

**HIPERDOCUMENTOS ESTRUTURADOS COMO
SUPORTE AO TRABALHO COOPERATIVO EM
SISTEMAS ABERTOS DISTRIBUÍDOS**

Marcos André Silveira Kutova

Orientação:
Prof^a. Dr^a. Maria da Graça Campos Pimentel

*Dissertação¹ apresentada ao Instituto de Ciências Matemáticas e de Computação -
USP, como parte dos requisitos para a obtenção do título de Mestre em Ciências -
Área de Ciências de Computação e Matemática Computacional.*

USP – São Carlos

Abril de 1999

¹ Trabalho realizado com apoio financeiro da FAPESP

*À minha mãe
que me incentivou
a fazer esse trabalho*

AGRADECIMENTOS

À Helcymara, pelo apoio e incentivo.

À Profa. Graça, por expandir meus horizontes.

Ao Roberto, Tavinho e Humberto, companheiros de república e outros episódios.

Aos colegas da USP-São Carlos, pela amizade.

Ao pessoal do laboratório Intermídia, pela ajuda em muitas etapas do trabalho.

À FAPESP, pelo auxílio concedido, e a todos que de alguma forma tornaram este trabalho possível.

RESUMO

Este trabalho contribui com questões relativas à integração das tecnologias de Hipermissão e Trabalho Cooperativo Suportado por Computador (*Computer Supported Cooperative Work - CSCW*), explorando a utilização de hiperdocumentos estruturados no suporte a sessões de trabalho cooperativo. São apresentados conceitos da área de Sistemas Hipermissão Distribuídos e a evolução das linguagens de especificação de hiperdocumentos até a XML (*Extensible Markup Language*). A área de CSCW é apresentada, com ênfase nas principais funcionalidades encontradas nas aplicações dessa área. Essa pesquisa motivou a proposta da metodologia CSCW-SH (*CSCW design based on Structured Hypermedia*) a qual, visando auxiliar a construção de aplicações de CSCW, explora hiperdocumentos estruturados para capturar o conteúdo das sessões de trabalho. Essa metodologia foi utilizada para orientar o projeto e a implementação do DocConf, um ambiente de apoio ao trabalho cooperativo que, por implementar funcionalidades CSCW como componentes, configura-se como um ambiente extensível e aberto. Para ilustrar a utilização do DocConf, o trabalho apresenta a integração do DocConf ao StudyConf, o qual é um ambiente que apoia a navegação e discussão de hiperdocumentos didáticos.

ABSTRACT

The work here reported contributes with the issues related to the integration of the Hypermedia and Computer Supported Cooperative Work (CSCW) technologies, exploiting the use of structured hyperdocuments in order to support cooperative working sessions. The background work presented includes concepts from Hypermedia Systems, with focus on markup languages including XML (Extensible Markup Language), and Computer Supported Cooperative Work, with focus on the main functionalities of CSCW applications. This research has motivated the proposal of CSCW-SH (CSCW design based on Structured Hypermedia): a methodology that, aimed at guiding the development of CSCW applications, exploits structured hyperdocuments as a tool to capture session contents. CSCW-SH was used to guide the project and implementation of the DocConf environment that, by implementing CSCW functionalities as components, is an open and extensible environment. The use of DocConf is illustrated by presenting its integration with the StudyConf environment, which supports cooperative discussion sessions associated with user navigation over didactic hyperdocuments.

ÍNDICE

1. INTRODUÇÃO	1
1.1 CSCW & HIPERDOCUMENTOS ESTRUTURADOS	1
1.2 OBJETIVO.....	3
1.3 ORGANIZAÇÃO DO TRABALHO	3
2. HIPERDOCUMENTOS ESTRUTURADOS.....	5
2.1 CONSIDERAÇÕES INICIAIS	5
2.2 A INTRODUÇÃO DE SGML	6
2.3 A INTRODUÇÃO DE HTML	7
2.4 A INTRODUÇÃO DE XML	8
2.4.1 <i>História e Objetivos</i>	9
2.4.2 <i>Documentos XML</i>	10
2.4.3 <i>Declarações de Tipo de Documento - DTD</i>	12
2.4.4 <i>Ligações</i>	15
2.4.5 <i>Estilos</i>	15
2.5 APLICAÇÕES XML	16
2.5.1 <i>CDF</i>	16
2.5.2 <i>SMIL</i>	17
2.5.3 <i>MathML</i>	18
2.6 CONSIDERAÇÕES FINAIS	19
3. CSCW.....	21
3.1 CONSIDERAÇÕES INICIAIS	21
3.2 HISTÓRIA	22
3.3 DEFINIÇÕES	23
3.4 FUNCIONALIDADES	25
3.4.1 <i>Correio Eletrônico</i>	25
3.4.2 <i>Whiteboards</i>	26
3.4.3 <i>Agendas de Grupo</i>	26
3.4.4 <i>Editores Multiusuários</i>	26
3.4.5 <i>Conferências Eletrônicas</i>	26
3.4.6 <i>Suporte à Decisão</i>	27
3.5 TOOLKITS	28
3.6 CONSIDERAÇÕES FINAIS	29

4.	HIPERDOCUMENTOS ESTRUTURADOS & CSCW.....	30
4.1	CONSIDERAÇÕES INICIAIS.....	30
4.2	CSCW-SH.....	30
4.3	EXEMPLO DE APLICAÇÃO DA METODOLOGIA CSCW-SH.....	32
4.4	CONSIDERAÇÕES FINAIS.....	39
5.	DOCCONF: ESPECIFICAÇÃO E PROJETO	41
5.1	CONSIDERAÇÕES INICIAIS.....	41
5.2	CARACTERÍSTICAS GERAIS DO AMBIENTE DOCCONF	41
5.3	USANDO CSCW-SH NO PROJETO DAS SESSÕES	42
5.3.1	<i>Levantamento dos requisitos das sessões</i>	<i>42</i>
5.3.2	<i>Definição de uma estrutura formal para as sessões</i>	<i>44</i>
5.3.3	<i>Definição da arquitetura de comunicação cliente/servidor.....</i>	<i>45</i>
5.3.4	<i>Projeto e implementação das ferramentas de suporte à sessão.....</i>	<i>47</i>
5.3.5	<i>Projeto e implementação de ferramentas de exploração dos documentos</i>	<i>49</i>
5.4	USANDO CSCW-SH NO PROJETO DOS COMPONENTES	50
5.4.1	<i>Usuários.....</i>	<i>50</i>
5.4.2	<i>Chat.....</i>	<i>51</i>
5.4.3	<i>Whiteboard</i>	<i>52</i>
5.4.4	<i>Votação.....</i>	<i>54</i>
5.5	TRABALHOS RELACIONADOS.....	55
5.6	CONSIDERAÇÕES FINAIS.....	56
6.	DOCCONF: IMPLEMENTAÇÃO, EXTENSÃO E REUSO.....	57
6.1	CONSIDERAÇÕES INICIAIS.....	57
6.2	DOCCONF: IMPLEMENTAÇÃO.....	58
6.2.1	<i>O Ambiente</i>	<i>58</i>
6.2.2	<i>A Sessão.....</i>	<i>59</i>
6.2.3	<i>Os Componentes</i>	<i>59</i>
6.3	DOCCONF: EXTENSÃO COM NOVOS COMPONENTES	61
6.4	DOCCONF: REUSO	62
6.5	CONSIDERAÇÕES FINAIS.....	64
7.	CONCLUSÕES.....	65
7.1	MOTIVAÇÃO	65
7.2	CONTRIBUIÇÕES.....	65
7.3	TRABALHOS FUTUROS.....	66
7.4	CONSIDERAÇÕES FINAIS.....	68
8.	REFERÊNCIAS BIBLIOGRÁFICAS.....	69

ÍNDICE DE FIGURAS

Figura 1: Exemplo de documento XML	11
Figura 2: Seção CDATA para inclusão de caracteres reservados	12
Figura 3: Declaração de elemento para o exemplo da Figura 1	13
Figura 4: Declaração de atributos para o exemplo da Figura 1	13
Figura 5: Declaração de entidade	14
Figura 6: Declaração de notação	14
Figura 7: Exemplo do CDF da Microsoft [Paoli et al 97]	16
Figura 8: Exemplo de documento SMIL [Hoschka et al 98]	18
Figura 9: Exemplo de MathML para a equação $x^2+4x+4=0$ [W3C 98]	19
Figura 10: DTD simplificado para sessão de troca de mensagens [Pimentel et al 98]	35
Figura 11: Arquitetura de comunicação do DocChat	36
Figura 12: Transição de estados para um objeto Sessão	36
Figura 13: Transição de estados em objeto ClienteSessão	37
Figura 14: Tela de acesso a uma sessão	38
Figura 15: Tela de acesso ao gerenciador de sessões	38
Figura 16: Tela de criação de uma sessão	39
Figura 17: Parte do DTD para o documento de uma sessão	45
Figura 18: Processo de comunicação entre clientes e servidor	46
Figura 19: Telas de acesso ao ambiente	47
Figura 20: Gerenciador de sessões	48
Figura 21: Exemplo de um cliente de sessão	48
Figura 22: Documento gerado por uma sessão de DocConf	49
Figura 23: Elementos do componente Usuários	51
Figura 24: Janela do componente Usuários	51
Figura 25: Elementos do componente Chat	51
Figura 26: Configuração do componente Chat	52
Figura 27: Janela do componente Chat	52
Figura 28: Elementos do componente WhiteBoard	53

Figura 29: Janela do componente Whiteboard	54
Figura 30: Elementos do componente Votação.....	54
Figura 31: Componente "Votação"	55
Figura 32: Comunicação entre objetos no cliente e no servidor	59
Figura 33: Comunicação entre uma sessão e seus clientes	59
Figura 34: Interface ClienteComponenteGenérico	60
Figura 35: Interface ClienteComponenteGenérico	60
Figura 36: Diagrama de classes do StudyConf [Macedo et al 99a]	63
Figura 37: Exemplo de utilização do DocConf no StudyConf [Macedo et al 99b]	64

ÍNDICE DE QUADROS

Quadro 1: Classificação espaço/tempo das aplicações de CSCW [Ellis et al 91].....	22
Quadro 2: Metodologia CSCW-SH [Pimentel et al 98].....	32

ÍNDICE DE TABELAS

Tabela 1: Funcionalidades de aplicações de CSCW e do DocChat [Pimentel et al 98]	33
--	----

1 INTRODUÇÃO

A necessidade de se criarem aplicações que permitam às pessoas trabalhar em cooperação, de forma fácil e efetiva, levou à criação de uma nova área de pesquisa, denominada CSCW – Trabalho Cooperativo Suportado por Computador (*Computer Supported Cooperative Work*). Esta área envolve disciplinas tanto da área da ciência da computação quanto da área das ciências sociais (psicologia, sociologia e ciência organizacional) e procura pesquisar os grupos de indivíduos e de que maneira a tecnologia pode auxiliar seus trabalhos [Hills 97; Ellis et al. 91].

A tecnologia de sistemas hipermídia é caracterizada por aplicações nas quais usuários navegam interativamente por documentos que contenham ligações embutidas em seu conteúdo e, como tal, chamados hiperdocumentos. A expansão da Internet e do correspondente aumento da comunidade de usuários da WWW – *World Wide Web* – têm provocado um crescente interesse pela disponibilização, nesse ambiente, de aplicações cooperativas.

A exploração conjunta das tecnologias de CSCW e hipermídia tem vários proponentes. No início da década de 90, Nielsen [90] e Streitz [91] propuseram a exploração de ambientes cooperativos suportados por hiperdocumentos, para facilitar a organização e coordenação de idéias. Streitz comenta que essa união resultaria em sistemas hipermídia multiusuários distribuídos, ao passo que as atividades de cooperação seriam beneficiadas com o suporte a documentos estruturados [Streitz et al 91]. Nesse mesmo contexto, Ishii relata que a tecnologia de hipertexto deve beneficiar o trabalho em grupo, ao sugerir que a própria memória de grupo fosse estruturada e suportada como um hiperdocumento, o qual tem o potencial de fazer referências (hipertextuais) a informações multimídia [Streitz et al 91]. O Sistema NoteCards, da segunda geração de sistemas hipertexto, foi um dos precursores no suporte a tarefas cooperativas com o uso de sistemas hipermídia [Halasz 88].

1.1 CSCW & HIPERDOCUMENTOS ESTRUTURADOS

No contexto de pesquisas na área de CSCW, a WWW pode ser utilizada como infra-estrutura para dar suporte a aplicações de trabalho cooperativo [Busbach et al 96]. A WWW foi criada

para fornecer acesso a documentos científicos distribuídos pela Internet. Os documentos nela disponibilizados são usualmente representados, ou formalizados, através de linguagens de especificação de hiperdocumentos que permitem descrever interligações entre esses documentos independentemente do formato ou plataforma computacional de apresentação.

Bentley et al [95] incluem entre as vantagens da WWW, para uso em CSCW, a sua independência quanto à plataforma, protocolo de rede ou sistema operacional e a interface uniforme entre plataformas. Dix [96] ressalta que a própria WWW pode ser considerada um ambiente de trabalho cooperativo, pois permite o compartilhamento de informações, e cita a possibilidade de execução de programas na WWW como forma de aumentar a interatividade entre usuários e provedores de informações e serviços.

O padrão atualmente utilizado para formalizar os documentos apresentados na WWW é definido pela linguagem HTML – *HyperText Markup Language* [Raggett et al 97] – que, por sua vez, é definida como uma aplicação da linguagem SGML – *Standard Generalized Markup Language* [ISO 86]. Ao formalizar a estrutura e o conteúdo do documento, as definições da HTML suportam referências a dados em várias mídias. A estrutura definida é simples e torna fácil a elaboração de novos documentos, fato que garantiu a expansão da WWW.

A HTML, entretanto, não é totalmente adequada para uso em alguns tipos de aplicações executadas sobre a WWW, como as de CSCW. Suas principais deficiências são a falta de suporte à estrutura do documento, o que limita a pesquisa apenas a trechos de texto, e a não extensibilidade, que impede os desenvolvedores de criar suas próprias *tags* para refletir as relações semânticas do conteúdo do documento [Bray et al 98; Bosak 97]. Já a linguagem SGML, por sua vez, é um tanto complexa para ser utilizada diretamente na definição de novas classes de documentos. Assim, membros do W3C – *World Wide Web Consortium* – propuseram uma nova linguagem para definição de documentos estruturados que tivesse a extensibilidade que falta à HTML e fosse mais simples que a SGML, que recebeu o nome de XML – *Extensible Markup Language* [Bray et al 98; W3C 97]. Outras linguagens complementares à XML estão sendo desenvolvidas no W3C e fornecerão, em termos de documentação estruturada, vários recursos úteis para aplicações na WWW.

Procurando avançar na união das tecnologias de hipermídia e trabalho cooperativo suportado por computador, Pimentel et al [98] propõem uma metodologia para desenvolvimento de aplicações

CSCW, utilizando hiperdocumentos para formalizar a estrutura das sessões de trabalho. Essa metodologia, CSCW-SH – *CSCW design based on Structured Hypermedia*, estabelece não só uma seqüência de passos a serem seguidos durante o desenvolvimento de aplicações como também quais resultados devem ser obtidos em cada um desses passos.

1.2 OBJETIVO

O objetivo deste trabalho é investigar o uso integrado das tecnologias de hipermídia e de CSCW no desenvolvimento de aplicações cooperativas que exploram a Internet, em geral, e a WWW, em particular. Nesse contexto, apresenta-se o projeto e desenvolvimento do ambiente DocConf que, utilizando a metodologia CSCW-SH, explora tal integração.

