelas aplicações e não fazem parte da especificação do protocolo. O elemento *header*, opcional, define informações como rota de transmissão e meta-informações relativas ao processamento da mensagem. Esse elemento possibilita a extensão da mensagem de forma específica por cada aplicação. O elemento *body*, obrigatório, define a informação que será transmitida na mensagem.

Embora ainda esteja em um estágio relativamente inicial de desenvolvimento, SOAP têm recebido um suporte considerável das indústrias de *software*. Já estão disponíveis inúmeras implementações SOAP, incluindo Java, Pearl, COM, Python e C#. Além disso, centenas de serviços SOAP já estão disponíveis na *Web*.

3.7 WSDL

Para expressar informações estruturais e semânticas, uma linguagem de descrição é utilizada. Essa linguagem é a WSDL (*Web Service Description Language*). Com as especificações contidas nesta linguagem, sistemas de *software* poderão gerar automaticamente implementações para a comunicação de aplicações com os *Web Services*. Baseado nisso, a linguagem WSDL é o formato de representação utilizado. A linguagem de descrição WSDL, também baseada em XML, permite a descrição completa dos serviços fornecidos por um *Web Service* [W3C, 2004b].

Através da linguagem WSDL, é possível que *Web Services* publiquem a interface pública de seus serviços, ou seja, o formato das mensagens de requisição e resposta. Além disso, WSDL é naturalmente extensível, já que a especificação define as interfaces abstratas e as estruturas de relacionamento, permitindo que a implementação e a formatação das mensagens sejam realizadas através de extensões.

Através da interface pública de um *Web Service* é possível saber informações sobre todas as funções disponíveis, tipos de dados para todas as mensagens XML, informações de ligação sobre o protocolo de transporte usado e informações de endereçamento para localização do serviço especificado. Usando a WSDL, um cliente pode localizar um *Web Service* e invocar qualquer de suas funções disponíveis. Com a utilização de ferramentas de apoio, esse processo pode ser feito de forma automática, permitindo que aplicações facilmente integrem novos serviços com pouco ou nenhum esforço para geração de código. Dentre as ferramentas que processam WSDL e facilitam o trabalho do desenvolvedor, estão a IBM *Web Services* Invocation Framework (WSIF) [The Apache Software Foundation, 2005] e o *Plugin* WSDL2JAVA [MySpotter Consulting Services, 2005] para a ferramenta Eclipse [Eclipse Foundation, 2005].

Resumindo, a linguagem WSDL representa o contrato entre o requerente do serviço e seu provedor, permitindo descrever quatro partes essenciais de dados:

- Informações da interface que descreve as funções disponíveis.
- Informações sobre os tipos de dados para todas as mensagens de requisições e respostas.
- Informações sobre a ligação do protocolo de transporte a ser usado.
- Informação de endereçamento para localização do serviço especificado.

A especificação WSDL baseia-se na gramática XML para descrição dos *Web Services* e divide-se em oito elementos, conforme descritos a seguir.

definitions

Deve ser o elemento raiz de todo documento WSDL. Define o nome do *Web Service*, declara *namespaces* e contém todos os elementos restantes do documento.

types

Descreve todos os tipos de dados usados entre cliente e servidor. Se o serviço usa apenas tipos simples baseados em XML-Schema, como *strings* e inteiros, esse elemento pode ser dispensado.

message

Descreve uma mensagem unidirecional, seja ela uma mensagem de requisição ou resposta. Define o nome da mensagem e contém zero ou mais elementos filhos, os quais podem referenciar parâmetros da mensagem requisição ou valores de retorno.

portType

Combina múltiplos elementos *message* para formar uma operação completa unidirecional ou de ida e volta. Por exemplo, pode combinar uma requisição e uma resposta dentro de uma única operação requisição/resposta.

binding

Descreve as especificações concretas de como o serviço será implementado na rede e informações específicas SOAP.

service

Define o endereço para invocar o *Web Service* especificado.

import

Permite importar outro documento WSDL ou XML Schema. Isso permite a construção de documentos WSDL mais modulares. Por exemplo, dois documentos WSDL podem importar os mesmo elementos básicos e ainda incluir seus próprios elementos *service* para tornar o mesmo serviço disponível em dois endereços diferentes.

documentation

É usado para disponibilizar uma documentação legível pelo homem e pode ser incluído dentro de qualquer um dos elementos descritos.

3.8 UDDI

A UDDI (*Universal Description, Discovery and Integration*) representa a camada de descoberta do serviço dentro da pilha de protocolos exibida na Figura 3.1. UDDI representa a especificação técnica para publicação e descoberta de *Web Services*.

UDDI consiste basicamente de duas partes. A primeira é a especificação UDDI, uma especificação técnica para a descoberta distribuída de *Web Services* e negócios. Dados são armazenados dentro de um formato específico de XML. Essa especificação inclui detalhes de API para busca de dados existentes e para publicação de novos dados. A segunda parte refere-se ao Registro UDDI “*UDDI Business Registry*”, uma implementação completamente operacional da especificação UDDI.

Lançado em maio de 2001 pela Microsoft e IBM, o UDDI *Registry* permite que qualquer um pesquise dados UDDI existentes ou que uma companhia se registre e registre seus serviços.

Os dados contidos dentro de um UDDI dividem-se em três principais categorias, conforme descritas a seguir.

White Pages

Inclui informações gerais sobre uma companhia, como nome do negócio, descrição do negócio, informação de contato e endereço.

Yellow Pages

Nessa categoria são incluídos os dados gerais de classificação da companhia ou do serviço oferecido.

Green Pages

Essa categoria inclui informações técnicas sobre um *Web Service*, como um ponteiro para uma especificação externa e um endereço para a invocação do *Web Service*.

UDDI cria uma plataforma padrão que permite que companhias e aplicações busquem e descubram *Web Services* de forma rápida, fácil e dinâmica. Além disso, UDDI também permite que registros operacionais sejam mantidos por diferentes propósitos em diferentes contextos [Oasis Open, 2005].

3.9 Considerações Finais

Este capítulo abordou os conceitos e características de *Web Services*, sistemas que disponibilizam um meio de comunicação padrão entre diferentes aplicações de *software*, permitindo a interoperabilidade e extensibilidade entre elas, além de permitir a combinação de aplicativos para a execução de tarefas mais complexas. Desta forma, os *Web Services* podem fornecer uma plataforma para o desenvolvimento de sistemas de *software* distribuídos que possam se comunicar independente das possíveis diferenças existentes entre eles, como sistema operacional ou linguagem de programação.

A comunicação e troca de mensagens é realizada utilizando o protocolo SOAP, que define um modelo extensível para o intercâmbio de mensagens XML de um sistema de *software* para outro. O protocolo SOAP define como essas mensagens serão enviadas ao *Web Service* utilizando um determinado protocolo de transporte. A mensagem SOAP de uma aplicação é inserida no corpo da mensagem de requisição, que é então enviada ao *Web Service* e processada por ele. Após o processamento, uma mensagem SOAP de

resposta é gerada e incluída no corpo da mensagem de resposta, para então ser enviada de volta à aplicação requisitante.

Atualmente, o protocolo HTTP é amplamente utilizado pelos *Web Services* para transporte de suas mensagens. A linguagem WSDL é utilizada para a descrição dos *Web Services*, a qual permite disponibilizar a definição da interface e a identificação dos serviços fornecidos.

Por ser uma tecnologia relativamente recente, alguns pontos fracos ainda estão em desenvolvimento. Em virtude disso, há a necessidade de implementação de mecanismos que fortaleçam alguns pontos que foram pouco explorados por seus desenvolvedores, como segurança, confiabilidade na troca de mensagens e especificação de suas informações. No entanto, os benefícios providos estão tornando os *Web Services* amplamente aceitos pelas principais companhias de desenvolvimento de *software*, as quais estão promovendo a iniciativa do uso de *Web Services* em seus novos produtos.

WS4CSCW

4.1 Considerações Iniciais

Ultimamente têm sido notados grandes esforços na integração de aplicações e componentes de *software* [Flores et al., 2005], [Vieira and Fortes, 2004]. Com o objetivo de agregar funcionalidades e melhorar os potenciais usos dos sistemas de *software*, aplicações integradas possuem, em geral, mais funcionalidades e maiores possibilidades de uso do que aplicações isoladas.

Uma característica comum na maioria das aplicações CSCW refere-se ao controle e gerenciamento de usuários e grupos. Quando essas ferramentas são integradas para trabalhar em conjunto, com o objetivo de acrescentar funcionalidades e facilidades ao usuário ou grupos de usuários, é possível que essas informações fiquem redundantes ou mesmo inconsistentes caso sejam tratadas isolada e localmente. Através de *Web Services*, aplicações CSCW podem ser desenvolvidas e integradas mesmo estando geograficamente distribuídas, tornando o desenvolvimento mais fácil e rápido, já que é possível aproveitar funcionalidades já implementadas.

Assim, foi desenvolvido o WS4CSCW (*Web Services for Computer Supported Cooperative Work*), um *Web Service* que tem como objetivo facilitar a integração de ferramentas e sistemas CSCW (*Computer Supported Cooperative Work*). O WS4CSCW disponibiliza funcionalidades para aplicações voltadas ao trabalho em grupo e permite o gerenciamento de usuários, grupos, recursos, permissões e papéis de usuários e grupos, com o objetivo de facilitar a integração entre essas aplicações, permitindo um co-funcionamento mais direto, consistente e coeso.

O WS4CSCW foi implementado na linguagem Java e o banco de dados usado foi o MySQL¹. Foram usadas as ferramentas Eclipse², para desenvolvimento e testes, Argo

¹<http://www.mysql.com/>

²<http://www.elipse.org>

UML³, para projeto e modelagem UML, e DBDesigner 4⁴, para projeto e modelagem do banco de dados. Também foram usados o servidor Apache Tomcat⁵ e a implementação SOAP Apache Axis⁶.

Este capítulo está organizado da seguinte maneira: na Seção 4.2 são apresentados os conceitos relacionados ao modelo de permissões e às funcionalidades do WS4CSCW. Na Seção 4.3 são descritos os principais cenários possíveis de uso do WS4CSCW, de forma a ilustrar como as aplicações CSCW podem tirar proveito dos serviços fornecidos e serem integradas. Na Seção 4.4 é apresentada a modelagem UML do WS4CSCW, com o diagrama de classes e de casos de uso. Os serviços do WS4CSCW são descritos na Seção 4.5. Na Seção 4.6 são descritos alguns trabalhos relacionados à integração de aplicações CSCW e ao WS4CSCW. Na Seção 4.7 são discutidos dois cenários de integração de aplicações CSCW como forma de avaliar e validar o WS4CSCW.

4.2 Conceitos

Uma característica comum na maioria das aplicações CSCW refere-se ao controle e gerenciamento de usuários e grupos. Quando essas ferramentas são integradas para trabalhar em conjunto, com o objetivo de acrescentar funcionalidades e facilidades ao usuário ou grupos de usuários, é possível que essas informações fiquem redundantes ou mesmo inconsistentes, caso sejam tratadas isolada e localmente.

O gerenciamento centralizado, conforme descrito na seção 2.4.2, objetiva resolver esse problema. Através de chamadas remotas ao WS4CSCW, as aplicações que estão cadastradas em um determinado contexto de integração podem acrescentar, modificar, remover e consultar informações referentes a usuários e grupos de usuários.

Outra característica referente a aplicações CSCW diz respeito a permissões e acessos a recursos por parte dos usuários, grupos e aplicações. O WS4CSCW também permite o gerenciamento e compartilhamento de permissões a recursos por usuários e grupos. Desta forma, é possível que uma aplicação altere a permissão que um usuário ou grupo terá em um recurso de outra aplicação ou sistema.

³<http://argouml.tigris.org/>

⁴<http://fabforce.net/dbdesigner4/>

⁵<http://tomcat.apache.org/>

⁶<http://ws.apache.org/axis/>

O modelo de permissões utilizado é baseado no RBAC (*Role-based access control*) [Ferraiolo and Kuhn, 1992, Ferraiolo et al., 2003], um modelo de permissões que considera papéis para controlar o acesso dos usuários.

O modelo RBAC tem sido utilizado mais recentemente como uma alternativa amplamente aceitável para modelos de controle de acesso. RBAC é uma tecnologia de controle de acesso bastante flexível, baseando-se nos conceitos de usuário, papel e permissão. Usuários são autorizados a usar as permissões que estão associadas aos papéis que estão associados aos usuários. Isso simplifica o gerenciamento, pois seria muito mais complexo gerenciar os usuários e permissões individualmente. Além de reduzir o custo de administrar as políticas de controle de acesso, esse método torna o processo de gerenciamento menos suscetível a erros. Trabalhos atuais têm direcionado esforços no desempenho e comportamento de sistemas concorrentes em um cenário RBAC [Braghin et al., 2004].

Na Figura 4.1 é ilustrado o relacionamento de permissões, papéis e usuários do modelo de permissões RBAC. Esse modelo não considera a utilização de papéis para grupos.

Figura 4.1: Relacionamentos do modelo RBAC [Ferraiolo and Kuhn, 1992]

No WS4CSCW é implementada a funcionalidade de papéis de grupos. Na Figura 4.2 é ilustrado um exemplo. Numa instituição que possui inúmeros grupos de professores (professores de computação, matemática, física, química, etc) e cada um destes grupos possui suas respectivas permissões a recursos (professores de matemática têm acesso ao laboratório cálculo, mas os outros não). Além disso, pode ser necessário atribuir permissões a todos os professores, ou a vários grupos. Então, pode ser criado um papel de grupo, atribuir ao papel as permissões necessárias, e associar este papel aos grupos de profes-

sores, ao invés de atribuir as permissões individualmente a cada um dos professores ou grupo de professores. Quando for necessário alterar alguma destas permissões para todos os grupos, basta alterá-la no papel que está atribuído aos grupos de professores. Quando é criado um novo grupo, basta associar o papel de grupo a este grupo, que herdará todas as permissões atribuídas ao papel.

Da mesma forma, também é implementado o papel de usuário. No mesmo exemplo da Figura 4.2, o cargo de administrador de laboratório possui um conjunto de permissões e esse cargo é revezado frequentemente entre os professores. Então, é criado um papel de usuário que armazena o conjunto de permissões e este papel é atribuído ao professor que atualmente administra o laboratório.

Figura 4.2: Exemplo do modelo de permissões usado no WS4CSCW

No WS4CSCW, o conceito de recurso é bem amplo, e pode representar, por exemplo, uma funcionalidade de uma aplicação, acesso a documentos, arquivos, base de dados, subsistemas, ou mesmo, restrições dentro de uma mesma funcionalidade. Por exemplo,

considerando-se uma aplicação de edição de projetos por grupos “Application Ap1”, é possível cadastrar uma permissão que indica que um certo grupo “Group G5” tem permissão de uso da Funcionalidade F1 desta aplicação apenas para o projeto P4.

Desta forma, com um conjunto de permissões armazenadas para cada usuário e grupo, é possível responder aos seguintes tipos de consultas:

- *Quem são os usuários do grupo G?*
- *Quais os grupos aos quais o usuário U pertence?*
- *Qual permissão o usuário U tem para o recurso R?*
- *Qual permissão o grupo G tem para o recurso R?*
- *O usuário U tem a permissão de valor V para o recurso R?*
- *Quais usuários têm permissão para o recurso R?*
- *Quais grupos têm permissão para o recurso R?*

O WS4CSCW também possibilita a utilização do conceito de papéis. Um papel possui várias permissões associadas a ele e permite que um usuário ou grupo herde essas permissões de uma única vez, por exemplo, quando é cadastrado ou inserido em algum grupo. Com isso, é possível, por exemplo, que a um novo usuário, membro de um grupo de desenvolvimento de projetos, sejam atribuídas todas as permissões previamente associadas ao papel de usuário “desenvolvedor”, ou que a um novo grupo de analistas sejam atribuídas as permissões do papel de grupo “analistas”.

O WS4CSCW tem como objetivo auxiliar ferramentas CSCW, bem como possibilitar a integração de ferramentas CSCW. Mesmo trabalhando de forma isolada, sem comunicação com outras aplicações, uma aplicação CSCW também pode usufruir os serviços WS4CSCW, eliminando da fase desenvolvimento as complicações relativas ao gerenciamento de usuários, grupos, recursos e permissões. Assim, são priorizados os esforços do desenvolvimento nas particularidades e funcionalidades da nova aplicação e aproveitados os serviços do WS4CSCW.

Pesquisadores na área têm discutido a necessidade de infra-estrutura para apoiar o desenvolvimento de aplicações CSCW [Roseman and Greenberg, 1996,

Chabert et al., 1998]. Uma das abordagens consiste na criação de componentes que implementam funções que podem ser utilizadas em muitos cenários. O conceito é permitir que novas aplicações sejam desenvolvidas reutilizando componentes que implementem as funções desejadas. Nesse caso, os componentes funcionam como suporte para que as aplicações executem em um sistema distribuído, e a infra-estrutura trata problemas relativos à sincronização de eventos, controle de fluxo, compartilhamento, etc. O WS4CSCW segue esse conceito e disponibiliza funcionalidades que podem ser usadas por novas aplicações. Desta forma, os esforços de projeto e desenvolvimento podem ser concentrados nas questões mais específicas das aplicações, que não sejam o gerenciamento de grupos, usuários, recursos e permissões.

4.3 Cenário de Uso

Na Figura 4.3 são ilustrados os principais cenários de uso do WS4CSCW. As aplicações podem usar o WS4CSCW em dois casos:

- O primeiro refere-se ao contexto de integração de aplicações voltadas ao trabalho em grupo. Nesse caso, são compartilhadas pelas aplicações as informações referentes a usuários, grupos, recursos e permissões.
- O segundo cenário refere-se a uma aplicação que deseja utilizar as funcionalidades do WS4CSCW para simplificar seu projeto e implementação, priorizando o esforço de desenvolvimento em outras funcionalidades que não as já fornecidas pelo WS4CSCW.

Assim, na Figura 4.3, temos que no contexto de integração as aplicações App1 e App2 compartilham as informações de grupo (a) e a App1 exporta funcionalidades (d), também através de *Web Services*, para a App2. Em outro contexto de integração, as aplicações App3 e App4 trocam funcionalidades via *Web Services* (e), e a App4 faz uso do WS4CSCW, também via *Web Services* (c), para gerenciamento das informações de grupo. E em outro possível uso, para reutilização de funcionalidades e simplificação de projeto, a aplicação App5 utiliza o WS4CSCW (b) para simplificar seu projeto, concentrando os esforços de desenvolvimento em outros problemas, que não se referem ao gerenciamento realizado pelo WS4CSCW.

Figura 4.3: Cenários de uso do WS4CSCW

4.4 Modelagem UML

Para melhor documentar o funcionamento do WS4CSCW com o intuito de facilitar futuras extensões, o sistema foi documentado utilizando a notação UML [Object Management Group, 2004]. Essa documentação não é necessária para a utilização do WS4CSCW, já que apenas a definição WSDL é necessária para a invocação de um *Web Service*. O arquivo WSDL do WS4CSCW encontra-se no apêndice deste documento.

Na Figura 4.4 é apresentado o diagrama de classes referente ao WS4CSCW. Um contexto de integração, representado pela classe *Context* no diagrama, pode ter várias aplicações, representadas pela classe *App*, cadastradas para fazerem uso dos serviços do WS4CSCW. As aplicações são os usuários do WS4CSCW, ou seja, são elas que fazem as requisições e obtêm as respostas das solicitações. O contexto pode ter vários recursos (classe *Resource* no diagrama), usuários (classe *User*) e grupos (classe *Group*) cadastrados, cada usuário podendo fazer parte de um ou vários grupos. Além disso, há as permissões de usuário (classe *UserPerm*), que é representada por um valor de acesso a um recurso. Uma permissão de usuário pode ser atribuída a um usuário específico ou a um papel de usuário (classe *UserRole*). Um papel de usuário é um conjunto de permissões de usuário

Figura 4.4: Diagrama de Classes do WS4CSCW

que pode ser atribuído a um usuário. O raciocínio é análogo para as permissões de grupo (classe *GroupRole*) e papéis de grupo (classe *GroupRole*).

Uma permissão de usuário possui o atributo *priorityOverGroup*, que permite definir se a permissão tem prioridade sobre uma eventual permissão de grupo para o mesmo recurso. Por exemplo, se um usuário possui permissão para um recurso e esse usuário é membro de um grupo que também tem permissão para o mesmo recurso, a permissão do usuário pode prevalecer ou não sobre a permissão do grupo.

Uma permissão de grupo possui os atributos tipo (*type*, no diagrama) e (*rank*). O tipo da permissão pode assumir os valores “*cumulative*” ou “*rank*”. Quando um usuário pertence a dois ou mais grupos que possuem permissão para um dado recurso, e essas permissões são do tipo “*cumulative*” o usuário terá como permissão resultante para aquele recurso a reunião das permissões dos grupos aos quais ele pertence. Caso a permissão seja do tipo “*rank*”, o usuário terá como permissão resultante ao recurso a permissão que tiver o maior valor de *rank* (nível da permissão).

Figura 4.5: Diagrama de Casos de Uso do WS4CSCW

Conforme exibido na Figura 4.5, o WS4CSCW pode ter dois possíveis usuários de seus serviços, *Modify Application* e *Consult Application*. O primeiro tipo de usuário, na forma de uma aplicação, tem permissões para adicionar, remover e alterar usuários, grupos, recursos, permissões, papéis e as associações entre estes (adicionar e remover usuários de grupos, atribuir e remover permissões de usuários, grupos, papéis, etc.) e fazer consultas, como quais usuários pertencem a determinado grupo ou qual a permissão de determinado usuário para um dado recurso. O segundo tipo de aplicação tem permissão apenas para realizar consultas.

Quando é instanciado um contexto de integração, a ele são atribuídos dois identificadores: um público e um privado. De posse do identificador privado, é possível registrar aplicações que farão parte deste contexto, intercambiando funcionalidades e compartilhando informações referentes aos grupos e usuários. Quando uma aplicação é registrada (adicionada ao contexto), a ela é atribuído um identificador.

Uma aplicação registrada possuirá, além de seu identificador, um dos identificadores do contexto a que está associada (identificador público ou privado). Caso possua o identificador privado, a aplicação se enquadra no tipo “*Modify Application*”, descrito anterior-

mente. Caso possua o identificador público, ela é classificada como “*Consult Application*”.

Essa abordagem permite que uma mesma instância instalada do WS4CSCW permita o funcionamento de vários contextos de integração. Por exemplo, o contexto *NRP-iClass* refere-se à integração das aplicações *No Risk Planning* e *iClass*, conforme descritas na seção 4.7.2. Já o contexto *ICMC* refere-se à integração de outras aplicações de apoio ao ensino, e esta outra integração possui informações referentes a diferentes usuários e grupos. Além disso, é possível restringir o acesso à modificação de informações pelas aplicações, de modo que apenas as aplicações autorizadas realizem as alterações e outras possam apenas executar consultas.

4.5 Serviços do WS4CSCW

Nesta seção são descritos os serviços disponibilizados pelo WS4CSCW. Esses serviços relacionam-se a: contexto, aplicação, usuário, grupo, recurso, permissão e papel. A seguir são descritos, para cada uma das entidades a que se referem os serviços, o funcionamento, entrada e saída dos serviços disponíveis. No apêndice é exibido o documento WSDL referente ao serviço. Com exceção do serviço *createContext*, todos os serviços requerem autenticação. Para autenticação, são passadas duas informações, o identificador da aplicação, que uma aplicação recebe no momento que se cadastra em determinado contexto de integração, e o identificador do contexto, que pode ser público ou privado, e são fornecidos no momento de criação de um contexto. De posse do identificador privado, uma aplicação caracteriza-se como “*Modify Application*”. Se a aplicação possui apenas o identificador público, ela é caracterizada como uma “*Consult Application*”, podendo apenas solicitar consultas ao WS4CSCW. Uma descrição técnica do WS4CSCW e seus serviços é feita em [Vieira and Fortes, 2005].

4.5.1 Contexto

- **createContext:**

Cria um novo contexto e retorna um identificador privado e um público referentes a este contexto. Tem como parâmetros de entrada o nome e a descrição do contexto, conforme exibido no trecho XML a seguir.

```
<context>
```

```

    <name>No Risk - iClass</name>
    <description>Contexto de integração das aplicações No Risk e iClass</
        description>
</context>

```

Esse serviço retorna dois identificadores, um público e um privado (para serem usados pelas aplicações para autenticação), ou uma mensagem de erro, caso ocorra algum problema ou já exista um contexto cadastrado com esse nome. O fragmento XML a seguir descreve um exemplo de retorno na criação de um contexto de integração no WS4CSCW.

```

<return>
    <privateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</privateId>
    <publicId>LMRg22yorSFXhDLaYOvDciomz0VXF</publicId>
</return>

```

4.5.2 Aplicação

- **addApp:**

Adiciona uma aplicação e a associa a um contexto existente (dado pelo identificador) e retorna o identificador privado correspondente. Tem como parâmetros de entrada o identificador privado do contexto ao qual a aplicação se refere, o nome e a descrição da aplicação, conforme exibido no trecho XML a seguir.

```

<application>
    <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
    <name>No Risk Planning</name>
    <description>Agenda web com suporte a grupos</description>
</application>

```

Esse serviço retorna um identificador (também para autenticação), em caso de sucesso ou uma mensagem de erro, caso não seja possível realizar o serviço, ou o identificador do contexto não é válido ou já existe alguma aplicação cadastrada com esse nome. O trecho XML, a seguir, exemplifica o retorno do identificador da aplicação.

```

<return>
    <privateId>b9mZDNF2lQh93DmvCeERzPUfdotgWj</privateId>
</return>

```

4.5.3 Usuário

- **addUser:**

Esse serviço adiciona um usuário a um contexto existente. Tem como entrada o identificador privado do contexto e o identificador da aplicação para validação da requisição. Para cadastro de um novo usuário são fornecidos o nome, *nickname*, email e descrição do usuário, conforme exemplificado a seguir.

```
<user>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <name nick="rpontin">Renata Pontin</name>
  <email>renata@icmc.usp.br</email>
  <description>Professora</description>
</user>
```

É retornada uma confirmação de inclusão do novo usuário ou uma mensagem de erro, caso a validação seja inválida ou o usuário já esteja cadastrado.

- **removeUser:**

Esse serviço remove um usuário de um contexto de integração. Tem como entrada os parâmetros para validação e o nome do usuário que se deseja remover. É retornada uma mensagem de confirmação da exclusão ou uma mensagem de erro, em caso de falha na autenticação ou caso o usuário com o nome dado não esteja cadastrado no contexto.

- **addUserertoGroup:**

Adiciona um usuário a um grupo já cadastrado. Como entrada devem ser fornecidos os identificadores do contexto e da aplicação, o nome do usuário e nome do grupo que se deseja cadastrar o usuário, conforme exibido no código a seguir.

```
<addUserertoGroup>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Renata Pontin</userName>
  <groupName>Professores SCE</groupName>
</addUserertoGroup>
```

É retornada uma mensagem de sucesso ou de erro, caso haja falha na autenticação.

- **removeUserfromGroup:**

Esse serviço remove um usuário de um grupo. Como entrada são fornecidos os identificadores para autenticação, o nome do grupo e do usuário. É retornada uma confirmação da remoção ou uma mensagem de erro, em caso de falha na autenticação, usuário ou grupo desconhecidos ou caso o usuário não pertença ao grupo em questão.

- **getUsersinGroup:**

Esse serviço retorna os nomes dos usuários pertencentes ao grupo dado. São fornecidos os identificadores de autenticação e o nome do grupo, conforme exemplificado no código a seguir.

```
<getUsers>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <groupName>Analistas Suporte</groupName>
</getUsers>
```

São retornados os nomes dos usuários pertencentes ao grupo, conforme exibido no código a seguir, ou uma mensagem de erro, em caso de falha na autenticação ou nome de grupo desconhecido para o contexto requisitado.

```
<users>
  <user nick="abueno">Alberto Bueno</user>
  <user nick="pcv">Paulo Cesar Vieira</user>
  <user nick="angelm">Silvana Maria</user>
  <user nick="grfther">Acacio Simoes</user>
</users>
```

- **getUsersinResourcePermissions:**

Esse serviço retorna quais usuários têm permissão para o recurso dado. É útil para saber quais usuários têm acesso a um determinado recurso. Como entrada são fornecidas as informações para autenticação e o nome de um recurso já cadastrado. São retornados os nomes dos usuários que têm permissão para o recurso fornecido ou uma mensagem de erro, em caso de falha na autenticação ou recurso desconhecido.

- **getUsersAndPermissionsfromResource:**

Esse serviço retorna quais usuários têm permissão para o recurso dado e suas respectivas permissões. Como entrada são fornecidas as informações para autenticação e o nome de um recurso. O código XML a seguir representa uma requisição a esse

serviço para a seguinte questão “Que usuários possuem permissão para o recurso ‘Inserir compromissos’ da aplicação ‘*No Risk Planning*’ e quais são as respectivas permissões?”.

```
<User>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <resourceName>Inserir compromissos – NRP</resourceName>
</User>
```

São retornados os nomes dos usuários que têm permissão para o recurso fornecido e os respectivos valores de permissões ou uma mensagem de erro, em caso de falha na autenticação ou recurso desconhecido. O código a seguir exemplifica um exemplo de retorno desse serviço, com os nomes dos usuários e seus valores de permissões para o recurso pesquisado.

```
<return>
  <permission user="Renata Fortes">Fixos</permission>
  <permission user="Cleide Simoes">Todos</permission>
  <permission user="Sebastiao Vieira">Eventuais</permission>
  <permission user="Greice Kerr">Fixos</permission>
</return>
```

- **getUsersfromResourcePermissionsValue:**

Esse serviço retorna quais usuários têm permissão de valor “*value*” para o recurso dado. Importante para saber quem são os usuários com determinado tipo de acesso ao recurso. Como entrada são fornecidas as informações para autenticação, o nome de um recurso e o valor de permissão a esse recurso. O trecho de código XML a seguir representa a seguinte consulta ao serviço “Quais usuários possuem a permissão de valor ‘Fixos’ para o recurso ‘Inserir compromissos’ da aplicação ‘*No Risk Planning*’?”.

```
<User>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <resource>
    <name>Inserir compromissos – NRP</name>
    <value>Fixos</value>
  </resource>
</User>
```

São retornados os nomes dos usuários que têm permissão de valor dado para o

recurso fornecido ou uma mensagem de erro, em caso de falha na autenticação ou recurso desconhecido.

4.5.4 Grupo

- **addGroup:**

Esse serviço cria um novo grupo e o adiciona a um contexto existente. Além das informações de autenticação para uma “*Modify Application*”, são fornecidos os dados referentes ao grupo: nome, categoria, sigla e descrição. O trecho de código XML a seguir exibe os dados de uma requisição ao serviço para cadastrar um novo grupo.

```
<group>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <name>Gerentes Informatica</name>
  <category>Profissional</category>
  <abbreviation>GI</abbreviation>
  <description>Gerentes dos departamentos de Informática</description>
</group>
```

É retornada uma mensagem de confirmação da criação, ou uma mensagem de erro, em caso de falha na autenticação ou se já existir um grupo cadastrado com o mesmo nome.

- **removeGroup:**

Esse serviço remove um grupo de um contexto de integração. Tem como entrada os parâmetros para validação e o nome do grupo que se deseja remover. É retornada uma mensagem de confirmação da exclusão ou uma mensagem de erro, em caso de falha na autenticação ou caso o grupo com o nome dado não esteja cadastrado no contexto.