No ambiente DocConf, hiperdocumentos XML são utilizados para armazenar a memória resultante de sessões de trabalho cooperativo [Pimentel et al 99b]. O ambiente fornece um conjunto de ferramentas, denominadas componentes, que implementam várias formas de interação entre os participantes de uma sessão. Ao seguir a metodologia CSCW-SH, o DocConf configura-se como um ambiente extensível e aberto, como recomendado para as aplicações construídas para a Internet e a WWW.

1.3 ORGANIZAÇÃO DO TRABALHO

Essa seção apresenta o conteúdo dos demais capítulos desta dissertação.

O Capítulo 2 apresenta um histórico da tecnologia de documentação estruturada, desde o seu surgimento na década de 60 até os dias de hoje. São apresentados conceitos básicos da área e os motivos pelos quais se procurou desenvolver um padrão que garantisse extensibilidade e validação de documentos. Ao final, é detalhada a nova metalinguagem XML e são mostrados exemplos de aplicações que a utilizam.

O Capítulo 3 introduz conceitos da área de Trabalho Cooperativo Suportado por Computador após um breve histórico do desenvolvimento da área. São apresentadas também as funcionalidades encontradas nas diversas aplicações já desenvolvidas e os *toolkits* — conjuntos de ferramentas para desenvolvimento de novas aplicações.

No Capítulo 4, é apresentada a metodologia CSCW-SH criada para orientar o desenvolvimento de aplicações de trabalho cooperativo, utilizando hiperdocumentos que formalizem a estrutura das sessões de trabalho. Para complementar a descrição da metodologia, é também apresentada a aplicação desenvolvida, DocChat – *Documented Chat*, que suporta, na Internet, uma sessão de discussão estruturada de acordo com um DTD SGML.

O Capítulo 5 apresenta o projeto do ambiente de trabalho cooperativo DocConf, realizado de acordo com a metodologia CSCW-SH. Esse ambiente utiliza a arquitetura de componentes, na qual cada componente implementa uma funcionalidade das aplicações de CSCW. DocConf armazena e formaliza a memória das sessões de trabalho cooperativo em hiperdocumentos estruturados de acordo com um DTD XML.

Os detalhes da implementação do DocConf são discutidos no Capítulo 6, no qual também é apresentado um roteiro para a criação de novos componentes. A aplicação do DocConf no projeto StudyConf, que visa a apoiar a navegação e discussão on-line de hiperdocumentos didáticos disponibilizados na WWW [Macedo et al 99], é também discutida.

O Capítulo 7 conclui esta dissertação resumando a motivação do trabalho, retomando as contribuições resultadas e discutindo propostas para trabalhos futuros.

2

HIPERDOCUMENTOS ESTRUTURADOS

2.1 CONSIDERAÇÕES INICIAIS

Da análise de um documento, é possível identificar sua estrutura. Um livro, por exemplo, possui um título, um índice e vários capítulos; cada capítulo possui também um título e uma ou mais seções. Essa estrutura pode variar significativamente entre tipos de documentos diferentes. Uma carta não possui nenhum dos elementos citados para o livro, mas possui data, saudação, conteúdo e assinatura. Menus de restaurantes, programas de teatro e documentos financeiros são outros exemplos de documentos com estruturas próprias. Isso levou alguns pesquisadores a procurar um padrão de representação de documentos que pudesse ser utilizado em todas as estruturas que se fizessem necessárias.

Dentro de um documento existem alguns outros elementos que também merecem atenção especial. Citações, palavras destacadas e palavras de outro idioma devem ser representadas de forma diferente, para facilitar a compreensão do leitor. Em um documento bancário, pode-se ter, por exemplo, uma representação diferente para o número de conta corrente, o que facilita sua distinção e localização.

A representação de documentos deve considerar, também, a necessidade da especificação de relações intra e entre documentos. Exemplos dessas relações são as referências bibliográficas, que levam aos documentos originais (entre), e as citações de figuras, que posicionam o documento nas figuras correspondentes (intra). Nessas relações, o autor oferece ao leitor alternativas de leitura. Quando documentos são disponibilizados através de um ambiente computacional, pode-se ter um grande número de associações intra e entre documentos. Quando as associações podem ser selecionadas interativamente pelo leitor, resultando na apresentação automática do conteúdo associado, os documentos são chamados de hiperdocumentos. Um hiperdocumento bancário pode ter, por exemplo, ligações que levem a outros documentos como a ficha cadastral do cliente ou o resumo de suas aplicações financeiras.

2.2 A INTRODUÇÃO DE SGML

A marcação de documentos corresponde ao processo de incluir marcas ao documento para identificar sua estrutura e/ou formato no qual deve ser apresentado. Muito antes dos computadores, a marcação era usada por um editor para passar instruções para os tipógrafos. Quando as funções de tipografia começaram a ser computadorizadas, uma linguagem para marcação de documentos se fez necessária [Watson 92]. Com essa visão, no final da década de 60, a CGA – *Graphic Communications Association* – criou o comitê GenCode para especificar códigos genéricos de identificação de elementos nos documentos. Pouco tempo depois, a IBM propôs a linguagem GML – *Generalized Markup Language*, criada por Charles Goldfarb, Ed Mosher e Raymond Lorie, cujas premissas básicas eram: (1) a marcação devia descrever a estrutura do documento ao invés de suas características físicas e (2) a marcação devia ser legível para um programa ou ser humano.

À medida que novos tipos de documentos surgiam, cada um com seu conjunto de marcações e apresentador próprio, aumentava a necessidade de uma forma padronizada de descrevê-los. Em 1978, foi criado o comitê de Linguagens de Computadores para Processamento de Textos do ANSI – *American National Standards Institute*, composto por membros das comunidades GML e GenCode. O comitê apresentou em 1980 a primeira proposta da linguagem SGML – *Standard Generalized Markup Language* – que permitia a declaração dos tipos de documentos, formalizada por um DTD – *Document Type Declaration*.

As principais vantagens da SGML, que garantiram a sua aceitação e que até hoje permanecem, foram:

- **Estrutura hierárquica** — A SGML permite a representação da estrutura hierárquica dos elementos em um documento, o que facilita o processamento do documento, como em operações de pesquisa, antes limitadas a trechos de texto.
- **Flexibilidade** — A SGML não dita quais tipos de elementos devem ser criados ou como eles se relacionam, isso fica a critério do usuário. O autor pode, então, criar os diversos tipos de documentos que lhe convierem.
- **Especificação formal** — Todos os elementos contidos em um documento SGML são declarados antes de ele começar. Essa declaração é formalizada em um DTD. O programa

que utilizar o documento SGML processa esse documento em conjunto com a declaração, o que permite validá-lo.

- **Representação legível** — Um documento SGML tem conteúdo textual, podendo ser manipulado em qualquer editor ou lido e compreendido com facilidade por uma pessoa. Por exemplo, não deve ser difícil deduzir o significado do trecho:
<negrito> muito </negrito>

A SGML, entretanto, não é poderosa o suficiente para lidar com documentos multimídia e hipermídia nos quais apareçam relações temporais ou ligações hipertexto complexas. Um comitê do *International Organization for Standardization*, presidido por Charles Goldfarb, desenvolveu, então, a linguagem HyTime – *Hypermedia / Time-based Structuring Language*, uma extensão da SGML [ISO 92; DeRose & Durand 94].

A HyTime foi projetada para representar interconexões complexas de componentes de informação em uma forma processável por um computador. Os componentes a serem conectados com o uso de especificações HyTime podem ser documentos SGML, arquivos de áudio, gráficos, seqüências de vídeo ou mesmo simples arquivos texto. Com a HyTime é possível especificar os locais pretendidos dos dados em relação a lugar e tempo. Assim, os autores podem criar seus dados na mídia que desejarem e incorporá-los aos seus documentos através das ligações.

2.3 A INTRODUÇÃO DE HTML

No final dos anos 80, a linguagem SGML já estava sendo usada em várias organizações, entre elas o CERN – *European Laboratory for Particle Physics*. Utilizando um conjunto de elementos definidos em um DTD SGML, Tim Berners-Lee, criou a linguagem HTML – *HyperText Markup Language*. A HTML teve seu DTD definido formalmente pela NCSA – *Nebraska Council of School Administrators*, que o popularizou através do seu apresentador *Mosaic*, cuja interface gráfica é a predecessora de muitos outros apresentadores utilizados na atualidade.

A primeira versão da HTML surgiu como um padrão de fato. Com o surgimento de novos apresentadores (navegadores) e a evolução destes, surgiu a necessidade de se criar um padrão formal para a Internet sendo produzida, assim, a versão 2.0 da linguagem. Esta versão também incluiu na especificação a definição de formulários, para suprir a necessidade de troca de

informações de modo bidirecional, isto é, o usuário poder enviar dados ao servidor. A versão 3.2, trabalho conjunto do W3C – *World Wide Web Consortium*, que inclui empresas como a IBM, Microsoft, Netscape, Sun e outras, adicionou recursos como tabelas, Applets e fluxo de texto sobre as imagens.

A versão atual da HTML é a 4.0, que introduz recursos para internacionalização, facilidades de uso para pessoas com deficiências, novos modelos de tabelas e formulários, folhas de estilos, *scripts*, objetos embutidos e outros. A especificação completa encontra-se em [Raggett et al 97].

A HTML se tornou amplamente utilizada como um formato para especificação de documentos hipermídia simples, devido à simplicidade e facilidade com que os documentos podem ser escritos. A linguagem especifica um modelo para documentos textuais com interligações e objetos em outras mídias. Entretanto, mesmo que a HTML atenda à atual demanda dos documentos hipertextuais, essa linguagem foi projetada para definição da forma de apresentação de um documento e oferece poucos recursos para a definição do conteúdo de suas informações. Assim, a HTML é um formato inadequado para o armazenamento, a longo prazo, de documentos hipertexto que possam vir a ser apresentados por outros meios além dos apresentadores HTML.

2.4 A INTRODUÇÃO DE XML

A HTML é uma aplicação SGML e, como tal, sua linguagem possui uma gramática fixa e não-extensível. Isto a torna simples e fácil de usar, mas dificulta suas expansões e a reutilização dos documentos com ela especificados. Qualquer modificação requer uma atualização no padrão e, conseqüentemente, nos apresentadores. A SGML, por ser uma metalinguagem, é muito mais poderosa e extensível e poderia resolver esse problema; entretanto é bastante complexa. Os desenvolvedores preferem optar pela criação de ferramentas e *plug-ins* para atingir seus propósitos a dar suporte à SGML diretamente na WWW.

A XML – *Extensible Markup Language* – veio para solucionar esse impasse, sendo uma versão simplificada da SGML para ser utilizada na WWW, proposta pelo W3C.

2.4.1 História e Objetivos

A simplificação da SGML não é uma meta nova; há muitos anos, pesquisadores discutem essa idéia. Entretanto, apenas em julho de 1996 foi criado o grupo de trabalho *SGML Editorial Review Board*, cujo objetivo era especificar uma simplificação da SGML otimizada para a WWW.

Duas conferências foram cruciais para divulgar a XML: a SGML'96 e a WWW6 – *Sixth International World Wide Web Conference*. Nelas, o grupo de trabalho conseguiu atrair o interesse da comunidade de SGML mostrando as vantagens da simplificação e facilidade de uso. A comunidade da WWW também demonstrou interesse na XML, mas por outras razões: extensibilidade e validação. Mais um passo importante foi dado poucas semanas antes da WWW6, quando a Microsoft anunciou que o formato escolhido a ser utilizado em suas pesquisas com a tecnologia *push* seria baseado em XML. Logo, a Netscape também concordou em dar suporte à nova linguagem.

No dia 10 de fevereiro de 1997, o grupo (que viria a se chamar XML WG em julho de 1997), conseguiu que a XML se tornasse uma recomendação do W3C. Durante os trabalhos, o grupo foi acompanhado pelo *XML Special Interest Group*, conhecido no início como SGML WG.

Os objetivos propostos pelo grupo para a XML foram [Bray et al 98]:

- A XML deve ser usada de forma direta na Internet;
- A XML deve suportar uma variedade de aplicações;
- A XML deve ser compatível com a SGML;
- Deve ser fácil escrever programas que processem documentos XML;
- O número de características opcionais da XML deve ser mínimo, de preferência nulo;
- Os documentos XML devem ser legíveis pelo homem e razoavelmente claros;
- O projeto com a XML deve ser preparado rapidamente;
- O projeto com a XML deve ser formal e conciso;
- Documentos XML devem ser fáceis de criar;
- Concisão, na marcação XML, deve ser de mínima importância.

2.4.2 Documentos XML

A metalinguagem XML descreve uma classe de objetos de dados chamados documentos XML e descreve, parcialmente, o comportamento de programas que os processem [Bray et al 98]. Esse processamento é realizado através de dois módulos. O primeiro módulo, **processador XML**, é usado para ler o documento XML e prover acesso ao seu conteúdo e estrutura. Assume-se que o processador XML trabalha em benefício do segundo módulo, a **aplicação**.

Um documento XML é um objeto de dados que possui estruturas física e lógica [Bray et al 98]. Fisicamente, ele é composto por unidades de armazenamento chamadas **entidades**. As entidades podem ser compostas por dados textuais ou binários. Quando textuais, podem representar trechos de textos ou **marcações**. Quando binários, são incluídos no documento através de uma referência externa. Todo documento XML deve começar com a entidade documento, ou raiz. Logicamente, o documento é composto de declarações, elementos, comentários, referências e instruções de processamento, todas indicadas através das marcações nos documentos.

Em linguagem de marcação de modo geral, as marcações atendem a dois propósitos básicos: (a) separar os objetos lógicos no documento e (b) especificar quais funções devem ser executadas nesses objetos [Goldfarb 90]. Para garantir a independência do documento quanto à sua aplicação ou formatação, foi criada a **marcação generalizada** (*generalized markup*). Nesse caso, as informações adicionadas ao documento são apenas a respeito de sua estrutura. A questão da aparência fica a cargo das **folhas de estilos** (*stylesheets*), que associam aos objetos encontrados no documento a especificação de como devem ser apresentados.

As marcações são representadas por **tags**. Uma *tag* inicia com o símbolo menor-que (<) se for uma *start-tag*, ou por menor-que seguido do símbolo de divisão (</) se for *end-tag*. Em ambos os casos ela termina com o símbolo de maior-que (>). Todo o texto entre a *start-tag* e a *end-tag* de um objeto lógico faz parte do seu conteúdo.

A Figura 1 mostra um exemplo de documento XML. Este documento contém uma referência bibliográfica em conformidade com o DTD "bibref.dtd" (apresentado nas figuras 3 e 4 da próxima seção). Nesse documento pode-se perceber o uso das marcações separando os objetos lógicos. As *tags* <title> e </title>, por exemplo, envolvem o título da referência.

```

<?XML version="1.0" rmd="ALL" ?>
<!DOCTYPE bibref SYSTEM "bibref.dtd">
<bibref id="Bray97" type="journal">
  <title>Extensible Markup Language (XML)</title>
  <publication volume="2" number="4" pages="29-66">
    World Wide Web Journal</publication>
  <authors>
    <name>Tim <lastname>Bray</lastname></name>
    <name>Jean <lastname>Paoli</lastname></name>
    <name>C. M. <lastname>Sperberg-McQueen</lastname></name>
  </authors>
  <date>August 7, 1997</date>
  <abstract>
    <enfa>Extensible Markup Language (XML)</enfa> is an extremely simple
    dialect of <enfa>SGML</enfa> which is completely described in this
    document. <sep/>
    The goal is to enable generic SGML to be served, received, and
    processed on the Web in the way that is now possible with
    <enfa>HTML</enfa>. XML has been designed for ease of implementation
    and for interoperability with both SGML and HTML.
  </abstract>
</bibref>

```

Figura 1: Exemplo de documento XML

O objeto lógico mais simples em um documento é o **elemento**. Um elemento é composto de três partes: *start-tag*, conteúdo e *end-tag*. As *tags* são utilizadas para definir o início e o fim do elemento dentro do documento. O conteúdo identifica a natureza do elemento, que pode ser uma seqüência de caracteres ou um objeto filho. A possibilidade de se criarem elementos dentro de outro elemento garante a estrutura hierárquica dos documentos XML. Os elementos são, portanto, representados na forma: <nome_elemento> </nome_elemento>.