- **getGroupsfromUser:**

Esse serviço permite saber quais grupos têm um determinado usuário cadastrado como um de seus membros. São fornecidos como entrada os identificadores para autenticação e o nome do usuário, conforme exibido no código XML a seguir para consulta dos grupos do usuário 'Machado de Assis'.

```
<Groups>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
```

```

    <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
    <userName>Machado de Assis</groupName>
  </Groups>

```

São retornados os nomes dos grupos aos quais o usuário fornecido pertence, conforme exibido no trecho XML a seguir, ou uma mensagem de erro, em caso de falha na autenticação ou caso o usuário seja inválido para o contexto requisitado.

```

<return>
  <groupName>Escritores Brasil</groupName>
  <groupName>Academia Brasileira de Letras</groupName>
</return>

```

4.5.5 Recurso

- **addResource:**

Esse serviço é responsável por adicionar um recurso a um contexto de integração existente. Além dos dados referentes à autenticação, devem ser fornecidas as informações referentes aos recurso: nome, *boolType*, e descrição. *boolType* refere-se aos possíveis tipos de permissões para o recurso. Caso o valor de *boolType* seja “Yes”, as permissões para esse recurso só poderão assumir valores booleanos. Caso contrário, podem assumir qualquer valor. O fragmento de código XML a seguir exemplificado a mensagem enviada ao serviço para o cadastro do recurso ‘Inserir compromissos’ na agenda *No Risk Planning* (NRP).

```

<resource>
  <contextPrivateId>JTcjKA4g9spXPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <name>Inserir compromissos – NRP</name>
  <description>Insercao de compromissos na agenda No Risk Planning</description>
  <boolType>True</boolType>
</resource>

```

É retornada uma mensagem de sucesso de adição ou uma mensagem de erro em caso de falha na autenticação ou caso um recurso com o mesmo nome já tenha sido cadastrado.

- **removeResource:**

Esse serviço remove um recurso de um contexto de integração. Tem como entrada os parâmetros para validação e o nome do recurso que se deseja remover. É retornada

uma mensagem de confirmação da exclusão ou uma mensagem de erro, em caso de falha na autenticação ou caso o recurso com o nome dado não esteja cadastrado no contexto.

4.5.6 Papel

- **addUserRole:**

Esse serviço cria um papel de usuário em um contexto de integração existente. São fornecidos os identificadores para autenticação, o nome e a descrição do papel de usuário. Um papel permite que sejam a ele associadas várias permissões a recursos. Quando um papel é então atribuído a um usuário, este usuário herda todas as permissões anteriormente atribuídas ao papel de usuário. Esse procedimento facilita o gerenciamento de usuários e permissões quando o número de permissões não é pequeno, e torna-se inviável atribuir permissões individualmente aos usuários. O código XML a seguir ilustra uma mensagem de requisição ao serviço para cadastrar um papel de usuário.

```
<userRole>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <name>Monitor Administrador</name>
  <description>Papel de monitores administradores da rede</description>
</userRole>
```

Desta forma, quando esse papel de ‘Monitor Administrador’ é atribuído a um usuário, esse usuário herda todas as permissões associadas ao papel. Como retorno, é enviada uma mensagem de sucesso na criação do papel de usuário, ou então uma mensagem de erro, em caso de falha na autenticação ou caso já exista um papel de usuário cadastrado com o nome fornecido.

- **addGroupRole:**

Semelhante ao papel de usuário, o papel de grupo permite que sejam associadas permissões a ele e, posteriormente, atribuí-lo a grupos. Esses grupos herdam as permissões associadas ao papel de grupo. O trecho de código a seguir exemplifica uma requisição ao serviço para a criação do papel de grupo dos analistas.

```
<groupRole>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
```

```

<appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
<name>Analistas</name>
<description>Papel dos grupos de analistas</description>
</groupRole>

```

Assim, quando esse papel for atribuído a um dos grupos de analistas (Analista de Sistemas, Analista de Negócios, Analista de Riscos, etc.), esse grupo herdará as permissões atribuídas ao papel de grupo 'Analistas'. Caso seja necessário alterar uma permissão para todos os grupos de analistas, basta fazer essa alteração no papel, e não em cada um dos grupos. É retornada uma mensagem de sucesso da criação do papel de grupo ou uma mensagem de erro, em caso de falha na autenticação ou caso já exista um papel de grupo com o nome fornecido cadastrado no contexto de integração em questão.

- **addUserRoletoUser:**

Esse serviço atribui um papel de usuário a um usuário. Este usuário herda automaticamente todas as permissões associadas ao papel de usuário. Como entrada são fornecidos os identificadores para autenticação, o nome do usuário e do papel de usuário. O trecho de código a seguir representa a requisição para associar o papel de usuário 'Monitor Administrador' ao usuário 'Paulo Vieira'.

```

<userRole>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Paulo Vieira</userName>
  <userRoleName>Monitor Administrador</userRoleName>
</userRole>

```

É retornada uma mensagem de sucesso na atribuição do papel ao usuário ou uma mensagem de erro, em caso de falha na autenticação, nome de usuário ou papel de usuário desconhecidos ou caso o papel já esteja associado ao usuário.

- **removeUserRolefromUser:**

Esse serviço desfaz a ação do serviço *addUserRoletoUser*, ou seja, remove a atribuição do papel de usuário ao usuário. Como consequência, o usuário perde as permissões que estão associadas ao papel. Como entrada são fornecidas as informações referentes à autenticação, o nome do usuário e do papel de usuário. É retornada uma mensagem de sucesso ou uma mensagem de erro, em caso de falha

na autenticação, nome de usuário ou papel inválidos ou caso o papel não esteja atribuído ao usuário.

- **addGroupRoletoGroup:**

Esse serviço é responsável por atribuir um papel de grupo a um grupo, ambos já cadastrados no contexto de integração. Esse grupo herda de forma automática todas as permissões a recursos anteriormente associadas ao papel de grupo. Ou seja, todos os usuários que fazem parte deste grupo também passam a ter as permissões anteriormente associadas ao papel de grupo. Como entrada são fornecidos os identificadores para autenticação, o nome do grupo e do papel de grupo. O trecho de código a seguir representa a requisição para associar o papel de grupo ‘Analistas’ ao grupo ‘Analistas de suporte’.

```
<groupRole>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <groupName>Analistas de suporte</groupName>
  <groupRoleName>Analistas</groupRoleName>
</groupRole>
```

Como retorno, é enviada uma mensagem de sucesso na atribuição ou uma mensagem de erro, em caso de falha na autenticação, nome de grupo ou papel de grupo desconhecidos ou caso o papel já esteja atribuído ao grupo.

- **removeGroupRolefromGroup:**

Esse serviço desfaz a atribuição realizada pelo serviço *addGroupRoletoGroup*. O grupo não está mais associado ao papel de grupo e, como consequência, não possui mais as permissões associadas ao papel de grupo. Da mesma forma, os usuários pertencentes ao grupo também não possuem mais as permissões associadas ao papel de grupo. Como entradas são fornecidos os identificadores do contexto e da aplicação, o nome do grupo e do papel de grupo. É retornada uma mensagem de sucesso ou uma mensagem de erro, em caso de falha na autenticação, nome de grupo ou papel de grupo desconhecidos ou caso o papel não esteja atribuído ao grupo.

4.5.7 Permissão

- **atribUserPermissiontoUser:**

Esse serviço atribui uma permissão de usuário para o recurso dado a um usuário. Além dos identificadores de contexto e de aplicação, são fornecidos os nomes do recurso e do usuário, o valor e a descrição da permissão e um flag “*priorityOverGroup*”, que permite definir se a permissão tem prioridade sobre uma eventual permissão de grupo para o mesmo recurso. Por exemplo, se um usuário possui permissão para um recurso e esse usuário é membro de um grupo que também tem permissão para o mesmo recurso, a permissão do usuário pode prevalecer ou não sobre a permissão do grupo. O fragmento de XML a seguir representa a atribuição, ao usuário ‘Renata Fortes’, da permissão de valor ‘Modulo financeiro’ do recurso ‘Acesso sistema ABC’ e com prioridade sobre eventuais permissões de grupos aos quais o usuário pertence.

```
<permission>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Renata Fortes</userName>
  <resourceName>Acesso sistema ABC</resourceName>
  <value>Modulo financeiro</value>
  <description>Possui acesso ao modulo financeiro do sistema ABC</description>
  <priorityovergroup>True</priorityovergroup>
</permission>
```

É retornada uma mensagem de sucesso na atribuição da permissão ou uma mensagem de erro em caso de falha na autenticação, caso o nome do recurso ou do usuário seja inválido, caso a permissão já esteja cadastrada ou caso o valor da permissão seja inválido. Um valor de permissão será inválido quando for diferente de valores booleanos para um recurso que foi definido como tendo permissões apenas do tipo booleano.

- **removeUserPermissionfromUser:**

Esse serviço remove de um usuário uma permissão para um dado recurso, ou seja, cancela a permissão do usuário, anteriormente atribuída, para um recurso. São fornecidos os identificadores, os nomes do recurso e do usuário e o valor da permissão, conforme exhibe o fragmento a seguir.

```
<permission>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Renata Fortes</userName>
  <resourceName>Acesso sistema ABC</resourceName>
  <value>Modulo financeiro</value>
```

```
</permission>
```

Em caso de sucesso, é retornada uma mensagem de confirmação da remoção da permissão. Caso contrário, é retornada uma mensagem de erro, em caso de falha na autenticação, nome de usuário ou recurso inválido ou caso a permissão não esteja cadastrada.

- **atribGroupPermissiontoGroup:**

Esse serviço atribui uma permissão de grupo para o recurso dado a um grupo. Além das informações para autenticação, devem ser fornecidos os nomes do recurso e do grupo, descrição, valor, tipo e nível da permissão. O tipo da permissão pode assumir os valores “*cumulative*” ou “*rank*”. Quando um usuário pertence a dois ou mais grupos que possuem permissão para um dado recurso, e essas permissões são do tipo “*cumulative*” o usuário terá como permissão resultante para aquele recurso a reunião das permissões dos grupos aos quais ele pertence. Caso a permissão seja do tipo “*rank*”, o usuário terá como permissão resultante ao recurso a permissão que tiver o maior *rank* (nível da permissão). O fragmento XML seguinte representa a atribuição ao grupo ‘Professores Banco de Dados’ da permissão do tipo *cumulative*, de consultar a base de dados ‘Orçamento’.

```
<permission>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <groupName>Professores Banco de Dados</groupName>
  <resourceName>Base de Dados Orçamento</resourceName>
  <value>Consulta</value>
  <description>Podem realizar consultas na base de dados Orçamento</description>
  <type>cumulative</type>
  <rank>100</rank>
</permission>
```

Esse serviço retorna uma mensagem de sucesso na remoção da permissão ou uma mensagem de erro em caso de falha na autenticação, caso o nome do recurso ou do grupo seja inválido, caso a permissão já esteja cadastrada ou caso o valor da permissão seja inválido. Um valor de permissão será inválido quando for diferente de valores booleanos para um recurso que foi definido como tendo permissões apenas do tipo booleano.

- **removeGroupPermissionfromGroup:**

Esse serviço remove de um grupo uma permissão para um dado recurso, ou seja, cancela a permissão do grupo, anteriormente atribuída, para um recurso. São fornecidos os identificadores, os nomes do recurso e do grupo e o valor da permissão. É retornada uma mensagem de confirmação da remoção da permissão ou uma mensagem de erro, em caso de falha na autenticação, nome de recurso ou grupo inválido ou caso a permissão não esteja cadastrada.

- **atribUserPermissiontoUserRole:**

Esse serviço é responsável por associar uma permissão de usuário a um papel de usuário. Um papel permite reunir um conjunto de permissões e, de um única vez, associá-las a um ou mais usuários. Da mesma forma, é possível desfazer essa associação, retirando várias permissões de um usuário de forma instantânea. Esse serviço recebe como entrada os identificadores do contexto e da aplicação, os nomes do recurso e do papel de usuário, a descrição e o valor da permissão e o flag “*priority-OverGroup*”. O trecho de código XML a seguir representa a mensagem de requisição para a atribuição, ao papel ‘Monitor Administrador’, da permissão controle total ao sistema de segurança.

```
<permission>
  <contextPrivateId>JTcjKA4g9spXPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userRoleName>Monitor Administrador</userRoleName>
  <resourceName>Acesso sistema segurança</resourceName>
  <value>Controle Total</value>
  <description>Monitores Administradores possuem total controle no sistema de
    segurança</description>
  <priorityovergroup>True</priorityovergroup>
</permission>
```

O serviço retorna uma mensagem de sucesso da associação da permissão ao papel de usuário ou uma mensagem de erro, em caso de falha na autenticação, nome de papel ou recurso inválido ou caso o valor da permissão seja inválido.

- **removeUserPermissionfromUserRole:**

Esse recurso desfaz a associação feita pelo serviço *atribUserPermissiontoUserRole*, ou seja, a permissão ao recurso não fica mais associada ao papel de usuário. Desta forma, todos os usuários aos quais o papel foi atribuído também não possuirão mais essa permissão ao recurso. O serviço recebe os dados referentes à autenticação,

nome do recurso e do papel e valor da permissão. É retornada ao requisitante uma mensagem de sucesso da remoção ou uma mensagem de erro, em caso de falha na autenticação, nome de papel ou recurso inválido ou caso a permissão não esteja associada ao papel.

- **atribGroupPermissiontoGroupRole:**

Esse serviço associa uma permissão de grupo a um papel de grupo. Um papel permite reunir um conjunto de permissões e associá-las a um ou mais grupos. Da mesma forma, é possível desfazer essa associação, retirando várias permissões de um grupo de uma única vez. Além das informações para autenticação, devem ser fornecidos os nomes do recurso e do papel de grupo, descrição, valor, tipo e nível da permissão. O trecho de código XML a seguir representa a mensagem de requisição para a atribuição, ao papel de grupo 'Analistas', da permissão de realizar telefonemas nacionais. Sendo esta permissão do tipo 'rank', caso um usuário pertença a outro grupo que também possua permissão para o recurso 'Telefonemas', será considerada a permissão com o maior 'rank'. Esse tipo de atribuição permite, por exemplo, modelar o cenário em que o usuário 'Paulo Vieira' pertença ao grupo 'Analista de negócio' e ao grupo 'Gerentes', e, portanto, possua as permissões de realizar telefonemas nacionais e internacionais, considerando-se a permissão exibida no código abaixo e supondo-se que tenha sido atribuída a permissão de valor 'Internacionais' ao grupo de Gerentes ou a um papel associado ao grupo.

```
<permission>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <groupName>Analistas</groupName>
  <resourceName>Telefonemas</resourceName>
  <value>Nacionais</value>
  <description>Grupos de analistas podem realizar telefonemas nacionais</
    description>
  <type>rank</type>
  <rank>50</rank>
</permission>
```

O serviço retorna uma mensagem de sucesso da associação da permissão ao papel de grupo ou uma mensagem de erro, em caso de falha na autenticação, nome de papel ou recurso inválido ou caso o valor da permissão seja inválido.

- **removeGroupPermissionfromGroupRole:**

Esse recurso desfaz a associação feita pelo serviço *atribGroupPermissionToGroupRole*, ou seja, a permissão ao recurso não fica mais associada ao papel de grupo. Assim, todos os grupos aos quais o papel foi atribuído também não possuirão mais essa permissão ao recurso. O serviço recebe os dados referentes à autenticação, os nomes do recurso e do papel e o valor da permissão. É retornada uma mensagem de sucesso da remoção ou uma mensagem de erro, em caso de falha na autenticação, nome de papel ou recurso inválido ou caso a permissão não esteja associada ao papel.

- **getUserPermissionfromUser:**

Esse serviço retorna qual permissão ou permissões o usuário possui para o recurso dado. Além dos identificadores para autenticação, devem ser fornecidos o nome do recurso e do usuário. Essa busca considera todas as permissões para um dado recurso atribuídas a um determinado usuário, as quais estão em um dos quatro subconjuntos a seguir:

1. Atribuídas diretamente ao usuário.
2. Atribuídas a papéis de usuário que estejam associados ao usuário.
3. Atribuídas a grupos aos quais o usuário pertence.
4. Atribuídas a papéis de grupos associados aos grupos aos quais o usuário pertence.

Para o cálculo da permissão(ões) resultante(s), o algoritmo considera o tipo da permissão de usuário (*priorityOverGroup*):

- **SIM:** são desconsideradas as permissões de grupo para o recurso, já que a permissão de usuário deve prevalecer. São retornadas apenas as permissões dos subconjuntos 1) e 2) descritos anteriormente. Isso pode ser bastante útil caso seja necessário fornecer ao usuário algum nível de permissão diferente (maior ou menor) da permissão associada ao(s) grupo(s) ao(s) qual(is) ele pertence.
- **NÃO:** são consideradas também as permissões associadas ao(s) grupo(s) ao qual(is) o usuário pertence. Isso pode ser útil caso a permissão atribuída ao usuário não precisa substituir a permissão atribuída ao(s) grupo(s) ao(s) qual(is) ele pertence. Ou seja, a permissão atribuída ao usuário deve ser adi-

cional, acumular-se com as permissões dos grupos. Neste caso, são retornadas as permissões dos subconjuntos 1), 2), 3) e 4) descritos anteriormente.

Com relação aos subconjuntos 3) e 4), há duas possibilidades:

- * Permissões de grupo para esse recurso são do tipo “*rank*”: é retornada apenas a permissão que possui o maior nível (*rank*), já que as permissões são classificadoras.
- * Permissões de grupo para esse recurso são do tipo “*cumulative*”: são retornadas todas as permissões (atribuídas a todos os grupos ou papéis de grupos associados a grupos aos quais o usuário pertence), já que as permissões são acumulativas.

O trecho XML a seguir exemplifica a mensagem de requisição para saber quais as permissões que o usuário 'Paulo Vieira' possui para o recurso 'Telefonemas'.

```
<permission>
  <contextPrivateId>JTcjKA4g9spxPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Paulo  Vieira</userName>
  <resourceName>Telefonemas</resourceName>
</permission>
```

O serviço retorna um conjunto de valores correspondentes às permissões do usuário para o recurso dado, ou uma mensagem de erro, em caso de falha na autenticação ou nome de recurso ou usuário inválidos para o contexto em questão. O código a seguir representa um possível retorno para a consulta exemplificada anteriormente.

```
<return>
  <value>Nacionais</value>
  <value>Locais</value>
  <value>Internacionais</value>
</return>
```

- **getUserHasPermissionValue:**

Esse serviço verifica se o usuário possui o valor de permissão dado para o um determinado recurso. Esse serviço é útil quando uma aplicação precisa saber se um usuário possui uma permissão específica para um recurso ou não, permitindo ou não o acesso. Além das informações de autenticação, são fornecidos o nome do usuário e do recurso, e o valor da permissão que se deseja verificar se o usuário possui ou não.

É retornado '1' caso o usuário possua a permissão ou '0' caso contrário, incluindo falha de autenticação ou nomes inválidos de usuário ou recurso. O trecho de código XML a seguir exibe uma consulta do tipo “o usuário 'Paulo Vieira' possui permissão de realizar ligações internacionais?”.

```
<permission>
  <contextPrivateId>JTcjKA4g9spXPPiM8knREeUdQLnck4</contextPrivateId>
  <appPrivateId>b9mZDNF2IQh93DmvCeERzPUfdotgWj</appPrivateId>
  <userName>Paulo  Vieira</userName>
  <resourceName>Telefonemas</resourceName>
  <value>Internacionais</value>
</permission>
```

Baseando-se então no retorno, a aplicação que realiza o controle ao recurso e que requisitou a consulta permite ou nega o acesso do usuário ao recurso.

4.6 Trabalhos relacionados

Na literatura, as pesquisas têm se mostrado promissoras para facilitar o suporte a aplicações CSCW. Assim, o BSCW (*Basic Support for Cooperative Work*) permite o trabalho cooperativo via rede [Bentley et al., 1997]. Tem como objetivo ser acessível a partir de browsers convencionais, sem a necessidade de instalação de ferramentas adicionais nos clientes. O BSCW fornece funcionalidades básicas para cooperação de grupos e utiliza a web como infra-estrutura de comunicação. Possui recursos para armazenamento de vários tipos de documentos, gerenciamento de versões, administração de membros e grupos e edição colaborativa de documentos, podendo conter vários tipos de objetos (documentos, imagens ou *links*). Os objetos são organizados em hierarquias de pastas. Os membros de um determinado grupo podem postar objetos no espaço compartilhado ou transferir objetos deste espaço para o seu sistema local [Klockner, 2000]. Em [Appelt, 2001], baseando-se no histórico de utilização do BSCW, são analisadas quais funcionalidades os usuários mais utilizam, dentre elas: autenticação, fóruns de discussão, buscas, conversão de formatos de documentos, controle de versões e suporte a vários idiomas.

Algumas possibilidades para usar *Web Services* como suporte a *groupware* têm sido propostas. Uma motivação para usar *Web Services* é a possibilidade de se ter recursos e aplicações de interesse para membros de grupos residindo em vários sistemas de forma

distribuída, e não em um único servidor central, caracterizando um sistema totalmente distribuído [Kammer, 2002].

Uma interessante arquitetura, baseada em *Web Services*, para aglutinar vários sistemas de *groupware* foi proposta na literatura. A arquitetura consiste em três níveis de *Web Services* para suporte a *groupware*: negócios, aplicações e gerenciamento de serviço. O uso desta arquitetura por sistemas de *groupware* proveria uma camada de *middleware* de trabalho em equipe com os mecanismos e abstrações requeridos por cenários de colaboração móveis e distribuídos. [Dustdar et al., 2004].

Essa camada de *middleware* torna possível a adequação de aplicações *groupware* através de interfaces que permitem a colaboração entre aplicações, mas sem restringir as vantagens e características das aplicações. Essa abordagem prioriza a disponibilização de uma camada de abstração de serviços de colaboração para integrar principalmente aplicações móveis de *groupware*. O problema alvo foi o intercâmbio de serviços entre os vários parceiros de negócios, mas não implementa o gerenciamento de informações relativas a grupos e usuários comuns entre os vários sistemas, bem como administração dos recursos e permissões.

Outro problema comum no cenário de integração refere-se ao gerenciamento distribuído de recursos e informações. Algumas pesquisas relatam os esforços em se prover tal gerenciamento. Em [Ma et al., 2003] é descrito um sistema *groupware* P2P com arquitetura distribuída de gerenciamento. O gerenciamento é totalmente descentralizado. As partes comunicam-se entre si, requisitando e replicando informações. É descrita a desvantagem de um atraso considerável (entre 1 e 10 minutos) para troca de informações relacionadas aos grupos (gerenciamento de usuários e grupos). A razão para esse atraso é devido aos mecanismos de publicação, distribuição e busca usados no JXTA [JXTA Community, 2005], um *framework* que disponibiliza um conjunto de funções para auxiliar as partes a procurarem e contatarem umas às outras e também a troca de mensagens de forma direta. Diferentemente do WS4CSCW, nessa arquitetura não há nenhum tipo de servidor central para auxílio ao gerenciamento, já que é uma arquitetura baseada no gerenciamento distribuído, conforme discutido na Seção 2.4.2.

Em [Younas and Iqbal, 2003] os autores afirmam que CEAs (*Collaborative Editing Applications*) podem ser desenvolvidas usando sistemas de edição colaborativa geograficamente distribuídos, os quais podem fornecer serviços especializados e baratos. Avaliam

a possibilidade de desenvolver CEAs usando diferentes serviços de vários sistemas heterogêneos e autônomos. Além disso, apresentam uma abordagem baseada em transações para garantir que um grupo de usuários possa editar documentos simultaneamente e remotamente. Avaliam ainda, que a interação e cooperação de serviços podem ser temporários. Por exemplo, um sistema pode ser usado para o projeto de uma interface de usuário de uma CEA, mas pode não ser mais usado por outras CEAs se um serviço alternativo com melhor qualidade ou mais barato é oferecido. A abordagem proposta é baseada em transações, que garantem a consistência dos dados acessados e modificados remotamente por múltiplos usuários. Transações usam mecanismos como o ACID (*Atomic, Consistent, Isolated, Durable*) para assegurar a consistência dos dados acessados.

Nesse trabalho também é descrito um exemplo de integração de componentes distribuídos para uma aplicação de edição colaborativa. Por exemplo, um usuário projeta a interface gráfica, enquanto outro implementa a lógica de programação e outro desenvolve um componente para a manipulação de dados. Cada componente pode ser implementado em uma linguagem diferente, e um programador nem precisa saber qual a plataforma ou linguagem que outro programador está usando, uma característica dos *Web Services*.

Em [Li and Li, 2002] é proposta uma abordagem visando investigar os aspectos de interoperabilidade das aplicações colaborativas mono-usuário. Essa abordagem incorpora a técnica da 'caixa preta' para manter a heterogeneidade e autonomia das aplicações envolvidas. Nenhuma modificação é permitida nas aplicações com o objetivo de manter a autonomia.

Em [Dewan and Sharma, 1999] são estudados os casos de endereçamento de aplicações colaborativas heterogêneas. Através da abordagem que integra mecanismos de controle com sistemas de travamento ou bloqueio das aplicações envolvidas. Essa abordagem baseia-se no mecanismo de controle de concorrência. É uma abordagem limitada e não disponibiliza suporte para múltiplos usuários.

Em [Mock et al., 1997] os autores propõem um mecanismo de controle de concorrência para aplicações cooperativas *Web*. No mecanismo proposto, a cooperação é obtida através da introdução de travamentos, os quais são baseados na hipótese de que os usuários são colaborativos e não competitivos. Os usuários solicitam uns aos outros a liberação do travamento quando necessário.

Em [Mehra et al., 2004] é descrita uma abordagem para a construção de aplicações

de edição colaborativa usando *Web Services*. Os autores adicionaram a funcionalidade de edição colaborativa como um componente *plug-in* para o *Pounamu Meta Tool* [Stoeckle et al., 2003], uma ferramenta CASE (*Computer-Aided Software Engineering*). Essa ferramenta foi estendida para prover funcionalidades de edição colaborativa usando *Web Services*. Os autores avaliam as possibilidades de dois usuários trabalharem concorrentemente no mesmo documento (sincronamente) ou em cópias do mesmo documento (assincronamente), que depois podem ser combinadas.

Outro trabalho relacionado é o *Context Kernel*, um *Web Service* desenvolvido com o objetivo de disponibilizar armazenamento e recuperação de informações de contexto, permitindo que aplicações ubíquas possam tirar proveito de informações de contexto tais como personalização, navegação, histórico e consulta de informações [Arruda, 2002].

Uma extensão do *Context Kernel* permitiu a disponibilização de informações de contexto para aplicações CSCW [Bulcão Neto et al., 2004]. Essa extensão permite a atribuição de usuários a grupos, mas não considera o gerenciamento de recursos, permissões e papéis de usuários e grupos.

Com o objetivo de permitir a integração de sistemas CSCW evitando a manipulação de suas características de baixo nível, foi proposta a LEICA - *Loosely-coupled Environment for Integrating Collaborative Applications* [Gomes et al., 2005]. Baseada em *Web Services* e sistema de notificação de eventos, diferentes aplicações colaborativas podem interoperar através da troca de informações dentro do contexto de uma “*global SuperSession*”, uma sessão global que armazena algumas informações referentes às aplicações. A LEICA utiliza uma abordagem de integração fracamente acoplada, permitindo que, após integradas, as aplicações mantenham sua autonomia.

A definição de políticas de colaboração controla as interações entre as aplicações integradas, isto é, como a atividade de colaboração suportada por uma aplicação será afetada pela informação recebida de outras aplicações. Na prática, essas aplicações interagem através da notificação de eventos que podem comandar a execução de ações específicas em algumas dessas aplicações.

O grau de interação entre as aplicações integradas depende da natureza dos eventos que elas são capazes de trocar, e que ações são capazes de executar. Três principais casos podem ser considerados quando aplicações são integradas: 1 - Aplicações *open source*. 2 - aplicações baseadas em API. 3 - aplicações sem qualquer API. A integração de aplicações

open source pode atingir o mais alto grau de interação, desde que qualquer evento ou ação interna possa ser exportado/executado. Isso pode, no entanto, implicar grandes esforços de desenvolvimento. A integração de aplicações baseadas em API é direta, e a interação é limitada à API fornecida.

A abordagem de integração implementada pela LEICA é principalmente guiada pelo segundo caso, de aplicações baseadas em API. A LEICA aplica *Web Services* somente no mecanismo inicial para registrar as novas aplicações a serem integradas, ajustar e iniciar as sessões colaborativas. Assim, uma diferente infra-estrutura é usada para implementar o sistema de notificação de eventos responsável por interconectar as aplicações colaborativas durante a execução de uma sessão colaborativa integrada. As aplicações registradas podem interagir com a LEICA através do *Session Configuration Service*, que é utilizado para configurar as *SuperSessions* globais.

Uma *SuperSession* armazena as atividades de colaboração e é composta por *specificSessions*, que são sessões convencionais definidas dentro do contexto de uma única aplicação, por exemplo, sessão de videoconferência ou sessão de lousa digital. Durante o processo de configuração, o *Session Configuration Service* dinamicamente contacta cada aplicação para requisitar: (i) que dados específicos são requeridos para criar as *specificSessions* e (ii) que tipos de eventos ela pode notificar e que requisições de ações ela pode receber. Essa segunda informação será usada durante o processo de definição das políticas de colaboração.

Políticas de colaboração são um conjunto de regras seguindo um modelo de condição e ação. Essas regras definem como as aplicações podem reagir quando eventos disparados por outras aplicações são notificados.

Depois que as sessões são configuradas e iniciadas, *Web Services* não são mais usados. Conforme o progresso da atividade colaborativa, as aplicações trocam notificação de eventos no modelo P2P.

LEICA fornece um mecanismo de intercâmbio de funcionalidades entre as aplicações através da notificação de eventos e disparo de ações, mas não se preocupa com as informações de usuários, grupos, recursos e permissões. As próprias aplicações envolvidas na integração devem tratar esses problemas, o que poderia causar redundância e inconsistência de usuários, grupos e permissões. Uma alternativa com maiores possibilidades de usos seria o uso da LEICA para intercâmbio de funcionalidades e do WS4CSCW para

o gerenciamento de grupos, usuários, recursos e permissões.

4.7 Cenário de Integração

4.7.1 Considerações Iniciais

Como estudo de caso do WS4CSCW foram propostos dois cenários de integração. O primeiro refere-se à integração das aplicações *iClass* e *No Risk Planning*. Com a finalidade de validação, essa seção descreve os aplicativos envolvidos, bem como o procedimento realizado para a integração e uso do WS4CSCW. O segundo cenário de uso refere-se ao projeto SAFE [Fortes et al., 2004b], que tem como objetivo o desenvolvimento de um *framework* para suportar o processo de desenvolvimento de *software* livre através da integração de diversas ferramentas que apóiam a tarefa de desenvolvimento de *software* livre.

4.7.2 Cenário 1: iClass e No Risk Planning

O *iClass* [Cattelan et al., 2002] é um sistema de computação ubíqua que oferece um suporte efetivo à captura de experiências ao vivo, tanto de aulas como de reuniões informais, disponibilizando um acesso posterior ao material capturado, na forma de documentos estruturados gerados automaticamente.

Para a captura de uma experiência, o sistema utiliza uma sala de aula equipada com microfones, câmeras de vídeo, projetores e uma lousa eletrônica sensível ao toque que, conectada a um computador, permite armazenar as interações do usuário com a tela.

Para prover acesso ao material capturado, um conjunto de aplicações cliente-servidor é responsável pela integração e sincronização dos vários fluxos de informação capturados. Após a aula, as informações são armazenadas em uma base de dados e, a partir da mesma, *scripts* geram hiperdocumentos que podem ser visualizados em um *web browser*. Cada aula é gravada como um único fluxo de áudio ou vídeo, e a interface de acesso disponibiliza diferentes formas de indexação das informações gravadas. Por exemplo, ao ver um slide, o estudante tem a possibilidade de clicar em uma anotação e visualizar o trecho de áudio ou vídeo capturado no momento em que tal anotação foi feita.

A outra aplicação utilizada na integração é a *No Risk Planning*

[Freire and Fortes, 2004], uma agenda compartilhada por grupos de pessoas de modo que os usuários possam utilizá-la de qualquer lugar com um computador conectado à Internet [Ribeiro et al., 2002]. Inicialmente a agenda *No Risk Planning* foi desenvolvida em um projeto de graduação [Moura, 2001]. O processo de engenharia, manutenção e evolução do sistema foi tema de uma Dissertação de mestrado [Ribeiro, 2003].

Considerando-se os serviços e funcionalidades que uma agenda para grupos deve suportar, foi realizado um estudo comparativo entre a ferramenta *No Risk Planning* e outras ferramentas atualmente disponíveis ou em desenvolvimento. Esse estudo engloba tanto aspectos funcionais, tais como características, funcionalidades e particularidades providas, bem como aspectos técnicos, como linguagens e recursos usados na implementação. O estudo descreve ainda as vantagens e desvantagens de cada aplicativo analisado [Vieira, 2003]. Além desta análise comparativa, o processo de desenvolvimento da agenda *No Risk Planning* foi analisado e reportado em [Fortes et al., 2004a].