Os elementos podem ser vazios, isto é, não possuir qualquer conteúdo. Quando isso ocorrer, devem ser representados por uma *tag* única, na forma: <nome_elemento/>.

Um elemento pode ser classificado qualitativamente através de **atributos**. Um atributo é um par com nome e valor presente na *start-tag*, logo após o nome desse elemento. O valor do atributo deve estar sempre entre aspas, como: <nome_elemento nome_atributo="valor_atributo">....

Um documento pode ter também **referências** a outras entidades. Estas entidades podem ser internas ou externas. Quando internas, normalmente se referem a um caráter de língua estrangeira ou a um símbolo reservado para a marcação. Quando externas, referem a arquivos,

documentos ou imagens. A referência a uma entidade é feita iniciando-se por “&”, seguido do nome da entidade e terminando-se com “;”, como em “&”.

Outra forma de se incluírem em um documento trechos que contenham os caracteres reservados para marcação, como uma listagem do código fonte de um programa, é através de uma **seção CDATA**, de *Character DATA*, que se inicia com “<![CDATA[” e termina com “]]>”, como indicado na Figura 2. Todos os caracteres dentro dessa seção são ignorados.

```
<![CDATA[
    *p = &q;
    b = (i <= 3 );
]]>
```

Figura 2: Seção CDATA para inclusão de caracteres reservados

A XML também permite que se adicionem **comentários** ao texto. Os comentários se iniciam com “<!--” e terminam com “-->” e não fazem parte do conteúdo textual do documento. Os comentários podem ser incluídos em qualquer lugar do documento e podem conter quaisquer caracteres, à exceção da seqüência “--”.

As **instruções de processamento** são uma forma de se passarem informações para a aplicação. Assim como os comentários, elas não fazem parte do conteúdo textual do documento XML. As instruções de processamento são da forma: <?nome_ip dados_ip?>. As aplicações devem processar apenas as instruções que reconhecerem, ignorando as demais. As instruções que começam com XML são reservadas para extensões da linguagem.

2.4.3 Declarações de Tipo de Documento - DTD

A declaração permite ao documento passar metainformações sobre seu conteúdo ao *parser*. A metainformação inclui a seqüência e aninhamento de *tags* permitidas, valores e tipos dos atributos, nomes dos arquivos externos que podem ser referenciados e outras entidades.

Existem quatro tipos de declarações permitidas na XML: declaração de elementos, declaração de lista de atributos, declaração de entidades e declaração de notações.

A declaração de elementos identifica os nomes dos elementos e a natureza de seu conteúdo. A Figura 3 mostra a declaração de elementos para o documento da Figura 1.

```

<!ELEMENT bibref      (title, publication, authors, date, abstract?) >
<!ELEMENT title       (#PCDATA)* >
<!ELEMENT publication (#PCDATA)* >
<!ELEMENT authors     (name)* >
<!ELEMENT name        ((#PCDATA)*, lastname) >
<!ELEMENT lastname    (#PCDATA)*> >
<!ELEMENT date        (#PCDATA)* >
<!ELEMENT abstract    (#PCDATA | enfa | sep)* >
<!ELEMENT enfa        (#PCDATA)* >
<!ELEMENT sep         EMPTY >

```

Figura 3: Declaração de elemento para o exemplo da Figura 1.

Alguns símbolos possuem significados especiais na declaração do conteúdo do elemento. A vírgula (,) significa que os elementos devem ocorrer na ordem especificada; o mais (+) indica que o elemento deve ocorrer uma ou mais vezes; o asterisco (*) indica que o elemento deve ocorrer zero ou mais vezes; a interrogação (?) indica que o elemento é opcional; #PCDATA (*parseable character data*) indica uma seqüência de caracteres; EMPTY indica elemento vazio e ANY indica que o elemento pode ter qualquer conteúdo.

Os atributos são declarados após os elementos, identificando quais os atributos de cada elemento e que valores eles podem conter. A Figura 4 mostra a declaração dos atributos dos elementos do exemplo da Figura 1.

```

<!ATTLIST bibref
  id      ID          #required
  type    (journal | book) 'journal' >
<!ATTLIST publication
  volume  CDATA       #implied
  number  CDATA       #implied
  pages   CDATA       #implied >

```

Figura 4: Declaração de atributos para o exemplo da Figura 1

Cada atributo é composto por três partes: nome, tipo e valor padrão. Os nomes são de livre escolha, mas com algumas restrições. O tipo pode ser CDATA, que define seqüências de caracteres, ID, que indica um identificador, IDREF, que faz referência a um ID, ENTITY, que faz referência a uma entidade, NMTOKEN, que é uma forma restrita do CDATA, ou uma lista de nomes. Existem quatro tipos de valor padrão: #REQUIRED, que indica que o atributo deve obrigatoriamente ser especificado; #IMPLIED, que indica que o atributo é opcional; "valor", que

indica o valor padrão do atributo quando outro não for especificado; #FIXED "valor", que indica o valor obrigatório do atributo quando esse for especificado.

A declaração seguinte é a de entidades. Existem três tipos de entidades: as internas, que associam uma seqüência de caracteres a um nome, como mostrado na Figura 5; as externas, que permitem a inclusão de um arquivo no documento; as entidades parâmetro, que são declaradas e usadas dentro do DTD (identificadas pelo símbolo %).

```
<!ENTITY W3C "World Wide Web Consortium">
<!ENTITY logomarca SYSTEM "/imagens/logo.gif" NDATA GIF87A>
```

Figura 5: Declaração de entidade

O último tipo de declaração encontrado no documento XML é a declaração de notação, que identifica tipos de dados binários externos. Essa informação é passada para a aplicação, que a interpreta da forma que desejar. Uma declaração de notação é a da forma apresentada na Figura 6.

```
<!NOTATION GIF87A SYSTEM "GIF">
```

Figura 6: Declaração de notação

As declarações devem vir no início do documento, logo após instruções de processamento opcionais. Em todos os documentos XML deve haver uma raiz, que contém todos os demais elementos. A declaração pode estar presente dentro do arquivo, ser referenciada externamente ou de ambas as formas. Como muitas vezes não se deseja validar o documento, mas apenas ler seu conteúdo, o parâmetro RMD – *Required Markup Declaration* – na instrução de processamento inicial informa se se deve validar apenas as declarações internas (rmd="internal"), tanto as internas quanto as externas (rmd="all") ou nenhuma delas (rmd="none").

Os documentos XML podem ser bem formados ou válidos. Documentos são bem formados se obedecerem à sintaxe da XML. Os documentos que não puderem passar por um *parser* XML não são documentos bem formados. Para que um documento bem formado seja também válido, deve conter um DTD apropriado, e obedecer às restrições daquela declaração.

2.4.4 Ligações

A funcionalidade das ligações implementada na HTML é uma parte mínima do que é associado ao conceito de hipertexto. A HTML suporta apenas uma ligação unidirecional especificada integralmente dentro do documento. Em um sistema hipertexto, como definido nas décadas de 70 e 80, devem aparecer os seguintes mecanismos [Bosak 97; Trindade & Pimentel 97]:

- Ligações bidirecionais;
- Ligações especificadas e gerenciadas fora do documento;
- Ligações com vários destinos;
- Ligações agregadas (com várias fontes);
- Atributos nas ligações.

A especificação de ligações na XML está em desenvolvimento no W3C e está dividida em duas partes: XLink – *XML Linking Language*, que define ligações simples e estendidas, e XPointer – *XML Pointer Language*, que permite uma ligação para um ponto qualquer de um documento (p.ex.: a segunda frase do terceiro parágrafo) bastando percorrer a sua estrutura.

2.4.5 Estilos

Como os documentos XML não possuem um conjunto fixo de *tags*, sua apresentação fica dependente de uma folha de estilos. Uma das opções atuais é a utilização dos mecanismos da CSS (*Cascading Style Sheet*) para conferir fontes, cores, posicionamentos, fundos e muitos outros aspectos de apresentação a esses documentos [Culshaw et al 97].

O W3C está trabalhando na criação da XSL – *Extensible Stylesheet Language*, baseado no padrão DSSSL (*Document Style Semantics and Specification Language*), que oferecerá funcionalidades além da CSS, como reordenação de elementos. Os autores da XSL esperam uma coexistência dos dois padrões, sendo a CSS destinada a documentos XML simples e a XSL destinada a documentos complexos que necessitariam de funções como acesso à estrutura hierárquica dos elementos ou inclusão de scripts ou programas [Adler 97].

2.5 APLICAÇÕES XML

As próximas seções mostram iniciativas de algumas organizações e pesquisadores na adoção da linguagem XML para estruturação de documentos.

2.5.1 CDF

Na forma convencional de acesso à WWW, as páginas são enviadas ao usuário mediante uma solicitação desse. A tecnologia *push* fornece uma forma alternativa, em que páginas selecionadas são enviadas pelo servidor ao cliente de maneira automática. Depois que todo o conteúdo dessas páginas tiver sido recebido, o usuário poderá se desconectar da Internet e, ainda assim, acessar as informações.

Pensando nessa tecnologia, a Microsoft lançou a primeira forma de descrever o conteúdo de um site, o CDF – *Channel Definition Format* [Paoli et al 97]. Através do CDF, um provedor de informações pode criar um *Active Channel*, um conjunto de páginas da WWW a ser disponibilizado para *push*. Os arquivos CDF são compostos por listas de URLs, que apontam para as páginas. Eles podem ter também título e resumo informativo, indicando calendário de atualização e organização hierárquica do canal. A Figura 7 mostra um exemplo de um arquivo CDF.

```
<?XML version="1.0" rmd="NONE" ?>
<!DOCTYPE Channel SYSTEM
      "http://www.microsoft.com/standards/channels.dtd">
<CHANNEL>
  <SCHEDULE>
    <INTERVALTIME HOUR="2"/>
    <LATESTTIME MIN="30"/>
  </SCHEDULE>
  <TITLE>Internet Explorer News</TITLE>
  <ITEM HREF="http://www.microsoft.com/ie/new/666784.html">
    <ABSTRACT>The latest news on Internet Explorer.</ABSTRACT>
    <TITLE>Latest support for CDF</TITLE>
  </ITEM>
</CHANNEL>
```

Figura 7: Exemplo do CDF da Microsoft [Paoli et al 97]

O CDF foi a primeira aplicação XML implementada pela Microsoft e levou a equipe a desenvolver dois *parsers* XML para serem distribuídos com o Microsoft Internet Explorer, um escrito em C++ e o outro em Java.

2.5.2 SMIL

Outro trabalho que está sendo desenvolvido por membros do W3C é o de sincronização de objetos multimídia para apresentação na WWW. A SMIL – *Synchronized Multimedia Integration Language* – é uma linguagem, em fase de proposta, que permite a integração de um conjunto independente de objetos multimídia em uma apresentação multimídia sincronizada [Hoschka et al 98]. Um documento SMIL é um documento XML definido de acordo com a especificação XML 1.0 [Bray et al 98].

Com a SMIL, pode-se criar apresentações que integrem textos, imagens, áudios ou vídeos. Pode-se integrar à apresentação um tipo diferente de arquivo, como uma página HTML, desde que a aplicação saiba como tratá-lo. Esses componentes podem ser acessados em algum servidor da WWW, através de sua URL.

Os elementos multimídia, programados para serem apresentados, podem ser atômicos (únicos) ou combinados. Quando combinados, podem ser de forma paralela, seqüencial ou de ambas as formas.

A Figura 8 mostra um exemplo de documento SMIL. Nesse documento é criado um layout com dois canais. Os canais controlam posição, tamanho e escala dos componentes. Na primeira cena do exemplo, é mostrada uma imagem do lado esquerdo, enquanto um vídeo é exibido do lado direito. O vídeo é acompanhado por um áudio. A segunda cena se inicia após 60 segundos e a imagem é substituída por um outro vídeo, que é acompanhado de um novo áudio. Esse novo vídeo é também utilizado como uma ligação para a página do interlocutor. As ligações na SMIL funcionam de forma semelhante às da HTML.

A programação de exibição dos componentes é especificada através das *tags*: `<par>` (*parallel*) e `<seq>` (*sequential*). Suas relações temporais são definidas através dos atributos dessas *tags*. O usuário pode controlar a apresentação utilizando botões com funções conhecidas tais como *stop*, *fast-forward* e *rewind*. A aplicação ainda pode fornecer funções adicionais como acesso aleatório (posicionamento em qualquer lugar) ou *slow-motion*.

```

<smil>
  <head>
    <layout type="text/smil-basic">
      <channel id="left-video" left="20" top="50" z-index="1"/>
      <channel id="right-video" left="150" top="50" z-index="1"/>
    </layout>
  </head>
  <body>
    <par>
      <seq>
        
        <a href="http://www.w3.org/People/Berners-Lee">
          <video src="tim-video" channel="left-video"/>
        </a>
      </seq>
      <seq>
        <audio src="joe-audio" dur="60s"/>
        <audio src="tim-audio"/>
      </seq>
      <video id="jv" src="joe-video" channel="right-video"/>
    </par>
  </body>
</smil>

```

Figura 8: Exemplo de documento SMIL [Hoschka et al 98]

A SMIL está sendo projetado para que seja fácil escrever apresentações, mesmo utilizando um editor de textos. A chave do sucesso da HTML foi a facilidade com que um hipertexto atrativo podia ser criado sem uma ferramenta de autoria sofisticada. A SMIL promete a mesma coisa para hipermídia sincronizada [Hoschka et al 98].

2.5.3 MathML

Comunicar notações matemáticas é uma tarefa importante. A estrutura intrincada de uma expressão matemática reflete a lógica das operações de uma forma sutil. Apesar da HTML fornecer uma forma precária de visualização de expressões matemáticas, ela não fornece qualquer condição para análise ou processamento dessas expressões. Por isso, o grupo de trabalho HTML-Math do W3C propôs a linguagem MathML, baseada na XML [Miner & Ion 97].

As *tags* da MathML são divididas em *tags* de apresentação e *tags* de conteúdo. As *tags* de apresentação são responsáveis pela formatação das expressões matemáticas. As *tags* de conteúdo suportam a codificação do conteúdo matemático por trás de uma expressão. Os tipos de *tag* podem ser misturados dentro de uma única expressão matemática. A Figura 9 mostra um

exemplo da MathML para a expressão $x^2+4x+4=0$ usando *tags* de apresentação e *tags* de conteúdo.