Além do agendamento de compromissos para grupos, a agenda disponibiliza ainda o compartilhamento de arquivos, bate-papo, visualização das informações do grupo, dos membros e de suas informações pessoais, exibição das agendas de todos os membros que fazem parte do grupo, permitindo que todos possam fazer comparações entre seus horários e os horários dos demais membros e exibição de avisos.

O objetivo da integração foi de permitir que durante a exibição e captura de uma aula através do *iClass*, o professor ou apresentador possa adicionar um compromisso na agenda *No Risk Planning*. Na Figura 4.6 é exibida a interface do *iClass* com um botão adicionado para acionar a funcionalidade da agenda, à esquerda. Quando um usuário realiza um login na ferramenta *iClass*, antes de iniciar a apresentação, é realizada uma consulta ao WS4CSCW sobre a permissão do usuário para o sistema de captura daquela aula e para a funcionalidade de interação com a agenda (Figura 4.7 (a)). Caso o usuário não tenha permissão de uso da ferramenta, não é iniciada a apresentação. Caso o usuário tenha permissão de uso da ferramenta, mas não de acesso à agenda, o botão é desabilitado.

Supondo que o usuário tenha permissão de interação com a agenda, quando o botão que aciona a funcionalidade da agenda é clicado, é disponibilizada a interface exibida na Figura 4.8. Através desta interface, o usuário pode definir a data, tipo, horário inicial e final, descrição, URL e grupo ao qual se refere o compromisso.

Ao definir os dados e clicar em inserir, primeiramente é realizada uma consulta ao

Figura 4.6: Interface *iClass* com o botão que aciona o *Web Service*

Figura 4.7: Cenário de uso: Integração *iClass* - *No Risk Planning*

WS4CSCW sobre a permissão do usuário para a inserção de compromissos para este grupo (Figura 4.7 (a)). Se o usuário for um aluno apresentando um trabalho, e não o professor, por exemplo, ele não tem permissão para inserir compromissos na agenda do grupo ou da disciplina. Caso tenha permissão, a requisição é então encaminhada ao *Web Service* da agenda, que realiza a inserção do compromisso, conforme exibido na Figura

Day:	Month:	Year:	Beg. Time:	End. Time:	Type:
22	June	2005	13:00	14:00	Common
Description:			URL:		
<input type="text"/>			<input type="text"/>		
☒ Group:			<input type="text"/>		
			<input type="button" value="Insert"/> - <input type="button" value="Cancel"/>		

Figura 4.8: Interface requisição de agendamento de compromissos

4.7(b).

O *Web Service* da agenda também faz uso do WS4CSCW. Quando recebe uma requisição de inserção de compromissos para grupo, é realizada uma consulta ao WS4CSCW sobre quais usuários pertencem àquele grupo (Figura 4.7 (c)). A partir desta informação, é inserido o compromisso nas agendas dos membros do grupo. As informações referentes às consultas descritas já foram anteriormente cadastradas, como usuários, grupos, recursos, quais usuários pertencem a cada grupo, permissões a recursos, etc.

Caso não seja necessário, ou permitido, o usuário poderia inserir um compromisso apenas na sua agenda, e não na do grupo. Ou, ainda, acrescentar um compromisso na agenda da disciplina, caso o usuário seja um monitor da disciplina e possua essa permissão.

Depois de capturadas as aulas e apresentações, o *iClass* disponibiliza acesso a esse material. Uma possível extensão de funcionalidades seria disponibilizar o acesso de forma personalizada, permitindo que os usuários acessem apenas os materiais referentes a ele (disciplinas que cursa, leciona ou monitora), ou reuniões de grupos dos quais ele faz parte. Essas informações de usuários, grupos e permissões poderiam ser obtidas do WS4CSCW.

4.7.3 Cenário 2: Projeto SAFE

A comunidade de *Open Source Software* (OSS) utiliza um novo paradigma para desenvolvimento de *software*, e as ferramentas freqüentemente utilizadas pelas equipes OSS para coordenar seus trabalhos incluem ferramentas de controle de versão, de comunicação e sistemas de controle de bugs. Entretanto, há uma carência por suporte a *groupware* para coordenar o desenvolvimento de projetos OSS devido ao fato de estas ferramentas não estarem integradas, mas em bases independentes e sem integração no sistema de controle de versões do repositório de códigos [Vieira et al., 2005].

O projeto SAFE (*Software Engineering Available for Everyone*) [Fortes et al., 2004b] tem como objetivo o desenvolvimento de um *framework* para suportar o processo de desenvolvimento OSS através da integração das diversas ferramentas que apóiam a tarefa de desenvolvimento de *software* livre.

Em projetos de desenvolvimento OSS pode-se identificar algumas características que devem ser suportadas por *groupware*. Há uma tarefa em comum e um ambiente compartilhado - criação e documentação de códigos fonte armazenados em um repositório. Os membros do grupo devem poder se comunicar uns com os outros para discutir e definir as linhas de direção do desenvolvimento, reportar acabamentos e manter o fluxo de comunicação ao longo do processo de desenvolvimento. Eles também devem colaborar através do repositório para criar a implementação do *software* e os artefatos de documentação. A coordenação e sincronização dessas tarefas é importante, especialmente relacionada ao papéis desempenhados por cada membro do grupo envolvido na equipe de desenvolvimento.

Para solucionar essas dificuldades, o projeto SAFE tem como objetivo a construção de um *framework* comum para o suporte de atividades de engenharia de *software* em projetos OSS, incluindo suporte das atividades em grupo. Esse *framework* está sendo desenvolvido através da integração de ferramentas de código aberto já existentes para suportar as diversas atividades dentro de um simples ambiente de *groupware*, no qual os usuários possam se comunicar, colaborar e coordenar seus trabalhos durante todo processo de desenvolvimento de *software*.

Dentre as ferramentas que estão sendo integradas nesse projeto, tem-se: o Bugzilla [The Mozilla Organization, 2005] para controle de bugs e coordenação da manutenção do *software*, a *No Risk Planning*, para agendamento e comunicação, conforme descrito na seção 4.7.2, e VersionWeb [Junqueira et al., 2004] para controle de versões em códigos.

Entretanto, essa integração não é simples. Para permitir um suporte a grupos completo, através de várias ferramentas independentes com bases de dados independentes e diferentes gerenciamentos de usuários e modelos de suporte a grupos, faz-se necessário um repositório comum e um modelo de grupos uniforme que deve ser usado por todas as aplicações integradas no *framework*, evitando inconsistências e redundâncias. Então, as ferramentas devem se comunicar e tornarem-se cientes de quais papéis, grupos e permissões cada usuário tem no contexto de determinado projeto de desenvolvimento.

Através do WS4CSCW as aplicações envolvidas podem se registrar e, através de

chamadas a *Web Services*, podem interagir com o WS4CSCW e fazer requisições relativas às informações de usuários, grupos, papéis, permissões e recursos do projeto OSS, evitando que cada uma das aplicações necessitem fazer esse gerenciamento, o que poderia causar inconsistências ou redundâncias. As funcionalidades de papéis podem ser muito utilizadas nessa integração. Por exemplo, um novo membro de um grupo de desenvolvimento pode herdar todas as permissões previamente atribuídas ao papel *developer* e um novo grupo de analistas pode herdar todas as permissões associadas ao papel de grupo *analysts*.

Considerando a integração de ferramentas para o suporte para grupos OSS no *framework* SAFE, WS4CSCW é de grande auxílio como um serviço que fornece um modelo de gerenciamento de grupos e usuários consistente entre as diferentes ferramentas envolvidas no contexto de integração. O *framework* SAFE também se beneficia dos recursos de gerenciamento de permissões e de papéis. Administradores de projetos podem definir os direitos de acesso permitindo ou negando a um grupo ou usuário o acesso a algum particular recurso ou ferramenta no projeto. Um protótipo da interface do *framework* SAFE, com acesso às ferramentas integradas, é ilustrado na Figura 4.9.

Figura 4.9: Interface do *framework* SAFE

O uso do WS4CSCW torna possível adicionar funcionalidades às aplicações CSCW e permite que elas trabalhem em conjunto sem que seja necessário mudanças custosas

nas aplicações envolvidas. A única alteração necessária é incluir uma chamada de *Web Service* em cada ponto onde as aplicações devem usar um serviço disponibilizado pelo *Web Service*, tal como a criação de um grupo de usuários ou a verificação dos direitos de acesso para um recurso, por exemplo. Isso torna possível um processo de integração mais rápido e evita a redundância e inconsistência de dados entre diferentes repositórios das ferramentas a serem integradas.

4.8 Considerações Finais

Neste capítulo foram descritos os conceitos do modelo de dados usado no projeto e implementação do WS4CSCW e os possíveis cenários de uso do WS4CSCW, caracterizando as vantagens fornecidas para a integração de aplicações CSCW. Também foram explicados todos os serviços disponibilizados, suas entradas e saídas. Além disso, foram discutidos os trabalhos relacionados e suas particularidades frente ao WS4CSCW. Com o objetivo de avaliar as potencialidades do WS4CSCW, foram descritos dois cenários de integração propostos para a validação, envolvendo a integração de aplicações CSCW.

Uma desvantagem da abordagem de integração de aplicações via *Web Services* refere-se ao esforço no desenvolvimento ao se acrescentar chamadas a *Web Services* em aplicações já existentes. Dependendo da arquitetura das aplicações existentes, essa integração pode implicar esforços de desenvolvimento ou até mesmo o reprojeto da aplicação.

Observa-se, no entanto, que as tecnologias para *Web Services* têm obtido adeptos de diversas áreas e se apresenta como uma boa prática, conforme os padrões W3C. Acreditamos, então, que os eventuais esforços necessários para a interação mencionados tendem a ser reduzidos com a utilização das funcionalidades fornecidas pelo WS4CSCW.

Conclusões

O desenvolvimento de trabalhos em grupo tornou-se uma tarefa cada vez mais comum nos dias atuais. Um dos principais objetivos do trabalho em grupo é incentivar o aspecto de ser cooperativo, o qual refere-se à forma de solucionar problemas através da divisão destes em partes que possam ser resolvidas por grupos, com o intuito de diminuir o tempo e os custos envolvidos, bem como reforçar o valor de colaboração entre os participantes.

A utilização de ferramentas computacionais é fundamental para que haja uma efetiva comunicação entre os membros dos grupos e possibilite ganho de produtividade. Nesse cenário, tecnologias baseadas em computação, desde que aplicadas de forma planejada e com a utilização dos recursos adequados, tornam-se um valioso recurso prático e indispensável para aumentar o desempenho dos indivíduos envolvidos na realização de tarefas em grupo.

As tarefas colaborativas, em geral, apresentam novas funcionalidades ao usuário quando várias aplicações trabalham em conjunto, melhorando os potenciais usos dos sistemas de *software*, ao invés de um única aplicação complexa que trabalha de forma isolada. Assim, alguns sistemas CSCW tentam combinar diferentes funcionalidades para suportar colaboração em um simples ambiente. Com isso, esforços na integração de aplicações e componentes de *software* têm sido notados. No entanto, a tarefa de integração de aplicações não é simples, pois quando várias aplicações CSCW são integradas, é possível que informações fiquem redundantes ou mesmo inconsistentes, ocasionando problemas consideráveis.

Para a integração de aplicações CSCW, neste trabalho foi pesquisado um mecanismo de suporte ao gerenciamento de grupos e usuários, de forma a facilitar e organizar o trabalho cooperativo entre as aplicações envolvidas, bem como o gerenciamento dos recursos da memória corporativa e as permissões atribuídas a cada usuário ou grupo, permitindo que o trabalho cooperativo tenha uma coordenação efetiva e consistente. Como isso, foi desenvolvido o WS4CSCW (*Web Services for Computer Supported Cooperative Work*),

um *Web Service* que tem como objetivo facilitar a integração de ferramentas e sistemas CSCW. O WS4CSCW permite o gerenciamento de usuários, grupos, recursos, permissões e papéis de usuários e grupos, disponibilizando funcionalidades para aplicações voltadas ao trabalho em grupo, bem como visa facilitar a integração entre essas aplicações, permitindo um co-funcionamento mais direto, consistente e coeso.

O WS4CSCW apresenta ainda o objetivo de simplificar o desenvolvimento de aplicações CSCW, bem como possibilitar a integração das mesmas. Mesmo trabalhando de forma isolada, sem comunicação com outras aplicações, uma aplicação CSCW também pode usufruir dos serviços WS4CSCW, eliminando da fase de desenvolvimento abstrações relativas ao gerenciamento de usuários, grupos, recursos e permissões. Assim, são priorizados os esforços do desenvolvimento nas particularidades e funcionalidades da nova aplicação e aproveitados os serviços do WS4CSCW.

Resumidamente, as principais contribuições obtidas com o desenvolvimento da pesquisa realizada neste trabalho foram:

- um mecanismo para facilitar a integração de aplicações CSCW;
- um mecanismo para facilitar o projeto de aplicações CSCW, aproveitando as funcionalidades já implementadas;
- definição do serviço para gerenciamento de usuários, grupos e recursos;
- definição do serviço de gerenciamento de permissões a recursos;
- definição do serviço de gerenciamento de papéis de usuários e de grupos;
- implementação de *Web Services* relativos aos serviços definidos e disponibilização de seu código para possível inclusão de novos serviços relativos a aplicações para grupos (bastando apenas acrescentar novas funções ao código do WS4CSCW);
- validação dos *Web Services* desenvolvidos, por meio de integração e comunicação entre aplicações CSCW já existentes.

5.1 Publicações

As publicações obtidas no contexto deste trabalho foram as seguintes:

- **Victor H. Vieira**, Daniel G. Sante, André P. Freire and Renata P. M. Fortes (2005). A Web Service for CSCW applications. In *WebMedia '05: Proceedings of the 11th Brazilian Symposium on Multimedia and the web*, pages 1–3, Poços de Caldas - MG, Brazil. ACM Press, New York, NY, USA.
- **Victor H. Vieira** and Renata P. M. Fortes (2004). Integração de Aplicações Web: Intercâmbio de Funcionalidades e Informações de Contexto usando Web Services. In *Proceedings of 2nd IW3C2 Latin American Web Congress*, volume 2, pages 217–221. Ribeirão Preto - SP, Brasil. SBC - Brazilian Computer Society.
- Renata P. M. Fortes, **Victor H. Vieira**, Andre P. Freire and Debora Paiva (2004). An Academic Web-based Agenda and Its Engineering Process. In *VII Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes Software (IDEAS2004)*, pages 151–156. Arequipa, Peru. Peruvian Computer Society and National University of San Agustin. ISBN 9972-9876-1-2.
- **Victor H. Vieira** and Renata P. M. Fortes (2005). Documentação do software WS4CSCW (Web services for CSCW). Relatório Técnico 263, Instituto de Ciências Matemáticas e de Computação – USP. 52 pages.

5.2 Trabalhos Futuros

Diversos desafios puderam também ser observados, como resultado dos estudos empreendidos durante a realização desta pesquisa. Assim, por exemplo, uma das primeiras metas a serem continuadas a partir do desenvolvimento do WS4CSCW, é a finalização da integração das ferramentas no *framework* SAFE e subsequente condução de testes de completitude que garantam a qualidade dos serviços oferecidos, bem como uma avaliação real do desempenho dos serviços disponibilizados pelo WS4CSCW.

Uma alternativa que parece ser bem promissora é a de prover uma interface gráfica para acessar os serviços e permitir o gerenciamento por parte de um administrador. Por exemplo, para cadastrar e alterar permissões a recursos, seria mais amigável um administrador fazer isso através de uma interface gráfica do que com chamadas ao *Web Service* no código das aplicações envolvidas. Para tanto, um projeto de pesquisa em nível de iniciação científica está sendo elaborado.

Finalmente, como meta de mais longo prazo, espera-se que uma pesquisa que avalie as possibilidades de inclusão e implementação de novos serviços no WS4CSCW, como por exemplo, notificação entre aplicações, troca de mensagens entre usuários e grupos, bloqueio e gerenciamento de acesso simultâneo de recursos compartilhados, seja conduzida.

Especificação WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://localhost:8080/axis/ws4cscw.jws" xmlns="http://
schemas.xmlsoap.org/wsdl/" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://localhost:8080/axis/ws4cscw.jws" xmlns:intf="http://localhost:8080
/axis/ws4cscw.jws" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsdlsoap="http://schemas.xmlsoap
.org/wsdl/soap/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema
      targetNamespace="http://localhost:8080/axis/ws4cscw.jws" xmlns="http://www.w3.org
/2001/XMLSchema">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/">
        <complexType name="ArrayOf_xsd_string">
          <complexContent>
            <restriction base="
soapenc:Array">
              <attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string []"/>
            </restriction>
          </complexContent>
        </complexType>
      </schema>
    </wsdl:types>
  </wsdl:types>
  <wsdl:message name="atribUserPermissiontoUserRoleResponse">
    <wsdl:part name="atribUserPermissiontoUserRoleReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="removeUserPermissionfromUserRoleResponse">
    <wsdl:part name="removeUserPermissionfromUserRoleReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="getUsersinResourcePermissionsResponse">
    <wsdl:part name="getUsersinResourcePermissionsReturn" type="impl:ArrayOf_xsd_string"/
    >
  </wsdl:message>
  <wsdl:message name="addUsertogroupResponse">
    <wsdl:part name="addUsertogroupReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="addGroupRoletogroupRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="groupName" type="xsd:string"/>
    <wsdl:part name="groupRoleName" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="atribUserPermissiontoUserResponse">
    <wsdl:part name="atribUserPermissiontoUserReturn" type="xsd:string"/>
  </wsdl:message>
  <wsdl:message name="getUsersfromResourcePermissionsValueRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
  </wsdl:message>
</wsdl:definitions>
```

```

    <wsdl:part name="value" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeUserResponse">
    <wsdl:part name="removeUserReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeUserPermissionfromUserRoleRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userRoleName" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
    <wsdl:part name="value" type="xsd:string"/>
    <wsdl:part name="priorityovergroup" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addGroupResponse">
    <wsdl:part name="addGroupReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addGroupRoleResponse">
    <wsdl:part name="addGroupRoleReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addUserRoletoUserResponse">
    <wsdl:part name="addUserRoletoUserReturn" type="xsd:string"/>
</wsdl:message>
50 <wsdl:message name="getUsersingroupRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="groupName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="createContextResponse">
    <wsdl:part name="createContextReturn" type="impl:ArrayOf_xsd_string"/>
</wsdl:message>
<wsdl:message name="getGroupsfromUserRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getUserHasPermissionValueResponse">
    <wsdl:part name="getUserHasPermissionValueReturn" type="xsd:int"/>
</wsdl:message>
<wsdl:message name="atribGroupPermissiontoGroupRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="groupName" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
    <wsdl:part name="value" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
    <wsdl:part name="type" type="xsd:string"/>
    <wsdl:part name="rank" type="xsd:int"/>
</wsdl:message>
<wsdl:message name="removeGroupRolefromGroupResponse">

```

```

        <wsdl:part name="removeGroupRolefromGroupReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="atribUserPermissiontoUserRequest">
        <wsdl:part name="appPrivateID" type="xsd:string"/>
        <wsdl:part name="contextPrivateId" type="xsd:string"/>
        <wsdl:part name="userName" type="xsd:string"/>
        <wsdl:part name="resourceName" type="xsd:string"/>
        <wsdl:part name="value" type="xsd:string"/>
        <wsdl:part name="description" type="xsd:string"/>
        <wsdl:part name="priorityovergroup" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="addUserRoleRequest">
        <wsdl:part name="appPrivateID" type="xsd:string"/>
        <wsdl:part name="contextPrivateId" type="xsd:string"/>
        <wsdl:part name="name" type="xsd:string"/>
        <wsdl:part name="description" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="getUsersfromResourcePermissionsValueResponse">
        <wsdl:part name="getUsersfromResourcePermissionsValueReturn" type="
            impl:ArrayOf_xsd_string"/>
    </wsdl:message>
    <wsdl:message name="addUserRoleResponse">
        <wsdl:part name="addUserRoleReturn" type="xsd:string"/>
    </wsdl:message>
100 <wsdl:message name="removeUserPermissionfromUserRequest">
        <wsdl:part name="appPrivateID" type="xsd:string"/>
        <wsdl:part name="contextPrivateId" type="xsd:string"/>
        <wsdl:part name="userName" type="xsd:string"/>
        <wsdl:part name="resourceName" type="xsd:string"/>
        <wsdl:part name="value" type="xsd:string"/>
        <wsdl:part name="priorityovergroup" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeUserRolefromUserResponse">
        <wsdl:part name="removeUserRolefromUserReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeGroupRequest">
        <wsdl:part name="appPrivateID" type="xsd:string"/>
        <wsdl:part name="contextPrivateId" type="xsd:string"/>
        <wsdl:part name="name" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="addAppResponse">
        <wsdl:part name="addAppReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="getUserPermissionfromUserResponse">
        <wsdl:part name="getUserPermissionfromUserReturn" type="impl:ArrayOf_xsd_string"/>
    </wsdl:message>
    <wsdl:message name="removeUserfromGroupRequest">
        <wsdl:part name="appPrivateID" type="xsd:string"/>
        <wsdl:part name="contextPrivateId" type="xsd:string"/>

```

```

    <wsdl:part name="userName" type="xsd:string"/>
    <wsdl:part name="groupName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="atribGroupPermissiontoGroupResponse">
    <wsdl:part name="atribGroupPermissiontoGroupReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="atribGroupPermissiontoGroupRoleResponse">
    <wsdl:part name="atribGroupPermissiontoGroupRoleReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeUserRolefromUserRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userName" type="xsd:string"/>
    <wsdl:part name="userRoleName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addGroupRoletoGroupResponse">
    <wsdl:part name="addGroupRoletoGroupReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="atribUserPermissiontoUserRoleRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userRoleName" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
    <wsdl:part name="value" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
    <wsdl:part name="priorityovergroup" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getUsersinGroupResponse">
    <wsdl:part name="getUsersinGroupReturn" type="impl:ArrayOf_xsd_string"/>
</wsdl:message>
<wsdl:message name="addUserRoletoUserRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userName" type="xsd:string"/>
    <wsdl:part name="userRoleName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addResourceResponse">
    <wsdl:part name="addResourceReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addGroupRoleRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="name" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getGroupsfromUserResponse">
    <wsdl:part name="getGroupsfromUserReturn" type="impl:ArrayOf_xsd_string"/>
</wsdl:message>
<wsdl:message name="getUsersAndPermissionsfromResourceResponse">

```



```

        <wsdl:part name="getUsersAndPermissionsfromResourceReturn" type="
            impl:ArrayOf_xsd_string"/>
    </wsdl:message>
    <wsdl:message name="addUserResponse">
        <wsdl:part name="addUserReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeGroupResponse">
        <wsdl:part name="removeGroupReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeGroupPermissionfromGroupRoleRequest">
        <wsdl:part name="appPrivateld" type="xsd:string"/>
        <wsdl:part name="contextPrivateld" type="xsd:string"/>
        <wsdl:part name="groupRoleName" type="xsd:string"/>
        <wsdl:part name="resourceName" type="xsd:string"/>
        <wsdl:part name="value" type="xsd:string"/>
        <wsdl:part name="type" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeGroupPermissionfromGroupResponse">
        <wsdl:part name="removeGroupPermissionfromGroupReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeUserRequest">
        <wsdl:part name="appPrivateld" type="xsd:string"/>
        <wsdl:part name="contextPrivateld" type="xsd:string"/>
        <wsdl:part name="name" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeUserfromGroupResponse">
        <wsdl:part name="removeUserfromGroupReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="addGroupRequest">
        <wsdl:part name="appPrivateld" type="xsd:string"/>
        <wsdl:part name="contextPrivateld" type="xsd:string"/>
        <wsdl:part name="name" type="xsd:string"/>
        <wsdl:part name="category" type="xsd:string"/>
        <wsdl:part name="sigla" type="xsd:string"/>
        <wsdl:part name="description" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeGroupRolefromGroupRequest">
        <wsdl:part name="appPrivateld" type="xsd:string"/>
        <wsdl:part name="contextld" type="xsd:string"/>
        <wsdl:part name="groupName" type="xsd:string"/>
        <wsdl:part name="groupRoleName" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="removeResourceResponse">
        <wsdl:part name="removeResourceReturn" type="xsd:string"/>
    </wsdl:message>
    <wsdl:message name="getUserHasPermissionValueRequest">
        <wsdl:part name="appPrivateld" type="xsd:string"/>
        <wsdl:part name="contextPrivateld" type="xsd:string"/>
        <wsdl:part name="userName" type="xsd:string"/>

```

```

    <wsdl:part name="resourceName" type="xsd:string"/>
    <wsdl:part name="value" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getUsersAndPermissionsfromResourceRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addResourceRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="name" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
    <wsdl:part name="bool" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addUserRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="name" type="xsd:string"/>
    <wsdl:part name="nick" type="xsd:string"/>
    <wsdl:part name="email" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getUserPermissionfromUserRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="userName" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeGroupPermissionfromGroupRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="groupName" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>
    <wsdl:part name="value" type="xsd:string"/>
    <wsdl:part name="type" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addAppRequest">
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="app_name" type="xsd:string"/>
    <wsdl:part name="description" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeUserPermissionfromUserResponse">
    <wsdl:part name="removeUserPermissionfromUserReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="getUsersinResourcePermissionsRequest">
    <wsdl:part name="appPrivateld" type="xsd:string"/>
    <wsdl:part name="contextPrivateld" type="xsd:string"/>
    <wsdl:part name="resourceName" type="xsd:string"/>

```

```

</wsdl:message>
<wsdl:message name="createContextRequest">
  <wsdl:part name="context_name" type="xsd:string"/>
  <wsdl:part name="description" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="addUsertoGroupRequest">
  <wsdl:part name="appPrivateID" type="xsd:string"/>
  <wsdl:part name="contextPrivateId" type="xsd:string"/>
  <wsdl:part name="userName" type="xsd:string"/>
  <wsdl:part name="groupName" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="atribGroupPermissiontoGroupRoleRequest">
  <wsdl:part name="appPrivateID" type="xsd:string"/>
  <wsdl:part name="contextPrivateId" type="xsd:string"/>
  <wsdl:part name="groupRoleName" type="xsd:string"/>
  <wsdl:part name="resourceName" type="xsd:string"/>
  <wsdl:part name="value" type="xsd:string"/>
  <wsdl:part name="description" type="xsd:string"/>
  <wsdl:part name="type" type="xsd:string"/>
  <wsdl:part name="rank" type="xsd:int"/>
</wsdl:message>
<wsdl:message name="removeGroupPermissionfromGroupRoleResponse">
  <wsdl:part name="removeGroupPermissionfromGroupRoleReturn" type="xsd:string"/>
</wsdl:message>
<wsdl:message name="removeResourceRequest">
  <wsdl:part name="appPrivateID" type="xsd:string"/>
  <wsdl:part name="contextPrivateId" type="xsd:string"/>
  <wsdl:part name="name" type="xsd:string"/>
</wsdl:message>
300 <wsdl:portType name="ws4cscw">
  <wsdl:operation name="addResource" parameterOrder="appPrivateID contextPrivateId name
    description bool">
    <wsdl:input message="impl:addResourceRequest" name="addResourceRequest"/>
    <wsdl:output message="impl:addResourceResponse" name="addResourceResponse"/>
  </wsdl:operation>
  <wsdl:operation name="removeResource" parameterOrder="appPrivateID contextPrivateId
    name">
    <wsdl:input message="impl:removeResourceRequest" name="removeResourceRequest"/>
    <wsdl:output message="impl:removeResourceResponse" name="removeResourceResponse"/>
  </wsdl:operation>
  <wsdl:operation name="removeGroup" parameterOrder="appPrivateID contextPrivateId name
    ">
    <wsdl:input message="impl:removeGroupRequest" name="removeGroupRequest"/>
    <wsdl:output message="impl:removeGroupResponse" name="removeGroupResponse"/>
  </wsdl:operation>
  <wsdl:operation name="removeUser" parameterOrder="appPrivateID contextPrivateId name"
    >
    <wsdl:input message="impl:removeUserRequest" name="removeUserRequest"/>
    <wsdl:output message="impl:removeUserResponse" name="removeUserResponse"/>

```

```

</wsdl:operation>
<wsdl:operation name="addGroup" parameterOrder="appPrivateld contextPrivateld name
    category sigla description">
    <wsdl:input message="impl:addGroupRequest" name="addGroupRequest"/>
    <wsdl:output message="impl:addGroupResponse" name="addGroupResponse"/>
</wsdl:operation>
<wsdl:operation name="addApp" parameterOrder="contextPrivateld app_name description">
    <wsdl:input message="impl:addAppRequest" name="addAppRequest"/>
    <wsdl:output message="impl:addAppResponse" name="addAppResponse"/>
</wsdl:operation>
<wsdl:operation name="addGroupRole" parameterOrder="appPrivateld contextPrivateld
    name description">
    <wsdl:input message="impl:addGroupRoleRequest" name="addGroupRoleRequest"/>
    <wsdl:output message="impl:addGroupRoleResponse" name="addGroupRoleResponse"/>
</wsdl:operation>
<wsdl:operation name="addGroupRoletoGroup" parameterOrder="appPrivateld
    contextPrivateld groupName groupRoleName">
    <wsdl:input message="impl:addGroupRoletoGroupRequest" name="
        addGroupRoletoGroupRequest"/>
    <wsdl:output message="impl:addGroupRoletoGroupResponse" name="
        addGroupRoletoGroupResponse"/>
</wsdl:operation>
<wsdl:operation name="addUser" parameterOrder="appPrivateld contextPrivateld name
    nick email description">
    <wsdl:input message="impl:addUserRequest" name="addUserRequest"/>
    <wsdl:output message="impl:addUserResponse" name="addUserResponse"/>
</wsdl:operation>
<wsdl:operation name="addUserRole" parameterOrder="appPrivateld contextPrivateld name
    description">
    <wsdl:input message="impl:addUserRoleRequest" name="addUserRoleRequest"/>
    <wsdl:output message="impl:addUserRoleResponse" name="addUserRoleResponse"/>
</wsdl:operation>
<wsdl:operation name="addUserRoletoUser" parameterOrder="appPrivateld
    contextPrivateld userName userRoleName">
    <wsdl:input message="impl:addUserRoletoUserRequest" name="addUserRoletoUserRequest"
        />
    <wsdl:output message="impl:addUserRoletoUserResponse" name="
        addUserRoletoUserResponse"/>
</wsdl:operation>
<wsdl:operation name="addUsertoGroup" parameterOrder="appPrivateld contextPrivateld
    userName groupName">
    <wsdl:input message="impl:addUsertoGroupRequest" name="addUsertoGroupRequest"/>
    <wsdl:output message="impl:addUsertoGroupResponse" name="addUsertoGroupResponse"/>
</wsdl:operation>
<wsdl:operation name="atribGroupPermissiontoGroup" parameterOrder="appPrivateld
    contextPrivateld groupName resourceName value description type rank">
    <wsdl:input message="impl:atribGroupPermissiontoGroupRequest" name="
        atribGroupPermissiontoGroupRequest"/>
    <wsdl:output message="impl:atribGroupPermissiontoGroupResponse" name="

```