Tags de Apresentação	Tags de Conteúdo
<pre> <mrow> <mrow> <msup> <mi>x</mi> <mn>2</mn> </msup> <mo>+</mo> <mrow> <mn>4</mn> <mo>&invisibletimes;</mo> <mi>x</mi> </mrow> <mo>+</mo> <mn>4</mn> </mrow> <mo>=</mo> <mn>0</mn> </mrow> </pre>	<pre> <apply> <plus/> <apply> <power/> <ci>x</ci> <cn>2</cn> </apply> <apply> <times/> <cn>4</cn> <cin>x</ci> </apply> <cn>4</cn> </apply> </pre>

Figura 9: Exemplo de MathML para a equação $x^2+4x+4=0$ [W3C 98]

2.6 CONSIDERAÇÕES FINAIS

A XML traz muitas promessas para o futuro, mas já é reconhecida hoje como linguagem para representação da estrutura de documentos tanto no meio acadêmico quanto no meio comercial. Os anúncios da Sun Microsystems sobre a criação de um *parser* em Java para o desenvolvimento de aplicações utilizando a XML e da Microsoft sobre o suporte à XML e a XSL na versão 5.0 do Internet Explorer contribuíram significativamente para a massificação dessa tecnologia.

As aplicações de CSCW também podem tirar grande proveito dos avanços na área de documentos estruturados, pois contém necessidades similares às citadas acima. Por exemplo, em um ambiente onde aplicações necessitem compartilhar informações, a estruturação através de um DTD XML garante o acesso aos dados armazenados nos hiperdocumentos independentemente, da aplicação que tenha gerado esse hiperdocumento, de como armazena internamente seus dados ou em que linguagem foi desenvolvida.

Essa estruturação das informações, através de DTDs XML em aplicações de CSCW, é utilizada pelo ambiente DocConf proposto como objetivo deste trabalho. Para isso, o DocConf segue a metodologia CSCW-SH, que orienta o mapeamento da estrutura das sessões de trabalho cooperativo em elementos e atributos de um DTD. Para cada sessão executada, utilizando o DocConf, é gerado um hiperdocumento formalizado de acordo com esse DTD.

3

CSCW

3.1 CONSIDERAÇÕES INICIAIS

A necessidade de se criarem aplicações que permitam às pessoas trabalhar juntas de forma fácil e efetiva trouxe consigo uma nova área de pesquisa, denominada CSCW – Trabalho Cooperativo Suportado por Computador (*Computer Supported Cooperative Work*). Esta área envolve disciplinas tanto da área da ciência da computação quanto das ciências sociais (psicologia, sociologia e ciência organizacional) e procura pesquisar os grupos de indivíduos e como a tecnologia pode auxiliá-los em seus trabalhos.

As aplicações de CSCW servem a três propósitos básicos [Hills 97]:

- **Comunicação** — Permite que as pessoas compartilhem informações. Esse compartilhamento traz, além da velocidade de acesso aos dados e sua consistência, redução de custos com transportes e remessas de material. Como exemplo, há a grande aceitação e utilização, no mundo inteiro, do correio eletrônico (*e-mail*).
- **Coordenação** — Auxilia as pessoas a coordenar seus trabalhos individuais com os dos demais membros do grupo. Isso é necessário para evitar que as pessoas se engajem em tarefas conflitantes ou repetidas e para que as tarefas realizadas por um membro sejam percebidas por todos os demais, uma vez que trabalham com dados compartilhados [Ellis et al 91]
- **Cooperação** — Ajuda as pessoas a trabalhar em grupo. Mas a introdução dos computadores nesse trabalho acarreta a necessidade de novos protocolos sociais, pois os computadores não suportam toda a interatividade comum em uma reunião face a face [Lubich 95].

A introdução dos computadores nas reuniões de trabalho trouxe também novas possibilidades para o grupo: reuniões em locais e tempos diferentes. Ellis et al [91] citam uma classificação para as aplicações de CSCW, que levam em consideração a relação espaço/tempo (Quadro 1). Alguns pesquisadores afirmam, inclusive, que essas aplicações devem ser flexíveis o suficiente para abranger os quatro quadrantes de espaço e tempo.

	Mesmo Local	Locais Diferentes
Mesmo Momento	Interação face a face	Interação síncrona distribuída
Momentos Diferentes	Interação assíncrona	Interação assíncrona distribuída

Quadro 1: Classificação espaço/tempo das aplicações de CSCW [Ellis et al 91]

3.2 HISTÓRIA

O termo CSCW foi usado pela primeira vez em 1984 quando Paul Cashman e Irene Greif organizaram um workshop para reunir pessoas de áreas diferentes, mas com um interesse comum pelo trabalho em grupo e pelas maneiras que a tecnologia pode dar suporte a esse trabalho [Grudin 94].

A pesquisa nessa área, entretanto, não é tão recente. Desde o início da década de 60, Douglas C. Engelbart explorava o uso de computadores para o trabalho com grupos de alta performance. Na década seguinte, com o advento dos minicomputadores, surgiu a Automação de Escritórios (*Office Automation*). Procurou-se adaptar e integrar as aplicações monousuário mais bem sucedidas, como os processadores de texto e as planilhas eletrônicas, para suportarem o trabalho em grupo.

Entretanto, apenas o uso da tecnologia computacional não foi suficiente para sustentar essa transformação. Havia a necessidade de se estudar como as pessoas trabalham em grupo e como a tecnologia influi nesse processo. Assim, os tecnólogos começaram a aprender sobre a atividade de grupo com economistas, sociólogos, antropólogos, cientistas organizacionais e educadores. Passaram a surgir, nesse ponto, os primeiros estudos em direção ao Trabalho Cooperativo Suportado por Computador.

Em 1986, a ACM patrocinou o primeiro evento na área, denominado CSCW'86. Nesta conferência, foram apresentados doze artigos sobre tecnologia computacional e seu impacto nas organizações e dezessete artigos sobre o estudo de pessoas e grupos. A conferência passou a ser realizada bianualmente. Em 1989, iniciou-se a versão europeia da Conferência — ECSCW, também realizada bianualmente. Os trabalhos em CSCW não pararam por aí. Além de vários

outros eventos específicos da área, pode-se encontrar artigos dessa área em eventos de Hipermídia, Multimídia e HCI – *Human-Computer Interaction*.

Pesquisas, artigos e trabalhos em CSCW podem ser encontrados em todo o mundo, mas vale a pena ressaltar as grandes contribuições prestadas pelos pesquisadores japoneses, através de empresas de computadores, como a NEC e a Toshiba, universidades ou, até mesmo, das companhias de telecomunicações. O TeamWorkStation é um exemplo desses trabalhos, que enfatiza o uso da multimídia em um ambiente de trabalho cooperativo [Ishii & Miyake 91]. Projetado por Hiroshi Ishii e Naomi Miyake, essa aplicação procura mesclar as imagens da tela do computador com o vídeo ao vivo.

3.3 DEFINIÇÕES

A pesquisa de suporte computadorizado para trabalhos em grupo é relativamente recente e, como acontece com toda tecnologia nova, há certa controvérsia para definir e nomear os conceitos da área. Outro fator que acentua ainda mais essa confusão é o seu caráter interdisciplinar. Os pesquisadores vieram de diferentes áreas, leram e escreveram para diferentes periódicos, compareceram a diferentes conferências e acabaram, por consequência, desenvolvendo diferentes terminologias [Grudin 91].

À medida que CSCW amadurece, entretanto, começa a haver cada vez mais interação entre os pesquisadores das diversas disciplinas. A primeira geração dos projetos foi caracterizada por abordagens isoladas em diferentes áreas. Já a segunda mostra maior flexibilidade e cooperação entre pesquisadores, utilizando conceitos e avanços tanto da área da ciência da computação quanto das áreas de ciências sociais.

Mas não existe uma definição rígida para **CSCW**. Esse termo é usado como um fórum para pesquisadores de várias áreas discutirem seus trabalhos. Há apenas um consenso de que o CSCW deve estudar o trabalho em grupo e como os computadores podem auxiliá-los nesse trabalho. Os pesquisadores concordaram em manter o nome CSCW enquanto não surge uma definição mais precisa para seus trabalhos.

Outro termo que também está necessitando de uma definição mais precisa é “**cooperação**”. De acordo com alguns dicionários, cooperação implica o trabalho de um grupo que tenha objetivo

comum e trabalhe em direção a esse objetivo. Mas Hannes Lubich [95] levanta algumas questões sobre o termo: O que significa o termo “cooperativo” na sigla CSCW? Estaria ele descrevendo que um grupo de pessoas trabalha em um conjunto de tarefas? O estilo de “dividir para conquistar” seria ainda considerado trabalho cooperativo? Finalmente, no aproveitamento de trabalho feito por terceiros com o objetivo de alcançar novos resultados e onde não há qualquer interação entre os participantes, há cooperação?

Essa confusão quanto à terminologia se deve ao caráter interdisciplinar do CSCW. Como ressaltaram Liam Bannon e Kjeld Schmidt, na primeira conferência européia sobre CSCW (E-CSCW'89), o termo “trabalho cooperativo” muitas vezes é substituído por “trabalho colaborativo”, “trabalho de grupo” e “trabalho coletivo” [Lubich 95].

James Bair, [Bair 89] apud [Lubich 95], apresentou um modelo de níveis para classificação da colaboração em um trabalho:

1. **Informação** Comunicação anônima feita em listas de discussão, USENET, periódicos. O remetente não conhece os destinatários;
2. **Coordenação** Comunicação entre pessoas de um grupo que têm interesses em comum, mas que não trabalham com um mesmo objetivo;
3. **Colaboração** Comunicação entre pessoas que trabalham juntas e com um mesmo objetivo, porém essas pessoas ainda são avaliadas individualmente;
4. **Cooperação** Comunicação quando não existe mais o conceito do indivíduo, somente o do grupo.

Muitos autores ainda questionam se essa diferença é realmente importante para seus propósitos. Assim, esta dissertação adota os termos "cooperação" e "colaboração" como sinônimos.

Há ainda o termo **Groupware**, criado por Peter e Trudy Johnson-Lenz em 1978, [Johnson-Lenz 80]² apud [Grudin 91], para descrever sistemas que suportam interação de grupos. Uma definição de Groupware, dada por Ellis et al [91] é: “sistemas baseados em computadores que suportam grupos de pessoas envolvidas em uma tarefa (ou objetivo) comum e que fornecem uma interface para um ambiente compartilhado”.

² Apesar do termo *Groupware* ter sido criado em 1978 por Peter e Trudy Johnson-Lenz, [Johnson-Lenz 80] foi o primeiro trabalho publicado que o definiu.

Alguns pesquisadores equiparam groupware a CSCW. Essa idéia é, contudo, rejeitada por muitos outros. Existem dois motivos citados por Simon Gibbs para que groupware seja diferenciado de CSCW [Lubich 95]:

- Groupware está associado à tecnologia enquanto CSCW cobre uma variedade de questões e conceitos. Além disso, a parte “ware” sugere alguma analogia com as palavras hardware ou software, e ambas se referem a tecnologia;
- CSCW só está relacionado a trabalho. Groupware, por outro lado, liga-se a vários outros tipos de aplicação, desde jogos multiusuário até realidade virtual, portanto “trabalho” não deve aparecer na definição de groupware.

Finalmente, existe o termo **CMC** — Comunicação Mediada por Computador, que engloba todos e quaisquer contextos de comunicação que utilizem computadores.

3.4 FUNCIONALIDADES

Quando surgiram as primeiras aplicações CSCW, cada uma implementava uma funcionalidade diferente. Dentre estas funcionalidades, algumas eram destinadas à troca de mensagens; outras à troca de documentos. Com o passar do tempo, tais aplicações passaram a incorporar várias outras funcionalidades. Esta seção apresenta as principais funcionalidades encontradas nessas aplicações [Hills 97; Borges et al 95].

3.4.1 Correio Eletrônico

O correio eletrônico (*e-mail*) permite que uma mensagem seja enviada de uma pessoa a outra através de uma rede de computadores como a Internet. Os sistemas de correio eletrônico estão se tornando cada vez mais rápidos e mais confiáveis. Hoje, uma mensagem pode ser enviada para um destino internacional em apenas alguns minutos.

Esta é considerada a funcionalidade mais importante dentro de uma organização e vem, aos poucos, substituindo algumas formas de comunicação já existentes, como os memorandos e o fax. Graças aos novos protocolos, como o MIME – *Multipurpose Internet Mail Exchange* (RFC 2045), pode-se enviar mensagens ricas em conteúdo que incluam documentos, imagens, som ou, até mesmo, vídeo.

Terry Crowley e Bob Kraut [Ensor et al 90] arrolam como razões para o sucesso do correio eletrônico o baixo custo, a flexibilidade (não há necessidade de estruturação), a execução em diversas plataformas e a interface simples.

3.4.2 Whiteboards

Whiteboard é o nome dado ao recurso de compartilhamento de uma tela por vários usuários. Nesta tela pode ser visualizado um documento ou uma imagem em que os usuários podem fazer anotações ou marcações. A cada usuário é atribuída uma cor, pela qual se consegue identificar o autor das anotações. Esses sistemas seguem o conceito WYSIWIS (*What You See Is What I See*).

3.4.3 Agendas de Grupo

As agendas de grupos são utilizadas para sincronizar as atividades dos componentes de um grupo. Um sistema com esse recurso deve ter calendário, agenda coletiva, lista de telefones (ou *e-mail*) e lista de tarefas. A agenda deve ser capaz de programar eventos, como uma reunião, resolvendo automaticamente todos os conflitos com as agendas pessoais dos integrantes do grupo.

3.4.4 Editores Multiusuários

Pesquisas recentes têm mostrado que na maioria das organizações, muitas pessoas trabalham na composição, revisão e apresentação de um documento [Hills 97]. Com um editor multiusuário, essas pessoas podem escrever juntas esse documento de forma síncrona ou assíncrona. Na forma assíncrona, apenas uma pessoa pode editar o documento por vez. Já na forma síncrona, duas ou mais pessoas podem editar o documento, mas apenas uma por trecho de texto.

3.4.5 Conferências Eletrônicas

As conferências permitem que os participantes de uma conferência troquem informações em tempo real. Elas são amplamente utilizadas na Internet, e têm como exemplo mais simples o *chat*. Um chat, entretanto, só permite a troca de informações textuais. Sistemas mais avançados, que permitem a troca de áudio e vídeo, são conhecidos como sistemas de videoconferência.

O TVS (TeleMídia Videoconferencing System) é um sistema de videoconferência que estrutura as sessões de modo a permitir a definição do papel dos participantes e fornecer suporte à votação. O sistema possibilita a transmissão de mensagens de áudio e vídeo de forma síncrona e, além disso, a troca de documentos multimídia em conformidade com o MHEG [Oliveira & Soares 96]. Adicionalmente, o sistema permite uma ampla configuração das sessões. A cada configuração podem ser definidos o organizador, o interlocutor e os demais participantes, bem como a base de dados que contém documentos a serem compartilhados.

3.4.6 Suporte à Decisão

O objetivo desta funcionalidade é sobrepor as dificuldades encontradas nas reuniões comuns. Uma reunião de decisão é constituída por três etapas: geração de idéias, organização das idéias e votação. Com este recurso, idéias podem ser geradas simultaneamente e, automaticamente, anotadas na memória da reunião. A contribuição pode ser anônima, evitando prováveis conflitos hierárquicos ou raciais [Daily et al 96]. Ao final, o sistema fornece facilidades para organização e votação das idéias.

O SACE-CSCW (*Synchronous Asynchronous common Environment for Computer Supported Cooperative Work*) é uma aplicação que dá suporte ao processo de tomada de decisão em grupo [Santos et al 96]. Ele fornece recursos para a configuração e execução das três sessões necessárias a uma reunião desse tipo: geração, organização e votação de idéias. Na primeira, as idéias podem ser identificadas ou anônimas. Para cada reunião configurada, o sistema permite a criação de pauta e agenda, atribuição de papéis sociais para os participantes e o agendamento da reunião. As sessões podem ser síncronas ou assíncronas (via memória de grupo) e o compartilhamento de dados numa sessão de geração de idéias pode ser configurado em três modos diferentes para cada participante: global, parcial e individual.

QUORUM é um sistema de suporte à decisão em grupo que enfoca o processo de desenvolvimento de software [Borges & Araújo 94]. O sistema procura decompor o problema em subproblemas, na busca de linhas de ação para se chegar à conclusão. A reunião é controlada por um coordenador e o correio eletrônico é adotado para discussões e divulgação de informações. Toda informação gerada é armazenada para uma futura consulta.

Bacelo & Becker [97] apresentaram um Sistema de Apoio à Decisão em Grupo para reuniões distribuídas e assíncronas. O sistema possui um modelo de argumentação para estruturar e tornar objetiva a reunião, emprega técnicas de votação para selecionar as alternativas discutidas e adota uma posição rígida de coordenação da reunião através de uma norma e de um facilitador.

3.5 TOOLKITS

Os *toolkits* de groupware são conjuntos de ferramentas que oferecem os recursos necessários para o desenvolvimento de aplicações de trabalho cooperativo. Greenberg e Rosemann [98] afirmam que um *toolkit* de groupware deve fornecer os seguintes componentes: uma arquitetura de execução que gerencie a criação, interconexão e comunicação de processos centralizados ou distribuídos; um conjunto de abstrações de programação, que permitam ao desenvolvedor controlar o comportamento dos sistemas; construtores básicos, que possam ser integrados à aplicação; e gerenciadores de sessões. Todos esses aspectos são tratados no *toolkit* desenvolvido pelos autores, GroupKit, baseado na linguagem de scripts Tcl/Tk [Greenberg & Rosemann 96].

A arquitetura de execução deve fornecer recursos para facilitar a programação e o gerenciamento da comunicação entre processos e pode ser de dois tipos: centralizada ou replicada. Em arquiteturas centralizadas, existe uma máquina central rodando uma aplicação servidora que controla toda a entrada e saída de dados para os clientes. Já em arquiteturas replicadas, existe uma réplica do programa sendo executada em cada computador e que deve coordenar tanto as ações locais quanto remotas.

As abstrações de programação permitem aos desenvolvedores manter a consistência dos dados e visões do sistema. Exemplos comuns dessas abstrações são as chamadas remotas de procedimentos (RPC) e o gerenciamento de eventos (p.ex.: entrada e saída de participantes).

Os construtores básicos são utilizados na criação da interface do sistema e podem ser tanto versões de construtores monousuário já existentes (como botões e barras de rolagem) quanto novos construtores para atividades exclusivamente em grupo. Em ambos os casos, os construtores devem permitir acoplamento forte (as ações sobre o elemento serão vistas por todos) ou fraco (somente o resultado dessas ações será visto).

Os gerenciadores de sessão são responsáveis por estabelecer e controlar as conexões dos usuários às sessões. Alguns gerenciadores criam e encerram as sessões automaticamente enquanto que outros permitem que os usuários realizem estas atividades.

3.6 CONSIDERAÇÕES FINAIS

Existem muitos benefícios que as aplicações de CSCW podem trazer para grupos de trabalho. De acordo com Greenberg [Crow et al 97], no futuro todas as aplicações serão compartilhadas, mesmo as aplicações em que apenas uma pessoa esteja presente e justifica essa idéia dizendo que não há motivo para alguém usar um editor de textos monousuário quando terá que usar um editor compartilhado em várias ocasiões. Greenberg também espera que as primitivas hoje encontradas nos toolkits sejam embutidas nos sistemas operacionais tradicionais.

Com o avanço da WWW, projetos foram desenvolvidos para investigar o uso da WWW como plataforma para aplicações de trabalho cooperativo [Bentley et al 95; Chabert et al 98; Dix 96; Walther 96]. Seguindo essa linha de pesquisa, o ambiente DocConf procura permitir a colaboração na WWW através de componentes que implementam algumas das funcionalidades de aplicações de CSCW citadas neste capítulo. Assim como os toolkits, o DocConf possibilita o desenvolvimento de novos componentes, que permitem novas formas de interação entre os membros de um grupo. Para isto, o DocConf segue a metodologia CSCW-SH que orienta esse desenvolvimento e formaliza as sessões de trabalho utilizando DTDs XML.

4

HIPERDOCUMENTOS ESTRUTURADOS & CSCW

4.1 CONSIDERAÇÕES INICIAIS

Em um painel na ACM Hypertext'91, Streitz et al [91] discutiram o papel da hipermídia nas aplicações de CSCW. Streitz defendia que Hipermídia e CSCW eram áreas de pesquisa e desenvolvimento em evolução e que deviam convergir de forma a se tornarem complementares. As principais características da tecnologia de Hipermídia, que Streitz cita como úteis às aplicações de CSCW, são: a facilidade de criar módulos de informações e interligá-los; o reuso de material existente; a possibilidade de definir diferentes visões das informações para cada membro de um grupo; a associação de elementos. De acordo com o pesquisador, funcionalidades das ferramentas de CSCW (por exemplo, tela compartilhada, controle de visões e controle de acesso aos dados) têm importante papel nos sistemas hipermídia distribuídos.

Streitz ressaltava, entretanto, que faltara à tecnologia de CSCW o suporte às definições de estruturas de documentos necessárias para comunicação e colaboração. Nesse contexto, Pimentel et al [98] propõem a metodologia CSCW-SH – *CSCW design based on Structured Hypermedia*, para desenvolvimento de aplicações de trabalho cooperativo utilizando hiperdocumentos que formalizem a estrutura das sessões de trabalho.

Este capítulo apresenta a metodologia CSCW-SH e uma aplicação desenvolvida de acordo com ela. O autor desta dissertação colaborou tanto no desenvolvimento da metodologia quanto na construção dessa aplicação. A experiência adquirida durante esse trabalho serviu como base para o desenvolvimento do ambiente DocConf.

4.2 CSCW-SH

A metodologia CSCW-SH tem por objetivo facilitar o projeto de aplicações de CSCW, estabelecendo uma seqüência de passos a serem seguidos e apontando os resultados que devem ser esperados de cada passo. A metodologia propõe o uso de hiperdocumentos estruturados para formalizar as sessões de trabalho cooperativo em função dos seguintes benefícios:

- facilidade do processamento e intercâmbio dos documentos associados à sessão;
- geração dinâmica de documentos que contenham a memória da sessão conforme, o andamento da mesma;
- possibilidade do estabelecimento de ligações hipermídia, associadas à sessão como na troca de mensagens entre os participantes.

A metodologia impõe uma arquitetura cliente/servidor, como a possibilitada pela Internet, e a utilização de padrões para a definição da estrutura de documentos estruturados associados às sessões de trabalho cooperativo suportados.

O Quadro 2 dá uma visão geral da metodologia CSCW-SH, descrevendo os passos seguidos e o resultado esperado em cada um deles.

No primeiro passo, **levantamento dos requisitos das sessões**, deve-se realizar um levantamento exaustivo das funcionalidades necessárias à aplicação CSCW. Como resultado, deve ser gerado um quadro de requisitos, que contenha as informações necessárias para cada uma dessas funcionalidades (por exemplo, informações sobre presidência e moderador em uma sessão de discussão) e sua dinâmica (fases da sessão).

O quadro de requisitos será utilizado no passo dois, **definição de uma estrutura formal para as sessões**. Esta definição deve detalhar cada um dos componentes e produzir como resultado um DTD SGML cujos elementos e atributos correspondam à estrutura e à dinâmica especificada para as sessões.

No passo três, **definição da arquitetura de comunicação cliente/servidor**, são definidos os módulos cliente e servidor e a forma como eles se comunicam. Deve ser descrito, nesse passo, o protocolo de comunicação entre os módulos, por exemplo, com a utilização de um diagrama de transição de estados.

No passo quatro, **projeto e implementação das ferramentas de suporte à sessão**, são projetadas e construídas as ferramentas especificadas, nos passos anteriores, para a aplicação. Entre as ferramentas a serem construídas, estão o servidor de sessões e o cliente de sessões.

No último passo, **projeto e implementação de ferramentas de exploração dos documentos**, são projetadas e implementadas ferramentas para intercâmbio dos documentos gerados durante as sessões e armazenados no servidor. Ferramentas para comunicação com uma base de dados, ou para geração de páginas HTML correspondentes às sessões, são exemplos das ferramentas criadas nesse passo.

CSCW Design based on Structured Hypermedia

Passos	Resultados Intermediários
1: Levantamento de requisitos das sessões	<i>Quadro de requisitos</i>
2: Definição de uma estrutura formal para as sessões de trabalho	<i>DTD SGML definindo elementos, sua estrutura, conteúdo e atributos</i>
3: Definição da arquitetura de comunicação clientes/servidor	<i>Módulos da arquitetura, diagramas de transição para clientes e servidor</i>
4: Projeto e implementação das ferramentas de suporte à sessão de trabalho	<i>Ferramentas para criação e gerenciamento de sessões de trabalho</i>
5: Projeto e Implementação de ferramentas de exploração dos documentos	<i>Ferramentas para apresentação, processamento ou intercâmbio dos documentos correspondentes às sessões de trabalho</i>

Quadro 2: Metodologia CSCW-SH [Pimentel et al 98]

4.3 EXEMPLO DE APLICAÇÃO DA METODOLOGIA CSCW-SH

Esta seção apresenta um exemplo de utilização da metodologia CSCW-SH na construção de uma aplicação de CSCW sobre a Internet. A aplicação desenvolvida, *DocChat — Documented Chat*, suporta sessões de discussão, estruturadas de acordo com um DTD SGML, e armazena as informações correspondentes a cada sessão em um hiperdocumento estruturado. A ferramenta permite também a geração de uma página HTML, disponibilizada em um servidor da WWW, correspondente ao hiperdocumento relativo à sessão.

PASSO 1: LEVANTAMENTO DOS REQUISITOS DAS SESSÕES

A Tabela 1 mostra uma lista de funcionalidades encontradas em várias aplicações CSCW de modo geral. A terceira coluna mostra quais funcionalidades são desejadas para o *DocChat*.

Funcionalidades:	Implementas em:	Doc Chat
Suporte à sessão síncrona	SACE / BSCW / Quorum / TVS / Groco / Groupkit / Habanero / Bacelo & Becker / Mushroom	√
Suporte a sessão assíncrona	SACE / BSCW / Quorum / Bacelo & Becker	√
Controle do andamento da sessão	SACE / Quorum / Bacelo & Becker	
Sessão dividida em subseções	SACE / Quorum / Bacelo & Becker	
Cadastro prévio dos participantes	GroupKit / Quorum / SACE	√
Definição do papel dos participantes	SACE / TVS / Quorum / Bacelo & Becker	
Suporte a agendamento e ordem de pauta	SACE	
Facilitador / Moderador	TVS / Quorum / Bacelo & Becker / SACE	
Percepção de presença	GroupKit / GroCo / SACE	
Percepção das ações do grupo	SACE / TVS / GroCo	√
Coordenação das atividades	GroupKit / Quorum / Bacelo & Becker / GroCo / SACE	
Possibilidade de anonimato	SACE	√
Troca de mensagens textuais	SACE / Habanero / Quorum / Bacelo & Becker / GroCo	
Troca de mensagens de áudio e/ou vídeo	TVS	
Troca de documentos textuais	TVS / Habanero / BSCW	
Troca de documentos multimídia	TVS	
Seleção do modo de visão (Global, Parcial ou Individual)	SACE	
Suporte a votação	SACE / TVS / GroupKit / Quorum / Bacelo & Becker	
Armazenamento de documentos com dados relativos à sessão, como por exemplo: <ul style="list-style-type: none"> • relação dos participantes e seus papéis • conteúdo das intervenções de cada participante • tipos de intervenções (nova idéia, tipo do argumento) 	SACE / Quorum / Bacelo & Becker	√
Processamento posterior de documentos relativos à sessão, como por exemplo: <ul style="list-style-type: none"> • relatório com a contribuição de cada participante • relatórios de resumos • relatório sobre as conclusões e argumentos favoráveis 	SACE	
Conversão para HTML dos documentos armazenados	GroCo	√
Execução sobre a Internet	BSCW / GroupKit / GroCo / Habanero	√
Independência de plataforma	BSCW / Habanero / GroCo	√
Suporte ao desenvolvimento de aplicações	GroupKit / Mushroom / Habanero / GroCo	

Tabela 1: Funcionalidades de aplicações de CSCW e do DocChat [Pimentel et al 98]

PASSO 2: DEFINIÇÃO DE UMA ESTRUTURA FORMAL PARA AS SESSÕES

A Figura 10 corresponde à versão simplificada de um DTD SGML para um documento que contém as informações a serem manipuladas em uma sessão, e define a linguagem ChatML. Assim, para uma sessão desse tipo, são elementos o cabeçalho (HEAD) e o corpo (BODY).

Os seguintes atributos relativos a uma sessão ChatML estão associados ao cabeçalho:

- identificador (ID);
- status de sessão pública ou privada (PUBLIC);
- identificação do remetente quando da propagação das mensagens (IDENTIFIED).

O cabeçalho contém, como elementos obrigatórios, as informações de:

- assunto tratado (SUBJECT);
- criador da sessão (OWNER).

Os demais elementos do cabeçalho são opcionais:

- data e hora iniciais (STARTDATE e STARTTIME) e finais (ENDDATE e ENDTIME);
- um campo para informações adicionais (NOTE);
- um campo para identificação do presidente (PRESIDENT);
- um campo para identificação do secretário (SECRETARY);
- um campo para lista dos participantes permitidos na reunião (PERSONNEL), e identificação de cada um dos participantes (WHO) .

O corpo de um documento ChatML contém:

- informações referentes à entrada dos participantes da sessão (LOGIN), cujo conteúdo fornece a identificação do participante (#PCDATA), e cujo atributo obrigatório identifica o momento de entrada (DATE);
- idem para saída do participante da sessão (LOGOUT);
- conteúdo da mensagem enviada (MENSAGEM), que possui atributo obrigatório referente ao momento em que a mensagem foi enviada (DATE) e é composta dos elementos FROM, TO e MSG, correspondentes respectivamente ao remetente, destinatário (opcional) e conteúdo textual da mensagem (#PCDATA) propriamente dita.

```

<!--SIMPLIFIED DTD for CHAT SESSION -->
<!--      elements      min      content      -->
<! ELEMENT  CHATML      (HEAD, BODY)      >
<! ELEMENT  HEAD      - 0      (SUBJECT, OWNER, STARTDATE?, STARTTIME?,
ENDDATE?, ENDDTIME?, NOTE?, PRESIDENT?,
SECRETARY?, PERSONNEL?)      >
<! ELEMENT  SUBJECT      - 0      (#PCDATA)      >
<! ELEMENT  STARTDATE      - 0      (#PCDATA)      >
<! ELEMENT  ENDDATE      - 0      (#PCDATA)      >
<! ELEMENT  STARTTIME      - 0      (#PCDATA)      >
<! ELEMENT  ENDDTIME      - 0      (#PCDATA)      >
<! ELEMENT  NOTE      - 0      (#PCDATA)      >
<! ELEMENT  PRESIDENT      - 0      (#PCDATA)      >
<! ELEMENT  SECRETARY      - 0      (#PCDATA)      >
<! ELEMENT  PERSONNEL      - 0      (WHO+)      >
<! ELEMENT  WHO      - 0      (#PCDATA)      >
<! ELEMENT  BODY      - 0      (LOGIN+ | MESSAGE+ | LOGOUT+)      >
<! ELEMENT  LOGIN      - -      (#PCDATA)      >
<! ELEMENT  LOGOUT      - -      (#PCDATA)      >
<! ELEMENT  MESSAGE      - -      (FROM, TO?, MSG)      >
<! ELEMENT  FROM      - 0      (#PCDATA)      >
<! ELEMENT  TO      - 0      (#PCDATA)      >
<! ELEMENT  MESH      - 0      (#PCDATA)      >
<!--      element      attribute      value
-->
<ATTLIST  CHATML      ID      ID      #REQUIRED
          DATE      #CDATA      #REQUIRED
          PUBLIC      (YES|NO)      YES
          IDENTIFIED      (YES|NO)      YES      >
<!ATTLIST  LOGIN      DATE      #CDATA      #REQUIRED      >
<!ATTLIST  MESSAGE      DATE      #CDATA      #REQUIRED      >
<!ATTLIST  LOGOUT      DATE      #CDATA      #REQUIRED      >

```

Figura 10: DTD simplificado para sessão de troca de mensagens [Pimentel et al 98]

PASSO 3: DEFINIÇÃO DA ARQUITETURA DE COMUNICAÇÃO CLIENTE/SERVIDOR

A Figura 11 ilustra a arquitetura de comunicação do sistema. O módulo servidor mantém uma lista das sessões abertas. Quando uma nova sessão é aberta, o servidor cria um objeto da classe Sessão e registra esse objeto no servidor de nomes do Java RMI – *Remote Method Invocation*. Assim, quando um usuário se conecta ao servidor, recebe instantaneamente a lista de sessões abertas.