```

        atribGroupPermissiontoGroupResponse"/>
</wsdl:operation>
<wsdl:operation name="atribGroupPermissiontoGroupRole" parameterOrder="appPrivateld
    contextPrivateld groupRoleName resourceName value description type rank">
    <wsdl:input message="impl:atribGroupPermissiontoGroupRoleRequest" name="
        atribGroupPermissiontoGroupRoleRequest"/>
    <wsdl:output message="impl:atribGroupPermissiontoGroupRoleResponse" name="
        atribGroupPermissiontoGroupRoleResponse"/>
</wsdl:operation>
<wsdl:operation name="atribUserPermissiontoUser" parameterOrder="appPrivateld
    contextPrivateld userName resourceName value description priorityovergroup">
    <wsdl:input message="impl:atribUserPermissiontoUserRequest" name="
        atribUserPermissiontoUserRequest"/>
    <wsdl:output message="impl:atribUserPermissiontoUserResponse" name="
        atribUserPermissiontoUserResponse"/>
</wsdl:operation>
<wsdl:operation name="atribUserPermissiontoUserRole" parameterOrder="appPrivateld
    contextPrivateld userRoleName resourceName value description priorityovergroup">
    <wsdl:input message="impl:atribUserPermissiontoUserRoleRequest" name="
        atribUserPermissiontoUserRoleRequest"/>
    <wsdl:output message="impl:atribUserPermissiontoUserRoleResponse" name="
        atribUserPermissiontoUserRoleResponse"/>
</wsdl:operation>
<wsdl:operation name="createContext" parameterOrder="context_name description">
    <wsdl:input message="impl:createContextRequest" name="createContextRequest"/>
    <wsdl:output message="impl:createContextResponse" name="createContextResponse"/>
</wsdl:operation>
<wsdl:operation name="getGroupsfromUser" parameterOrder="appPrivateld
    contextPrivateld userName">
    <wsdl:input message="impl:getGroupsfromUserRequest" name="getGroupsfromUserRequest"
        />
    <wsdl:output message="impl:getGroupsfromUserResponse" name="
        getGroupsfromUserResponse"/>
</wsdl:operation>
<wsdl:operation name="getUserHasPermissionValue" parameterOrder="appPrivateld
    contextPrivateld userName resourceName value">
    <wsdl:input message="impl:getUserHasPermissionValueRequest" name="
        getUserHasPermissionValueRequest"/>
    <wsdl:output message="impl:getUserHasPermissionValueResponse" name="
        getUserHasPermissionValueResponse"/>
</wsdl:operation>
<wsdl:operation name="getUserPermissionfromUser" parameterOrder="appPrivateld
    contextPrivateld userName resourceName">
    <wsdl:input message="impl:getUserPermissionfromUserRequest" name="
        getUserPermissionfromUserRequest"/>
    <wsdl:output message="impl:getUserPermissionfromUserResponse" name="
        getUserPermissionfromUserResponse"/>
</wsdl:operation>
<wsdl:operation name="getUsersAndPermissionsfromResource" parameterOrder="

```

```

        appPrivateld contextPrivateld resourceName">
    <wsdl:input message="impl:getUsersAndPermissionsfromResourceRequest" name="
        getUsersAndPermissionsfromResourceRequest"/>
    <wsdl:output message="impl:getUsersAndPermissionsfromResourceResponse" name="
        getUsersAndPermissionsfromResourceResponse"/>
</wsdl:operation>
<wsdl:operation name="getUsersfromResourcePermissionsValue" parameterOrder="
    appPrivateld contextPrivateld resourceName value">
    <wsdl:input message="impl:getUsersfromResourcePermissionsValueRequest" name="
        getUsersfromResourcePermissionsValueRequest"/>
    <wsdl:output message="impl:getUsersfromResourcePermissionsValueResponse" name="
        getUsersfromResourcePermissionsValueResponse"/>
</wsdl:operation>
<wsdl:operation name="getUsersingroup" parameterOrder="appPrivateld contextPrivateld
    groupName">
    <wsdl:input message="impl:getUsersingroupRequest" name="getUsersingroupRequest"/>
    <wsdl:output message="impl:getUsersingroupResponse" name="getUsersingroupResponse"/
    >
</wsdl:operation>
<wsdl:operation name="getUsersinResourcePermissions" parameterOrder="appPrivateld
    contextPrivateld resourceName">
    <wsdl:input message="impl:getUsersinResourcePermissionsRequest" name="
        getUsersinResourcePermissionsRequest"/>
    <wsdl:output message="impl:getUsersinResourcePermissionsResponse" name="
        getUsersinResourcePermissionsResponse"/>
</wsdl:operation>
<wsdl:operation name="removeGroupPermissionfromGroup" parameterOrder="appPrivateld
    contextPrivateld groupName resourceName value type">
    <wsdl:input message="impl:removeGroupPermissionfromGroupRequest" name="
        removeGroupPermissionfromGroupRequest"/>
    <wsdl:output message="impl:removeGroupPermissionfromGroupResponse" name="
        removeGroupPermissionfromGroupResponse"/>
400 </wsdl:operation>
<wsdl:operation name="removeGroupPermissionfromGroupRole" parameterOrder="
    appPrivateld contextPrivateld groupRoleName resourceName value type">
    <wsdl:input message="impl:removeGroupPermissionfromGroupRoleRequest" name="
        removeGroupPermissionfromGroupRoleRequest"/>
    <wsdl:output message="impl:removeGroupPermissionfromGroupRoleResponse" name="
        removeGroupPermissionfromGroupRoleResponse"/>
</wsdl:operation>
<wsdl:operation name="removeGroupRolefromGroup" parameterOrder="appPrivateld
    contextId groupName groupRoleName">
    <wsdl:input message="impl:removeGroupRolefromGroupRequest" name="
        removeGroupRolefromGroupRequest"/>
    <wsdl:output message="impl:removeGroupRolefromGroupResponse" name="
        removeGroupRolefromGroupResponse"/>
</wsdl:operation>
<wsdl:operation name="removeUserPermissionfromUser" parameterOrder="appPrivateld
    contextPrivateld userName resourceName value priorityovergroup">

```

```

        <wsdl:input message="impl:removeUserPermissionfromUserRequest" name="
            removeUserPermissionfromUserRequest" />
        <wsdl:output message="impl:removeUserPermissionfromUserResponse" name="
            removeUserPermissionfromUserResponse" />
    </wsdl:operation>
    <wsdl:operation name="removeUserPermissionfromUserRole" parameterOrder="appPrivateID
        contextPrivateId userRoleName resourceName value priorityovergroup">
        <wsdl:input message="impl:removeUserPermissionfromUserRoleRequest" name="
            removeUserPermissionfromUserRoleRequest" />
        <wsdl:output message="impl:removeUserPermissionfromUserRoleResponse" name="
            removeUserPermissionfromUserRoleResponse" />
    </wsdl:operation>
    <wsdl:operation name="removeUserRolefromUser" parameterOrder="appPrivateID
        contextPrivateId userName userRoleName">
        <wsdl:input message="impl:removeUserRolefromUserRequest" name="
            removeUserRolefromUserRequest" />
        <wsdl:output message="impl:removeUserRolefromUserResponse" name="
            removeUserRolefromUserResponse" />
    </wsdl:operation>
    <wsdl:operation name="removeUserfromGroup" parameterOrder="appPrivateID
        contextPrivateId userName groupName">
        <wsdl:input message="impl:removeUserfromGroupRequest" name="
            removeUserfromGroupRequest" />
        <wsdl:output message="impl:removeUserfromGroupResponse" name="
            removeUserfromGroupResponse" />
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="ws4cscwSoapBinding" type="impl:ws4cscw">
    <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="addResource">
        <wsdlsoap:operation soapAction="" />
        <wsdl:input name="addResourceRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="http://DefaultNamespace" use="encoded" />
        </wsdl:input>
        <wsdl:output name="addResourceResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="removeResource">
        <wsdlsoap:operation soapAction="" />
        <wsdl:input name="removeResourceRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="http://DefaultNamespace" use="encoded" />
        </wsdl:input>
        <wsdl:output name="removeResourceResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
        </wsdl:output>
    </wsdl:operation>

```

```

    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeGroup">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="removeGroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded" />
    </wsdl:input>
    <wsdl:output name="removeGroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeUser">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="removeUserRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded" />
    </wsdl:input>
    <wsdl:output name="removeUserResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addGroup">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="addGroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded" />
    </wsdl:input>
    <wsdl:output name="addGroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addApp">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="addAppRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded" />
    </wsdl:input>
    <wsdl:output name="addAppResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addGroupRole">
    <wsdlsoap:operation soapAction="" />
    <wsdl:input name="addGroupRoleRequest">

```



```

        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addGroupRoleResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addGroupRoletoGroup">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="addGroupRoletoGroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addGroupRoletoGroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addUser">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="addUserRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addUserResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addUserRole">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="addUserRoleRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addUserRoleResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addUserRoletoUser">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="addUserRoletoUserRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addUserRoletoUserResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"

```

```

        namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="addUserToGroup">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="addUserToGroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="addUserToGroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="atribGroupPermissionToGroup">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="atribGroupPermissionToGroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="atribGroupPermissionToGroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="atribGroupPermissionToGroupRole">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="atribGroupPermissionToGroupRoleRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="atribGroupPermissionToGroupRoleResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="atribUserPermissionToUser">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="atribUserPermissionToUserRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="atribUserPermissionToUserResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="atribUserPermissionToUserRole">
    <wsdlsoap:operation soapAction=""/>

```

```

<wsdl:input name="atribUserPermissiontoUserRoleRequest">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://DefaultNamespace" use="encoded"/>
</wsdl:input>
<wsdl:output name="atribUserPermissiontoUserRoleResponse">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="createContext">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="createContextRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded"/>
  </wsdl:input>
  <wsdl:output name="createContextResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getGroupsfromUser">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getGroupsfromUserRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded"/>
  </wsdl:input>
  <wsdl:output name="getGroupsfromUserResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUserHasPermissionValue">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getUserHasPermissionValueRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded"/>
  </wsdl:input>
  <wsdl:output name="getUserHasPermissionValueResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUserPermissionfromUser">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getUserPermissionfromUserRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded"/>
  </wsdl:input>
  <wsdl:output name="getUserPermissionfromUserResponse">

```

```

        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUsersAndPermissionsfromResource">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getUsersAndPermissionsfromResourceRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="getUsersAndPermissionsfromResourceResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUsersfromResourcePermissionsValue">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getUsersfromResourcePermissionsValueRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="getUsersfromResourcePermissionsValueResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUsersingroup">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getUsersingroupRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="getUsersingroupResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getUsersinResourcePermissions">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getUsersinResourcePermissionsRequest">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="getUsersinResourcePermissionsResponse">
        <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeGroupPermissionfromGroup">

```

```

<wsdlsoap:operation soapAction="" />
<wsdl:input name="removeGroupPermissionfromGroupRequest">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://DefaultNamespace" use="encoded" />
</wsdl:input>
<wsdl:output name="removeGroupPermissionfromGroupResponse">
  <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeGroupPermissionfromGroupRole">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="removeGroupPermissionfromGroupRoleRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded" />
  </wsdl:input>
  <wsdl:output name="removeGroupPermissionfromGroupRoleResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeGroupRolefromGroup">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="removeGroupRolefromGroupRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded" />
  </wsdl:input>
  <wsdl:output name="removeGroupRolefromGroupResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeUserPermissionfromUser">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="removeUserPermissionfromUserRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded" />
  </wsdl:input>
  <wsdl:output name="removeUserPermissionfromUserResponse">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded" />
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="removeUserPermissionfromUserRole">
  <wsdlsoap:operation soapAction="" />
  <wsdl:input name="removeUserPermissionfromUserRoleRequest">
    <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      namespace="http://DefaultNamespace" use="encoded" />
  </wsdl:input>

```

```

    <wsdl:output name="removeUserPermissionfromUserRoleResponse">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="removeUserRolefromUser">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="removeUserRolefromUserRequest">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="removeUserRolefromUserResponse">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="removeUserfromGroup">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="removeUserfromGroupRequest">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://DefaultNamespace" use="encoded"/>
    </wsdl:input>
    <wsdl:output name="removeUserfromGroupResponse">
      <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="http://localhost:8080/axis/ws4cscw.jws" use="encoded"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="ws4cscwService">
  <wsdl:port binding="impl:ws4cscwSoapBinding" name="ws4cscw">
    <wsdlsoap:address location="http://localhost:8080/axis/ws4cscw.jws"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

Referências Bibliográficas

- [Appelt, 2001] Appelt, W. (2001). What groupware functionality do users really use? analysis of the usage of the bscw system. In *9th Euromicro Workshop on PDP*. IEEE Computer Society.
- [Arruda, 2002] Arruda, C. R. E. (2002). Context kernel: um web service baseado nas dimensões de informação de contexto. Dissertação de mestrado, USP, São Carlos, SP.
- [Bentley et al., 1997] Bentley, R., Appelt, W., Busbach, U., Hinrichs, E., Kerr, D., Sikkell, K., Trevor, J., and Woetzel, G. (1997). Basic suport for cooperative work on the world wide web. *International Journal of Human-Computer Studies: Special issue on Novel Applications of the WWW*.
- [Borges et al., 1995] Borges, M. R. S., Cavalcanti, M. C. R., and Campos, M. L. M. (1995). Suporte por computador ao trabalho cooperativo. In *XV Congresso da Sociedade Brasileira de Computação*.
- [Borges and Pino, 2001] Borges, M. R. S. and Pino, J. A. (2001). Interfaces for goupware. In *ACM CHI 2001 Development Consortium*. Disponível em <http://ict.udlap.mx/imc/chi2001/abstracts/borges.pdf>.
- [Braghin et al., 2004] Braghin, C., Gorla, D., and Sassone, V. (2004). A distributed calculus for role-based access control. In *Proc. of 17th Computer Security Foundations Workshop (CSFW'04)*, pages 48–70. IEEE Computer Society. Disponível em [cite-seer.ist.psu.edu/braghin04distributed.html](http://seer.ist.psu.edu/braghin04distributed.html).
- [Brink, 1998] Brink, T. (1998). Usability first: Groupware: Applications. In *CSCW in practice: an Intorudction and case studies*. Disponível em: <http://www.usabilityfirst.com/groupware/applications.txt>. Acesso em 23/02/2004.

- [Brooke, 1993] Brooke, J. (1993). User interfaces for cscw systems. In *CSCW in practice: an Introduction and case studies*. Dan Dapier and Colston Sanger.
- [Bulcão Neto et al., 2004] Bulcão Neto, R. F., Jardim, C. H. O., Camacho-Guerrero, J. A., and Pimentel, M. G. C. (2004). A web service approach for providing context information to cscw applications. In *Proceedings of the 2nd IW3C2 Latin American Web Congress*, pages 78–85. IEEE Computer Society Press.
- [Carstensen et al., 1999] Carstensen, P., Schmidt, K., and Wiil, U. (1999). Supporting shop floor intelligence: a cscw approach to production planning and control in flexible manufacturing. In *GROUP '99: Proceedings of the international ACM SIGGROUP conference on Supporting group work*, pages 111–120, New York, NY, USA. ACM Press.
- [Cattelan et al., 2002] Cattelan, R. G., Andrade, A. R., Penedo, C. F. R., and Pimentel, M. G. C. (2002). iClass: um sistema para captura e acesso de sessões em ambiente educacional. *REIC - Revista Eletrônica de Iniciação Científica*. SBC.
- [Cerami, 2002] Cerami, E. (2002). *Web Services Essentials - Distributed Applications with XML-RPC, SOAP, UDDI and WSDL*. O'Reilly.
- [Chabert et al., 1998] Chabert, A., Grossman, E., Jackson, L., Pietrowicz, S., and Seguin, C. (1998). Java object-sharing in habanero. In *Communications of the ACM*, volume 41, pages 69–76.
- [Chang et al., 2001] Chang, C. J., Zhang, J., and Jiang, T. M. (2001). Formalization of computer supported cooperative work applications. In *Proceedings of the Eighth IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS'01)*.
- [Corporation, 2006] Corporation, M. (2006). COM: Component Object Model Technologies. Disponível online em <http://www.microsoft.com/com/>.
- [Cosquer et al., 2000] Cosquer, F. J. N., Veríssimo, P., Krakowiak, S., and Decloedt, L. (2000). *Support for Distributed CSCW Applications - Lecture Notes in Computer Science*, volume 1752.
- [Dewan and Sharma, 1999] Dewan, P. and Sharma, A. (1999). An experiment in interoperating heterogeneous collaborative systems. In *6th European Conference on Computer Supported Cooperative Work - ECSCW'99*.