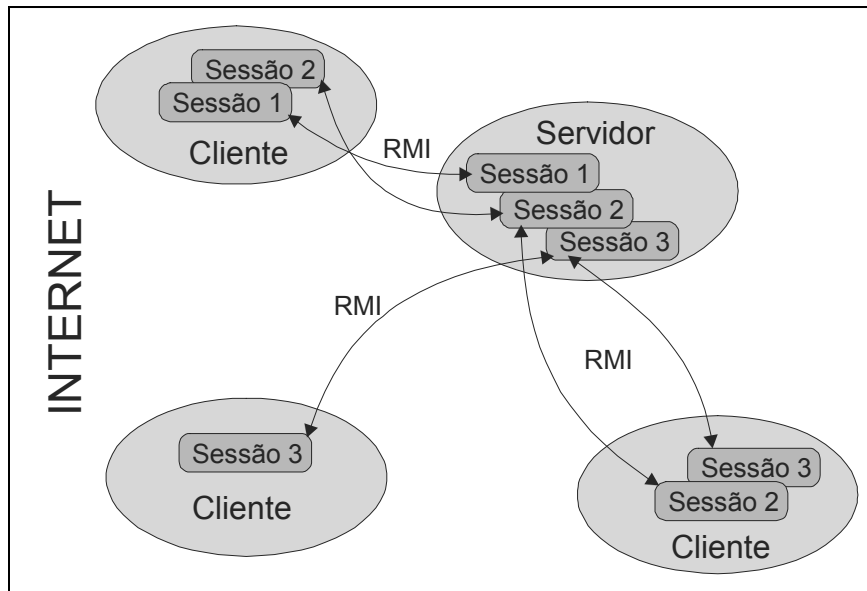


Figura 11: Arquitetura de comunicação do DocChat

O objeto Sessão mantém uma lista dos usuários que estão ativos e é responsável pela divulgação dos eventos recebidos de qualquer um dos usuários. Esses eventos podem ser: (a) conexão de um novo usuário (Conecta_Usuário), (b) envio de uma mensagem (Mensagem), ou (c) desconexão de um usuário (Desconecta_Usuário). Assim que recebe um desses eventos, o objeto Sessão realiza o processamento necessário e notifica seus clientes sobre o que ocorreu. Esse processamento é ilustrado na Figura 12.

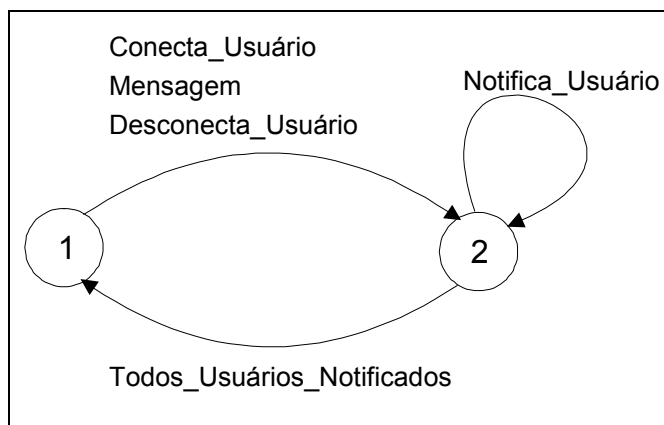


Figura 12: Transição de estados para um objeto Sessão

A Figura 13 ilustra as mensagens enviadas pelo objeto ClienteSessão: um cliente envia mensagens no caso da conexão (Conecta) ou desconexão (Desconecta) do usuário a ele associado; envia mensagens preparadas pelo usuário ao servidor (Envia_Mensagem); e recebe mensagens enviadas por outros usuários através do servidor (Recebe_Notificação).

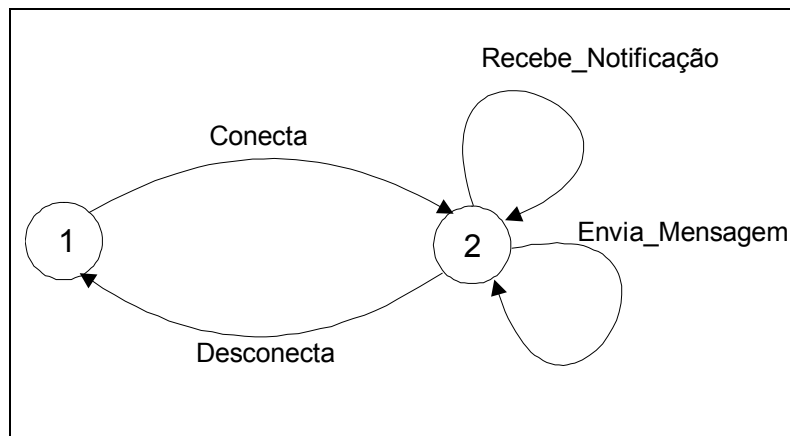


Figura 13: Transição de estados em objeto ClienteSessão

PASSO 4: PROJETO E IMPLEMENTAÇÃO DAS FERRAMENTAS DE SUPORTE À SESSÃO:

O *DocChat* possui dois módulos: cliente e servidor. O módulo servidor é responsável por gerenciar as sessões abertas e o acesso às mesmas. O módulo cliente é utilizado pelo usuário para acessar uma ou mais sessões.

O módulo cliente foi implementado como uma *Applet* Java e pode ser executado a partir de um navegador da Internet, como o HotJava ou o Netscape. A primeira tarefa executada pela Applet é localizar o servidor e buscar a lista das sessões abertas. O usuário pode se conectar a qualquer uma das sessões abertas (desde que essas sessões sejam públicas ou que o usuário esteja incluído na lista de participantes das sessões privadas). Para isso, a Applet cria um objeto ClienteSessão para cada sessão a que o usuário desejar se conectar. O objeto ClienteSessão se comunica diretamente com o objeto Sessão criado no servidor usando Java RMI. A Figura 14 mostra a tela que dá acesso a uma sessão.

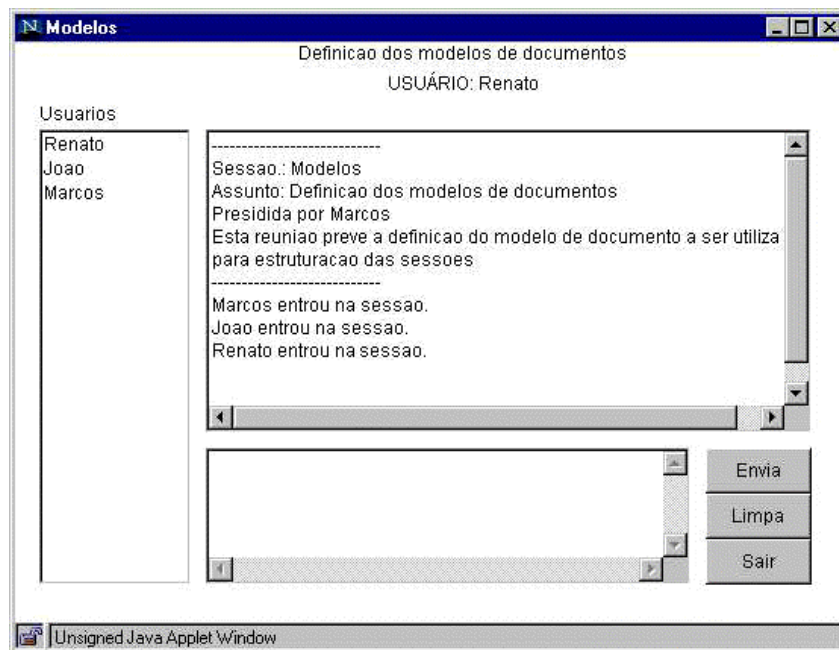


Figura 14: Tela de acesso a uma sessão

Já o módulo servidor tem como função atender às solicitações de conexão dos clientes. Quando detecta uma solicitação, o servidor envia a relação de sessões disponíveis nas quais o cliente pode entrar (Figura 15). O servidor habilita também o **Gerenciador de Sessões** para o cliente. Assim, ao usuário que executa o módulo cliente é permitido criar uma nova sessão, encerrar outra em andamento, desde que satisfeitos os requisitos de segurança, ou, ainda, gerar um relatório da sessão em HTML.

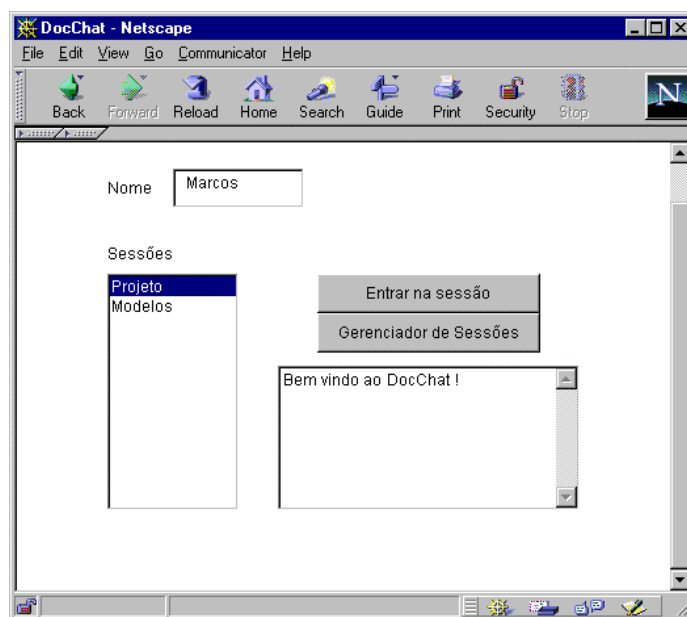
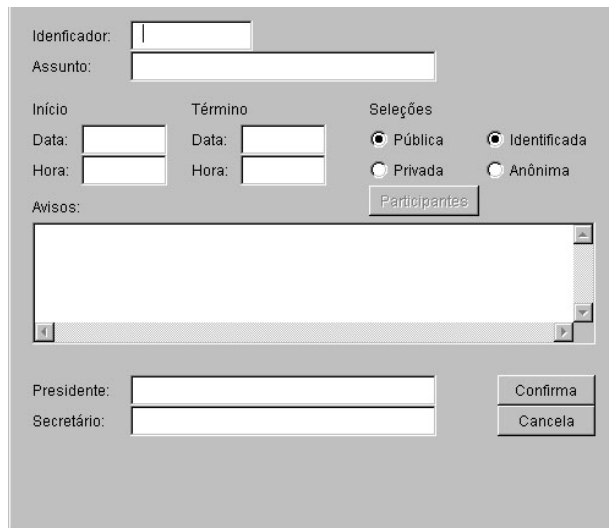


Figura 15: Tela de acesso ao gerenciador de sessões

Ao criar uma nova sessão, o usuário deve informar alguns atributos da sessão, como ilustrado na Figura 16: um identificador; o assunto; a data e hora previstos para início e fim da sessão; nomes do presidente e do secretário. Caso a sessão seja privada, devem ser informados os usuários permitidos. Finalmente, deve ser especificado se a sessão permite mensagens anônimas.



A interface de usuário para a criação de uma sessão, exibida em um estilo de janela de aplicativo. O formulário contém os seguintes campos e controles:

- Identificador: campo de texto.
- Assunto: campo de texto.
- Início: Data e Hora (dois campos).
- Término: Data e Hora (dois campos).
- Seleções: grupo de botões de opção com as opções: Pública (selecionada), Privada, Identificada (selecionada) e Anônima.
- Avisos: uma área de texto grande com uma barra de rolagem, precedida por um botão "Participantes".
- Presidente: campo de texto.
- Secretário: campo de texto.
- Botões "Confirma" e "Cancela" no canto inferior direito.

Figura 16: Tela de criação de uma sessão

PASSO 5: PROJETO E IMPLEMENTAÇÃO DE FERRAMENTAS DE EXPLORAÇÃO DOS DOCUMENTOS

Para cada sessão gerada, um documento na linguagem ChatML é criado. O documento é atualizado de acordo com o andamento da sessão. Tomando como base o conteúdo desse documento, o DocChat pode gerar uma página HTML deixando-a disponível em um servidor da WWW. Assim, as mensagens trocadas em uma sessão podem ser visualizadas diretamente através de um navegador HTML.

4.4 CONSIDERAÇÕES FINAIS

Tendo como ponto de partida as discussões levantadas por Streitz no painel da Hypertext'91, a metodologia proposta por Pimentel et al procura auxiliar na definição das estruturas dos hiperdocumentos a serem utilizados como suporte pelas sessões de aplicações CSCW, definindo uma seqüência de passos e os resultados que devem ser obtidos de cada passo. A aplicação *DocChat* foi desenvolvida com o objetivo de validar a metodologia.

O ambiente DocConf, apresentado nos próximos dois capítulos, teve como infra-estrutura teórica e prática as experiências adquiridas pelo autor desta dissertação no desenvolvimento da metodologia CSCW-SH e na construção do DocChat. De fato, o DocConf pode ser visto como uma evolução do DocChat, ao qual novas funcionalidades foram adicionadas. Por outro lado, o DocConf utiliza uma versão atualizada da CSCW-SH, a qual define o uso da metalinguagem XML, mais apropriada para a WWW, na especificação dos hiperdocumentos suportados.

5 DOCCONF: ESPECIFICAÇÃO E PROJETO

5.1 CONSIDERAÇÕES INICIAIS

Este capítulo apresenta a especificação e o projeto de um ambiente de trabalho cooperativo – DocConf – que explora a união das tecnologias de hipermídia e de CSCW, conforme proposto pela metodologia CSCW-SH. O ambiente DocConf é uma evolução da aplicação DocChat, que suporta sessões de discussões estruturadas de acordo com um DTD SGML. O DocConf adiciona ao DocChat novas funcionalidades ou formas de interação entre os participantes, implementadas em módulos denominados componentes. O uso da metodologia CSCW-SH também orienta o projeto de novos componentes.

Como o DocChat, o DocConf utiliza a Internet como plataforma de comunicação. Sua operação pode ser feita tanto através de uma aplicação cliente quanto através de uma Applet. Esta última permite que qualquer usuário, através de um navegador compatível com Applets Java, se torne um cliente do ambiente.

O DocConf adota a metalinguagem XML para estruturar os documentos gerados pelas sessões ao invés da SGML utilizada no DocChat, modificação essa também realizada na metodologia CSCW-SH.

5.2 CARACTERÍSTICAS GERAIS DO AMBIENTE DOCCONF

O DocConf é um ambiente de CSCW extensível que permite a configuração e execução de sessões de trabalho cooperativo, utilizando a Internet como plataforma de comunicação. Para isso, fornece recursos de *chat*, *whiteboard* e votação que permitem a troca de diferentes tipos de mensagens entre membros de um grupo.

O DocConf foi desenvolvido com dois objetivos básicos: fornecer um ambiente de trabalho cooperativo que utilize hiperdocumentos para formalizar as sessões e ser extensível para permitir que novos recursos ou funcionalidades sejam integrados à medida que haja necessidade.

A metáfora de salas, utilizada pelos diversos sistemas de chat e IRC comuns na Internet, também foi adotada para o DocConf. Nesse projeto, entretanto, foi utilizado o termo **sessão**. Uma sessão é um ambiente ao qual pessoas podem se conectar para interagir com os demais membros de seu grupo de trabalho. Uma vez conectado a uma sessão, o participante escolhe quais recursos serão necessários para as atividades do grupo.

O DocConf utiliza uma arquitetura de comunicação baseada em componentes. Cada componente fornece um recurso ou funcionalidade para uma sessão de trabalho cooperativo. À medida que os componentes são solicitados, eles se registram junto à sessão e são apresentados para o usuário em uma janela interna da janela da sessão (Figura 21). A arquitetura de componentes também foi estudada por Greenberg e Roseman [97].

5.3 USANDO CSCW-SH NO PROJETO DAS SESSÕES

As sessões do DocConf foram desenvolvidas de acordo com a metodologia CSCW-SH conforme detalhado a seguir. Deve-se ressaltar aqui que vários ciclos da metodologia foram necessários para se obterem os resultados apresentados. Durante a apresentação, o termo "componente" é utilizado para fazer referência ao módulo de software que fornece um recurso ou uma funcionalidade para a sessão. Para melhor clareza na apresentação, o projeto dos componentes é descrito na Seção 5.4.

5.3.1 Levantamento dos requisitos das sessões

Para a versão atual do DocConf, foram especificados os seguintes requisitos:

- Suporte a sessões síncronas e assíncronas — O DocConf opera tanto de forma síncrona quanto assíncrona simultaneamente: quando dois ou mais usuários estão conectados em uma sessão, no mesmo intervalo de tempo, trocando informações, o DocConf estará dando suporte a uma sessão síncrona e ao armazenar as informações trocadas por esses participantes e exibi-las para outros que venham a se conectar mais tarde, o DocConf estará dando suporte a sessões assíncronas.
- Sessões públicas ou privadas — Como em algumas sessões podem ser trocadas informações de conteúdo reservado, o DocConf permite que essas sessões sejam privadas. A escolha entre pública e privada deve ser feita durante a configuração da sessão, quando também deverão ser informados os usuários que serão permitidos.

- Definição de um moderador — O moderador é responsável por manter a sessão funcionando e por controlar, até certo ponto, as informações trocadas entre os participantes. O moderador deve evitar, por exemplo, que a sessão se desvie de seu objetivo. A escolha de um moderador também é feita na tela de configuração da sessão.
- Percepção de presença - Os usuários conectados a uma sessão são listados pelo componente USUÁRIOS. Assim, alguém que se conecta pode identificar cada um dos demais participantes. O componente USUÁRIOS será detalhado mais adiante neste capítulo.
- Percepção das ações do grupo — Todas as ações executadas por um participante da sessão são imediatamente transmitidas aos demais.
- Armazenamento dos dados relativos às sessões — O DocConf cria e mantém um documento XML para cada uma das sessões que executa. Esse documento registra não só os dados de configuração como todas as informações trocadas pelos componentes dentro da sessão. Ao seguir a XML, o documento se torna portátil, podendo ser processado por outra ferramenta ou, ainda, apresentado em uma página da WWW.
- Independência de plataforma — Ao ser implementado com a linguagem Java e ao armazenar as informações em documentos XML, o DocConf segue a idéia de código portátil e dados portáteis (*portable code, portable data*), discutida por inúmeros autores como, [Bosak 97; Morgenthal 99; Johnson 99].
- Início e término de funcionalidades a qualquer momento — Um componente do DocConf pode ser iniciado a qualquer momento, mesmo que já exista uma outra instância ativa desse componente. Quando um componente não se fizer mais necessário, poderá ser desativado (pelo moderador), mas as informações processadas serão mantidas para posterior consulta. Esse componente poderá ser reativado a qualquer momento.
- Suporte a novos tipos de interação — Os componentes existentes na versão atual do DocConf implementam algumas funcionalidades básicas de sistemas de CSCW, mas novas funcionalidades podem ser implementadas à medida que forem necessárias. Cada novo componente também deve ser desenvolvido de acordo com a metodologia CSCW-SH.

O DocConf foi projetado de forma que as sessões não dependam dos componentes. Cada sessão pode incorporar apenas os componentes que lhe convierem. Portanto, os requisitos destes não foram incluídos nesta lista, mas são discutidos na Seção 5.4.

5.3.2 Definição de uma estrutura formal para as sessões

A Figura 17 apresenta a parte do DTD para um documento que contém as informações de configuração e manutenção de uma sessão. O restante do DTD está relacionado aos componentes e será apresentado junto com esses na Seção 5.4.

Todo documento é composto pelos elementos <head> e <body>. O elemento <head> é utilizado para armazenar informações relativas à configuração da sessão. Essas informações são o identificador da sessão (atributo <id>), o nome da sessão (<name>), seu moderador (<moderator>), a data e a hora de início (<start>) e fim (<end>) e se a sessão é pública ou privada (atributo <public>). Caso seja privada, há a necessidade da especificação dos usuários permitidos na sessão através do elemento <users>.

Dentro do elemento <body> estão os eventos (início, ativação e desativação) e as mensagens geradas pelos componentes. Todo elemento armazenado dentro de <body> tem dois atributos básicos: <compid> e <date>. O primeiro é utilizado como identificador do componente que gerou a mensagem (ou evento). Já o segundo serve para informar a data e a hora em que a mensagem foi gerada.

Apesar de cada componente possuir seu próprio conjunto de elementos, a parte do DTD referente à sessão contém elementos para registrar os eventos desses componentes. É o que acontece quando um novo componente é iniciado e é escrito no documento um elemento <cstart>, que serve para informar o tipo e o identificador do novo componente. Cada componente criado pode conter uma série de parâmetros que ditarão seu funcionamento e/ou sua forma de exibição. Esses parâmetros são informados através de elementos <param>, que contêm o nome (<name>) e o valor (<value>) de cada parâmetro.

Outros eventos possíveis dos componentes são sua desativação e posterior reativação. Quando estiverem desativados, os comandos se tornam inacessíveis, apesar da tela continuar visível. Esses eventos, realizados pelo moderador da sessão, são registradas no documento através dos elementos <cenable> e <cdisable>.

<pre> <!-- DTD FOR THE DOCCONF SESSION --> <!ELEMENT SESSION (HEAD,BODY) > <!ATTLIST SESSION ID ID #REQUIRED DATE CDATA #REQUIRED PUBLIC (YES NO) 'YES' > <!ELEMENT HEAD (NAME, MODERATOR, USERS?, START?, END?) > <!ELEMENT NAME (#PCDATA) > <!ELEMENT MODERATOR (#PCDATA) > <!ELEMENT USERS (USER+) > <!ELEMENT USER (#PCDATA) > <!ELEMENT START EMPTY > <!ATTLIST START DATE CDATA #REQUIRED > <!ELEMENT END EMPTY > <!ATTLIST END DATE CDATA #REQUIRED > </pre>	<pre> <!ELEMENT BODY (LOGIN LOGOUT CSTART CENABLE CDISABLE CHAT WHITEBOARD VOTE) * > <!ELEMENT CSTART (PARAM*) > <!ATTLIST CSTART COMPID CDATA #REQUIRED DATE CDATA #REQUIRED TYPE CDATA #REQUIRED > <!ELEMENT PARAM EMPTY > <!ATTLIST PARAM NAME CDATA #REQUIRED VALUE CDATA #REQUIRED > <!ELEMENT CENABLE EMPTY > <!ATTLIST CENABLE COMPID CDATA #REQUIRED DATE CDATA #REQUIRED > <!ELEMENT CDISABLE EMPTY > <!ATTLIST CDISABLE COMPID CDATA #REQUIRED DATE CDATA #REQUIRED > </pre>
--	--

Figura 17: Parte do DTD para o documento de uma sessão

5.3.3 Definição da arquitetura de comunicação cliente/servidor

A comunicação entre os participantes de uma sessão é feita exclusivamente através dos componentes e cada um destes, por implementar um tipo diferente de interação entre os participantes, possui um tipo diferente de mensagem. O chat, por exemplo, envia mensagens textuais para os demais participantes. Já o whiteboard envia a representação vetorial de objetos gráficos (retângulos, linhas, elipses, etc.).

Os componentes são implementados exclusivamente do lado do cliente, não havendo contrapartida do lado do servidor. Dessa forma, o servidor é utilizado apenas para repassar as mensagens aos outros clientes. O servidor executa, porém, dois métodos de cada mensagem que retransmite. O primeiro desses métodos é utilizado para registrar a mensagem no hiperdocumento associado à sessão. O segundo é opcional e utilizado apenas se houver necessidade de algum processamento extra no servidor.

A Figura 18 exemplifica o processo de comunicação entre os clientes e o servidor. Nela, existem três usuários (Alice, João e Márcia) conectados a uma sessão identificada por "A". Se Alice digitar uma mensagem textual em sua janela de chat, essa mensagem será passada ao módulo *ClienteSessão* de Alice. Este módulo é responsável pelo gerenciamento dos componentes e pela comunicação entre eles e o servidor. Assim, os componentes não se comunicam diretamente com o servidor, apenas através do *ClienteSessão*.

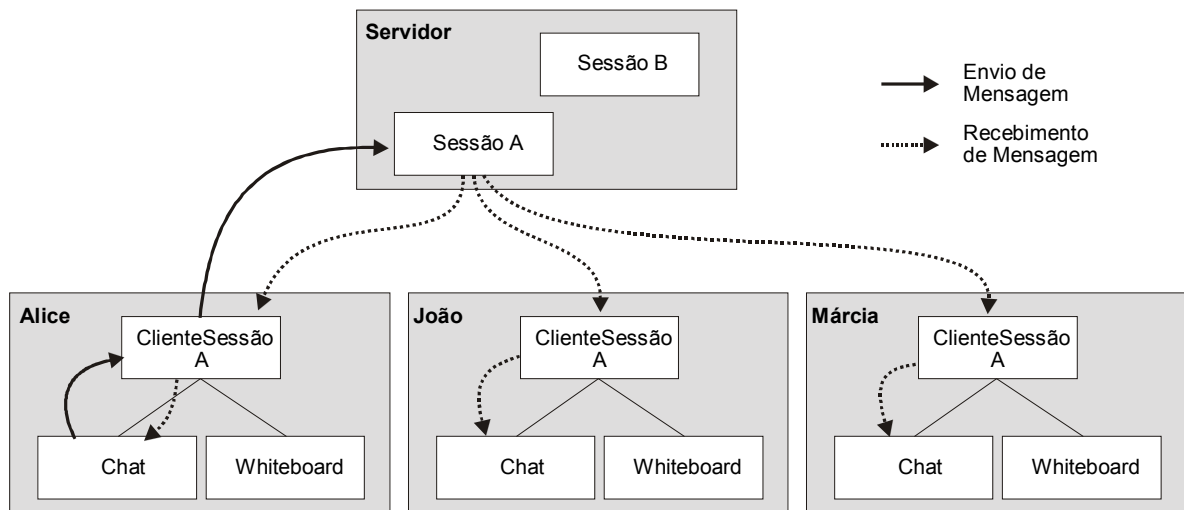


Figura 18: Processo de comunicação entre clientes e servidor

Quando o servidor receber a mensagem vinda de Alice, ele a redistribuirá a todos os módulos *ClienteSessão* com os quais mantém um canal de comunicação (inclusive ao remetente da mensagem, nesse caso Alice). Cada *ClienteSessão* passará, por fim, a mensagem ao seu componente correspondente.

Essa forma de comunicação apresenta grandes benefícios em relação à extensibilidade do DocConf, alcançada através do desenvolvimento de novos componentes. Como nem o servidor nem o módulo *ClienteSessão* necessitam reconhecer o conteúdo da mensagem, o desenvolvedor pode projetar sua aplicação como se esta fosse monousuária, sem se preocupar com os detalhes da comunicação.

O gerenciamento das sessões (criação, conexão e fechamento) fica a cargo de outros dois módulos, que podem ser modificados de acordo com a necessidade da aplicação. Esse gerenciamento pode ser feito de duas formas: os clientes podem gerenciar suas próprias sessões e as sessões podem ser gerenciadas por um terceiro agente (humano ou computacional). No primeiro caso, os próprios clientes podem definir os dados da sessão tais como os participantes, os papéis desses participantes e a duração da sessão. Já no segundo caso, as sessões são pré-configuradas e aos clientes só são permitidas a conexão, a troca de mensagens e a desconexão.

5.3.4 Projeto e implementação das ferramentas de suporte à sessão

O acesso ao ambiente pode ser feito através de uma aplicação cliente (Figura 19a) ou através de uma página da WWW (Figura 19b), utilizando um navegador compatível com Applets Java. Independentemente da forma de acesso, o ambiente oferece ao usuário a mesma interface e os mesmos recursos. A versão atual do DocConf não oferece recursos de segurança de dados, apenas o controle de acesso. Todos os usuários devem se registrar antes de utilizar o ambiente. Novos usuários podem se registrar através do botão CADAстра.

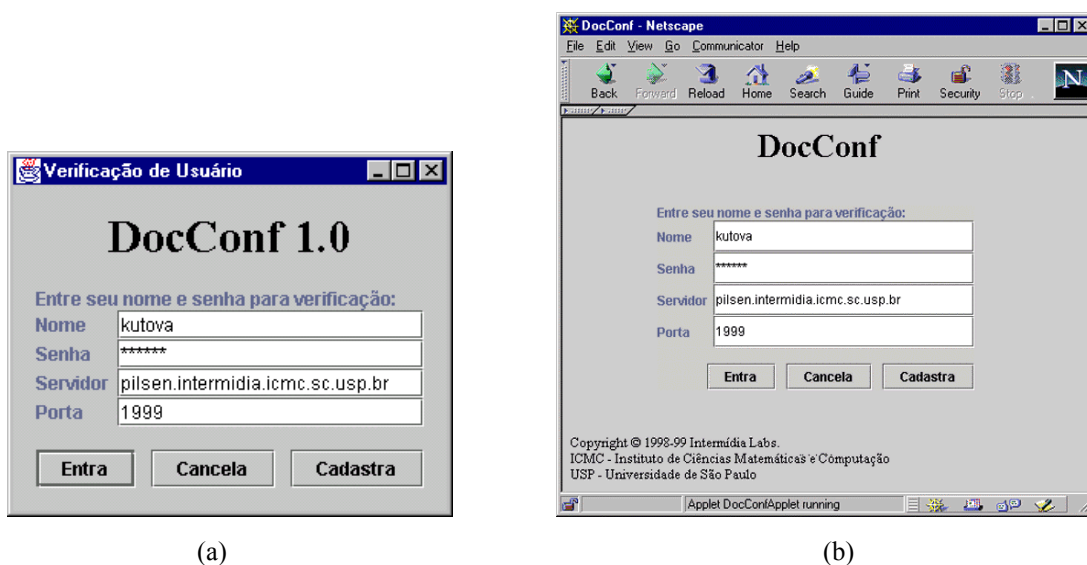


Figura 19: Telas de acesso ao ambiente

Uma vez conectado ao ambiente, o usuário recebe a tela de gerenciamento de sessões (Figura 20). Através dessa tela, ele pode criar uma nova sessão, entrar em uma sessão ativa ou, ainda, fechar uma sessão que esteja vazia. O usuário pode participar de várias sessões simultaneamente, mas não há qualquer troca de informações entre essas sessões, pois elas operam de forma independente.