- [Dey, 2001] Dey, A. (2001). Understanding and using context. *Personal UbiComp*, 5:4–7.
- [Dey and Abowd, 2000] Dey, A. and Abowd, G. D. (2000). Towards a better understanding of context and contextawareness. In *CHI 2000 Workshop on the What, Who, Where, When, and How of ContextAwareness*.
- [Dourish and Belloti, 1992] Dourish, P. and Belloti, V. (1992). Awareness and coordination in shared workspaces. In *Proceedings of CSCW'92*.
- [Dustdar et al., 2004] Dustdar, S., Gall, H., and Schmidt, R. (2004). Web services for groupware in distributed and mobile collaboration. In *12th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pages 241–247.
- [Eclipse Foundation, 2005] Eclipse Foundation (2005). Eclipse. Disponível online em <http://www.eclipse.org/>.
- [Emmerich, 2000] Emmerich, W. (2000). Software engineering and middleware: a roadmap. In *ICSE - Future of SE Track*, pages 117–129.
- [Ferraiolo and Kuhn, 1992] Ferraiolo, D. F. and Kuhn, D. R. (1992). Role-based access controls. In *15th NIST-NCSC National Computer Security Conference*, pages 554–563. Disponível em citeseer.ist.psu.edu/ferraiolo92rolebased.html.
- [Ferraiolo et al., 2003] Ferraiolo, D. F., Kuhn, D. R., and Chandramouli, R. (2003). *Role-Based Access Control*. Artech House.
- [Fielding et al., 1999] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). RFC 2616: Hypertext Transfer Protocol.
- [Flores et al., 2005] Flores, A., Garcia, I., and Polo, M. (2005). .Net Approach to Run-Time Component Integration. In *Third Latin America Web Congress*, pages 45–48. IEEE Computer Society.
- [Fluckiger, 1995] Fluckiger, F. (1995). *Taxonomy of multimedia applications in Understanding networked multimedia: Applications and technology*. Prentice-Hall.
- [Fortes et al., 2004a] Fortes, R. P. M., Freire, A. P., Vieira, V. H., and Paiva, D. (2004a). An academic web-based agenda and its engineering process. In *VII Workshop*

- Iberoamericano de Ingeniería de Requisitos y Ambientes Software (IDEAS2004)*, pages 151–156. Peruvian Computer Society and National University of San Agustín. ISBN 9972-9876-1-2.
- [Fortes et al., 2004b] Fortes, R. P. M., Silva, M. A. G., Freire, A. P., and Junqueira, D. C. (2004b). SAFE - Software engineering available for everyone. In *Anais V Workshop sobre Software Livre*, pages 203–206, Porto Alegre, RS.
- [Freire and Fortes, 2004] Freire, A. P. and Fortes, R. P. M. (2004). No Risk Planning - um sistema web de agenda para grupos. Proceedings of the 2ndWebMedia Demos and ToolsWorkshop.
- [Fucks et al., 2002] Fucks, H., Gerosa, M., and Lucena, C. (2002). The development and application of distance learning on the internet. *The Journal of Open and Distance Learning*, 17:23–38.
- [Fuks et al., 2002] Fuks, H., Raposo, A. B., and Gerosa, M. A. (2002). Engenharia de groupware: Desenvolvimento de aplicações colaborativas. In *Anais do XXII Congresso da Sociedade Brasileira de Computação*, volume 2, pages 89–128.
- [Gerosa et al., 2003] Gerosa, M. A., Fucks, H., and Lucena, C. J. P. (2003). Suporte à percepção em ambientes digitais de aprendizagem. *Revista Brasileira de Informática na Educação*, 11(2).
- [Gomes et al., 2005] Gomes, R., Rivera, G., and Courtiat, J. (2005). Loosely-coupled integration of csw systems. In *DAIS*, pages 38–49.
- [Gouveia, 2002] Gouveia, L. M. B. (2002). Csw - trabalho colaborativo apoiado por computador. Master’s thesis, Universidade de Aveiro, Coimbra, Portugal.
- [Greenberg, 1997] Greenberg, S. (1997). Collaborative interfaces for the web. *Human Factors and Web Development*, pages 241–254.
- [Gribble and Brewer, 1997] Gribble, S. D. and Brewer, E. A. (1997). System design issues for Internet middleware services: Deductions from a large client trace. In *Proceedings of the 1997 Usenix Symposium on Internet Technologies and Systems (USITS-97)*, Monterey, CA.

- [Gudgin et al., 2001] Gudgin, M., Hadley, M., Moreau, J. J., and Nielsen, H. F. (2001). Simple Object Access Protocol (SOAP) version 1.2. *W3C Working Draft*. Disponível online em <http://www.w3.org/TR/soap12/>.
- [Hawryszkiewicz, 1998] Hawryszkiewicz, I. (1998). An evolution to a CSCW design methodology. In *Third-First Annual Hawaii International Conference on Systems Sciences*, volume 1, Kohala Coast, Hawaii.
- [IBM Corporation, 2005] IBM Corporation (2005). QuickPlace. Programa de Computador. Disponível online: <http://www-130.ibm.com/developerworks/lotus/products/teamworkplace/>.
- [Junqueira et al., 2004] Junqueira, D., Sante, D. G., and Fortes, R. P. M. (2004). Supporting cooperative work in open source software development. In *Proceedings of I2TS'04 (3rd International Information and Telecommunication Technologies Symposium)*, pages 117–124.
- [JXTA Community, 2005] JXTA Community (2005). Projeto JXTA. Disponível online: <http://www.jxta.org>.
- [K. and Briggs, 1984] K., H. and Briggs, F. (1984). *Computer Architecture and Parallel Processing*. McGraw-Hill.
- [Kammer, 2002] Kammer, P. J. (2002). Distributed groupware and web services. In *CSCW 2002 Workshop: Network Services for Groupware*, New Orleans, LA.
- [Kleijnen and Raju, 2003] Kleijnen, S. and Raju, S. (2003). An Open Web Services Architecture. *ACM Queue*, 1(1):38–46.
- [Klockner, 2000] Klockner, K. (2000). BSCW - Educational Servers and Services on the WWW. In *International C4-ICDE Conf. on Distance Education and Open Learning - Competition, Collaboration, Continuity, Change*.
- [Krogdahl et al., 2005] Krogdahl, P., Luef, G., and Steindl, C. (2005). Service-Oriented Agility: Use agile methods for successful SOA development. In *IBM Developer Works*. Disponível online em <http://www-128.ibm.com/developerworks/webservices/library/>.

- [Lai, 2004] Lai, T. L. (2004). Service Quality and Perceived Values's Impact on Satisfaction, Intention and Usage of Short Messagin Service (SMS). In *Information Systems Frontiers*.
- [Li and Li, 2002] Li, D. and Li, R. (2002). Transparent sharing and interoperation of heterogeneous single-user applications. In *ACM Conference on Computer Supported Cooperative Work (CSCW'2002)*, pages 246–255.
- [Ma et al., 2003] Ma, J., Shizuka, M., Lee, J., and Huang, R. (2003). A p2p groupware system with decentralized topology for supporting synchronous collaboration. In *International Concerence on Cyberworlds (CW'03)*, volume 1. IEEE Computer Society.
- [Martin et al., 2003] Martin, J., Arsanjani, A., Tarr, P. L., and Hailpern, B. (2003). Web services: Promises and compromises. *ACM Queue*, 1(1):48–58.
- [Mayrhauser, 2000] Mayrhauser, A. V. (2000). *Software engineering: methods and management*. Academic Press.
- [Mehra et al., 2004] Mehra, A., Grundy, J. C., and Hosking, J. G. (2004). Supporting collaborative software design with a plug-in, web services-based architecture. In *Proceedings of the ICSE 2004 Workshop on directions in Software Engineering Environments*. IEEE Press.
- [Microsystems, 2002] Microsystems, S. (2002). Java Remote Method Invocation Specification. Sun Microsystems. Version 1.4.
- [Microsystems, 2004a] Microsystems, S. (2004a). Java 2 Platform, Micro Edition (J2ME) Web Services. A Technical White Paper.
- [Microsystems, 2004b] Microsystems, S. (2004b). XML - Extensible Markup Language. Disponível em <http://java.sun.com/xml/> Acesso em 18/09/2004.
- [Mills, 1999] Mills, K. L. (1999). Introduction to the eletronic symposium on computer supported cooperative work. *ACM Computing Surveys*, 31(2).
- [Mock et al., 1997] Mock, M., Gergeleit, M., and Nett, E. (1997). Cooperative concurrency control on the web. In *6th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS '97)*, pages 29–31. IEEE Computer Society.

- [Moreira, 1998] Moreira, G. (1998). Sumário executivo. Dissertação de mestrado, Universidade Federal do Rio de Janeiro.
- [Moura, 2001] Moura, M. V. (2001). *No Risk Planning - um sistema de suporte ao trabalho em grupo: uma proposta para Web*. Monografia de Projeto de Graduação.
- [MySpotter Consulting Services, 2005] MySpotter Consulting Services (2005). WSDL2Java Eclipse Plug-in. Disponível online em <http://www.myspotter.com/wsdl2java.shtml>.
- [Nitzke et al., 1999] Nitzke, J., Carneiro, M., and Geller, M. (1999). Aprendizagem cooperativa/colaborativa apoiada por computador - acac. In *Sociedade Brasileira de Informática e Educação (SBIE'99)*, Curitiba - Paraná. Disponível em <http://www.niee.ufrgs.br/alunospg99/mara/>. Acesso em 22/01/2004.
- [Nova et al., 2003] Nova, N., Dillenbourg, P., Wehrle, T., Goslin, J., and Bourquin, Y. (2003). The impacts of awareness tools on mutual modeling in a collaborative video-game. In *Groupware: Design, Implementation and Use - Lecture Notes in Computer Science - (CRIWG03)*, volume 2806, pages 99–108, Autrans, France.
- [Oasis Open, 2005] Oasis Open (2005). UDDI - Universal Description, Discovery and Integration. Disponível online em <http://www.uddi.org/>.
- [Object Management Group, 2003] Object Management Group (2003). CORBA/IIOP Specification. Disponível online em <http://www.omg.org/technology/documents/formal/corba-iiop.htm>.
- [Object Management Group, 2004] Object Management Group (2004). UML - Unified Modeling Language. Disponível em <http://www.omg.org/technology/documents/formal/uml.htm>.
- [Passos and Bernardo, 1998] Passos, C. and Bernardo, H. (1998). Trabalho em grupo e colaborativo - levantamento de informações na web. Dissertação de mestrado em gestão da informação, Universidade de Aveiro.
- [Patil and Kobsa, 2005] Patil, S. and Kobsa, A. (2005). Privacy in collaboration: Managing impression. In *Proceedings of the First International Conference on Online*

Communities and Social Computing, Las Vegas, EUA. Disponível online em cite-seer.ist.psu.edu/718384.html.

[Porto et al., 1994] Porto, A. M. C., Oliveira, P. S., and Kair, T. (1994). Trabalho cooperativo apoiado por computador. Technical report, Centro Tecnológico - Universidade Federal de Santa Catarina.

[Postel, 1980] Postel, J. (1980). RFC 768: User Datagram Protocol. Disponível online em <http://www.ietf.org/rfc/rfc768.txt>.

[Postel, 1981] Postel, J. (1981). RFC 793: Transmission Control Protocol. Disponível online em <http://www.ietf.org/rfc/rfc793.txt>.

[Putnam and Poole, 1987] Putnam, L. L. and Poole, M. S. (1987). Conflict and negotiation. *Handbook of Organizational Communication: An Interdisciplinary Perspective*, pages 549–599.

[Ramduny and Dix, 1997] Ramduny, D. and Dix, A. (1997). Why, what, where, when: Architectures for cooperative work on the world wide web. In *Proceedings of HCI'97*, page 283–301. Springer-Verlag.

[Raposo et al., 2001] Raposo, A. B., Magalhães, L. P., Ricarte, I. L. M., and h. Fuks (2001). Coordination of collaborative activities: A framework for the definition of tasks interdependencies. In *Proceedings of the 7th International Workshop on Groupware - CRIWG*.

[Ray and Kulchenko, 2002] Ray, R. J. and Kulchenko, P. (2002). *Web Services with Pearl*. O'Reilly.

[Rendon et al., 2005] Rendon, O. M. C., Pabon, F. O. M., Vargas, M. J. G., and Guaca, J. A. H. (2005). Architectures for Web Services Access from Mobile Devices. In *Third Latin America Web Congress*, pages 93–97. IEEE Computer Society.

[Ribeiro, 2003] Ribeiro, T. M. (2003). Evolução de uma agenda para grupos na web. Dissertação de mestrado, Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação.

- [Ribeiro et al., 2002] Ribeiro, T. M., Fortes, R. P. M., and Freire, A. P. (2002). Documentação do Software Agenda *No Risk Planning*. Relatórios Técnicos do ICMC, 182.
- [Rosa et al., 2003] Rosa, M. G. P., Borges, M. R. S., and Santoro, F. M. (2003). A conceptual framework for analyzing the use of context in groupware. In *Groupware: Design, Implementation and Use - Lecture Notes in Computer Science - (CRIWG03)*, volume 2806, pages 300–313, Autrans, France.
- [Rose, 2001] Rose, M. (2001). RFC 3080: The Blocks Extensible Exchange Protocol Core. Disponível online em <http://www.ietf.org/rfc/rfc3080.txt>.
- [Roseman and Greenberg, 1996] Roseman, M. and Greenberg, S. (1996). Groupware with groupkit, a groupware toolkit. In *ACM Transactions on Computer-Human Interaction*, volume 1, pages 66–106.
- [Schmidt, 1999] Schmidt, K. (1999). Cooperative work: a conceptual framework: The informatics center of the danish trade union movement. In *Network Management as Cooperative Work: Implications for the Design of Integrated Network Management Systems*.
- [Shneidermann, 1998] Shneidermann, B. (1998). *Design the user interface: strategies for effective human-computer interaction*. Addison-Wesley.
- [Stal, 2002] Stal, M. (2002). Web services: beyond component-based computing. *Communications of the ACM*, 45(10):71–76.
- [Stoeckle et al., 2003] Stoeckle, H., Grundy, J., and Hosking, J. (2003). Approaches to supporting software visual notation exchange. In *Proceedings of the 2003 IEEE Conference on Human-Centric Computing*. IEEE CS Press.
- [The Apache Software Foundation, 2005] The Apache Software Foundation (2005). WSIF: Web Services Invocation Framework. Disponível online em <http://ws.apache.org/wsif/>.
- [The Mozilla Organization, 2005] The Mozilla Organization (2005). Bugzilla Project Home. <http://www.bugzilla.org>.

- [UsabilityFirst, 2002] UsabilityFirst (2002). Usability first: Groupware. Disponível em <http://www.usabilityfirst.com/groupware/> Acesso em 12/11/2003.
- [Vieira and Fortes, 2004] Vieira, V. and Fortes, R. (2004). Integração de aplicações web: Intercâmbio de funcionalidades e informações de contexto usando web services. In *Proceedings of 2nd IW3C2 Latin American Web Congress*, volume 2, pages 217–221. SBC - Brazilian Computer Society.
- [Vieira, 2003] Vieira, V. H. (2003). Aplicações *Groupware*: Um Estudo Comparativo com a Agenda *No Risk Planning*. Monografia de Projeto de Graduação. Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos, Brazil.
- [Vieira and Fortes, 2005] Vieira, V. H. and Fortes, R. P. M. (2005). Documentação do software WS4CSCW (Web services for CSCW). Relatórios Técnicos do Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos, Brazil. número 263, 52 páginas.
- [Vieira et al., 2005] Vieira, V. H., Sante, D. G., Freire, A. P., and Fortes, R. P. M. (2005). A Web Service for cscw applications. In *WebMedia '05: Proceedings of the 11th Brazilian Symposium on Multimedia and the web*, pages 1–3, New York, NY, USA. ACM Press.
- [W3C, 2001] W3C (2001). XML Schema, W3C Recommendation, World Wide Web Consortium. Disponível online em <http://www.w3.org/TR/xmlschema-0/>.
- [W3C, 2002] W3C (2002). Web Services Architecture. W3C Recommendation, World Wide Web Consortium. Disponível online em <http://www.w3.org/TR/ws-arch/>.
- [W3C, 2004a] W3C (2004a). Extensible Markup Language (XML). W3C Recommendation, World Wide Web Consortium. Disponível online em www.w3.org/TR/xml11/.
- [W3C, 2004b] W3C (2004b). Web Services Description Language (WSDL). W3C Recommendation, World Wide Web Consortium. Disponível online em <http://www.w3.org/TR/wsdl12/>.
- [Younas and Iqbal, 2003] Younas, M. and Iqbal, R. (2003). Developing collaborative editing applications using web services. *IEEE Distributed Systems Online Journal of Collaborative Computing*, 4.