A tela de gerenciamento de sessões exibe também uma lista dos usuários conectados ao ambiente e permite que cada um altere seus dados de cadastro, consulte os de outro usuário e troque mensagens. Estas mensagens podem ser utilizadas, por exemplo, para convidar alguém a se juntar a uma sessão. Caso um usuário não deseje receber mensagens, ele pode ativar a opção "Não Incomodar", disponível no menu "Usuários".

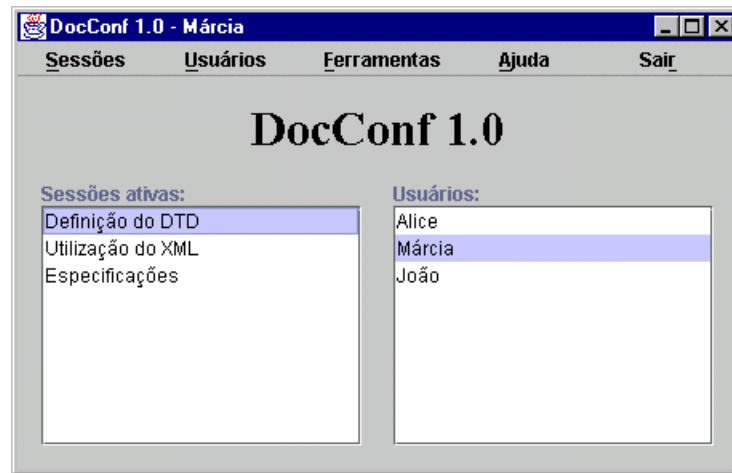


Figura 20: Gerenciador de sessões

A Figura 21 apresenta a tela do cliente de uma sessão. Pela figura, percebem-se três usuários conectados, Márcia, Alice e João, que interagem através dos componentes chat, whiteboard e votação — implementados através de janelas internas à janela da sessão. É importante ressaltar que somente um componente pode estar sendo operado a qualquer momento da sessão. Esse componente terá sua barra de título diferenciada e será desenhado sobre os demais. Para operar outro componente, basta selecioná-lo com o mouse. Assim como o ambiente, cada componente foi criado seguindo a metodologia CSCW-SH, conforme detalhado na Seção 5.4.

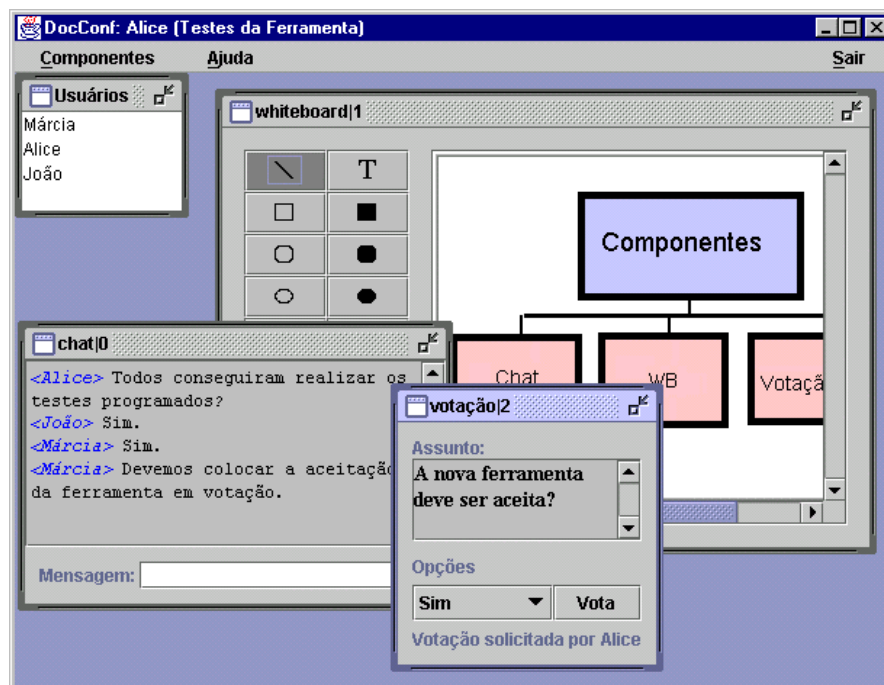


Figura 21: Exemplo de um cliente de sessão

5.3.5 Projeto e implementação de ferramentas de exploração dos documentos

Para permitir o processamento dos documentos gerados em uma sessão, está sendo implementada a integração com o *parser* XML fornecido pela Sun Microsystems. Através dessa integração, aplicações desenvolvidas em Java terão total acesso às informações armazenadas nos documentos.

Também está sendo projetado um documento XSL (*Extensible Stylesheet Language*) para converter o documento gerado (Figura 22) em um documento HTML que possa ser exibido em um navegador da Internet. Para tanto, o navegador deve ser capaz de interpretar documentos XML e XSL.

```
<?XML version="1.0"?>
<SESSION ID="testes" DATE="06/04/99 16:46:45">
<HEAD>
  <NAME>Testes da Ferramenta</NAME>
  <MODERATOR>Alice</MODERATOR>
</HEAD>
<BODY>
<LOGIN DATE="06/04/99 16:46:51">João</LOGIN>
<LOGIN DATE="06/04/99 16:48:58">Márcia</LOGIN>
<LOGIN DATE="06/04/99 16:51:08">Alice</LOGIN>
<CSTART COMPID="chat|0" DATE="06/04/99 16:57:27" TYPE="chat">
  <param name="anonymous" value="false"/>
  <param name="timevisible" value="false"/>
  <param name="linefeed" value="false"/>
</CSTART>
<CHAT COMPID="chat|0" DATE="06/04/99 16:58:11">
  <FROM>Alice</FROM>
  <MESSAGE>Os diagramas estão prontos?</MESSAGE>
</CHAT>
<CHAT COMPID="chat|0" DATE="06/04/99 16:58:37">
  <FROM>João</FROM>
  <MESSAGE>Ainda não. Faltam poucos.</MESSAGE>
</CHAT>
<CHAT COMPID="chat|0" DATE="06/04/99 16:58:58">
  <FROM>Alice</FROM>
  <MESSAGE>E o texto? Já foi revisado?</MESSAGE>
</CHAT>
<CHAT COMPID="chat|0" DATE="06/04/99 16:59:40">
  <FROM>Márcia</FROM>
  <MESSAGE>Sim. Já está sendo impresso.</MESSAGE>
</CHAT>
<CSTART COMPID="votação|1" DATE="06/04/99 17:42:34" TYPE="votação">
  <param name="subject" value="Aprovação do DTD dos novos hiperdocumentos"/>
  <param name="anonymous" value="true"/>
  <param name="option" value="Aprovado"/>
  <param name="option" value="Rejeitado"/>
</CSTART>
<VOTE COMPID="votação|1" DATE="06/04/99 17:43:26">Aprovado</VOTE>
<VOTE COMPID="votação|1" DATE="06/04/99 17:43:32">Rejeitado</VOTE>
<VOTE COMPID="votação|1" DATE="06/04/99 17:43:35">Aprovado</VOTE>
<LOGOUT DATE="06/04/99 17:46:53">Alice</LOGOUT>
<LOGOUT DATE="06/04/99 17:47:00">Márcia</LOGOUT>
<LOGOUT DATE="06/04/99 17:47:06">João</LOGOUT>
</BODY>
</SESSION>
```

Figura 22: Documento gerado por uma sessão de DocConf

5.4 USANDO CSCW-SH NO PROJETO DOS COMPONENTES

Greenberg e Roseman [97] sugerem que os componentes podem ser interpretados como pequenas aplicações de groupware, cada uma implementando uma funcionalidade diferente.

Os componentes de DocConf também foram especificados de acordo com a metodologia CSCW-SH e esta seção os discute porque cada um deles pode ser considerado uma aplicação de CSCW em si. O passo três da metodologia — *definição da arquitetura de comunicação* — foi omitido, pois os componentes seguem a arquitetura imposta pela sessão. O quinto passo — *projeto e implementação de ferramentas de exploração dos documentos* — também não será discutido, pois não foram projetadas ferramentas que possam manipular elementos isolados desses documentos.

Todos os componentes possuem um ou mais elementos que devem ser incorporados ao DTD da Figura 17. Os elementos principais de cada componente possuem dois atributos básicos: o identificador de componente (<compid>), que indica a qual componente pertence aquela informação, e a data e hora do registro da informação (<date>).

5.4.1 Usuários

Este componente indica quais usuários estão participando da sessão. Todos os usuários são identificados através de um "apelido" único, que deve ser informado pelo mesmo usuário no momento em que se cadastrar no ambiente.

Esse componente tem como finalidade, ou requisito, a informação imediata da entrada ou saída de um participante.

A Figura 23 apresenta os elementos que devem ser adicionados ao DTD para esse registro de entrada (<login>) e saída (<logout>). Ambos os elementos requerem o apelido do participante e a data e hora do evento. A Figura 24 mostra a interface projetada para tal componente.

<!ELEMENT	LOGIN	(#PCDATA)	>
<!ATTLIST	LOGIN		
		DATE	CDATA #REQUIRED >
<!ELEMENT	LOGOUT	(#PCDATA)	>
<!ATTLIST	LOGOUT		
		DATE	CDATA #REQUIRED >

Figura 23: Elementos do componente Usuários

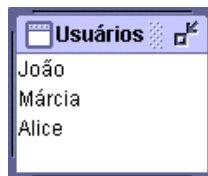


Figura 24: Janela do componente Usuários

5.4.2 Chat

O Chat é utilizado para troca de mensagens textuais entre os participantes e a apresenta como requisito a possibilidade do anonimato. Quando as mensagens não forem anônimas, deverão vir acompanhadas do apelido do usuário.

Os elementos que devem ser adicionados ao DTD para esse componente estão ilustrados na Figura 25. O elemento <chat> engloba todas as informações manipuladas. Essas informações são o apelido do remetente (<from>) e o conteúdo da mensagem (<message>). Se o chat for anônimo, o apelido será omitido.

<!ELEMENT	CHAT	(FROM?, MESSAGE)	>
<!ATTLIST	CHAT		
		COMPID	CDATA #REQUIRED >
		DATE	CDATA #REQUIRED >
<!ELEMENT	FROM	(#PCDATA)	>
<!ELEMENT	MESSAGE	(#PCDATA)	>

Figura 25: Elementos do componente Chat

A Figura 26 e a Figura 27 mostram, respectivamente, as telas de configuração e utilização do componente. Além da possibilidade de anonimato, esse componente permite que sejam registradas as horas em que as mensagens são postadas e a inclusão de uma linha de separação antes de cada nova mensagem.

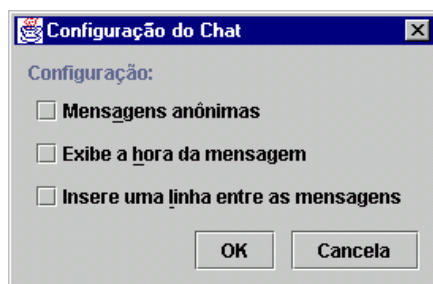


Figura 26: Configuração do componente Chat

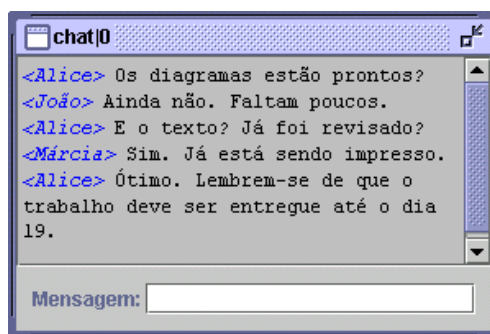


Figura 27: Janela do componente Chat

5.4.3 Whiteboard

Este componente é utilizado para a elaboração de um desenho ou diagrama de modo cooperativo. Os principais requisitos do whiteboard são: (a) permitir que os participantes da sessão desenhem vários tipos de objetos (retângulos, elipses, textos, linhas, etc.); (b) suportar diferentes espessuras de linhas; (c) permitir a definição de cores de linha e de preenchimento.

A Figura 28 mostra os elementos desse componente que devem ser adicionados ao DTD. O elemento `<whiteboard>` engloba as informações manipuladas pelo componente de desenho. Seus objetos estão agrupados em formas vazias (`<shape>`), formas preenchidas (`<filledshape>`), textos (`<text>`) e linhas poligonais (`<polyline>`). De acordo com a necessidade de cada um desses objetos, outros elementos são armazenados para dar várias informações: o tipo do objeto (`<type>`); sua posição na janela (`<coords>`); suas dimensões (`<dimensions>`); a espessura da linha (`<thickness>`); a cor da linha (`<foreground>`); a cor de preenchimento (`<background>`). Textos necessitam ainda da especificação do tamanho da letra (`<fontsize>`) e as linhas poligonais das coordenadas de cada um de seus pontos (`<xcoords>` e `<ycoords>`).

```

<!ELEMENT WHITEBOARD (SHAPE | FILLEDSHAPE
| TEXT | POLYLINE) >
<!ATTLIST WHITEBOARD
    COMPID CDATA #REQUIRED
    DATE CDATA #REQUIRED >
<!ELEMENT SHAPE EMPTY >
<!ATTLIST SHAPE
    TYPE CDATA #REQUIRED
    COORDS CDATA #REQUIRED
    DIMENSIONS CDATA #REQUIRED
    THICKNESS CDATA #REQUIRED
    FOREGROUND CDATA #REQUIRED >
<!ELEMENT FILLEDSHAPE EMPTY >
<!ATTLIST FILLEDSHAPE
    TYPE CDATA #REQUIRED
    COORDS CDATA #REQUIRED
    DIMENSIONS CDATA #REQUIRED
    THICKNESS CDATA #REQUIRED
    FOREGROUND CDATA #REQUIRED
    BACKGROUND CDATA #REQUIRED >
<!ELEMENT TEXT (#PCDATA) >
<!ATTLIST TEXT
    COORDS CDATA #REQUIRED
    FONTSIZE CDATA #REQUIRED
    FOREGROUND CDATA #REQUIRED >
<!ELEMENT POLYLINE EMPTY >
<!ATTLIST POLYLINE
    POINTS CDATA #REQUIRED
    XCOORDS CDATA #REQUIRED
    YCOORDS CDATA #REQUIRED
    THICKNESS CDATA #REQUIRED
    FOREGROUND CDATA #REQUIRED >

```

Figura 28: Elementos do componente WhiteBoard

A Figura 29 mostra a janela do whiteboard. Apesar de permitir o desenho de vários objetos diferentes, esse componente ainda não permite a alteração, exclusão ou reposicionamento dos objetos.

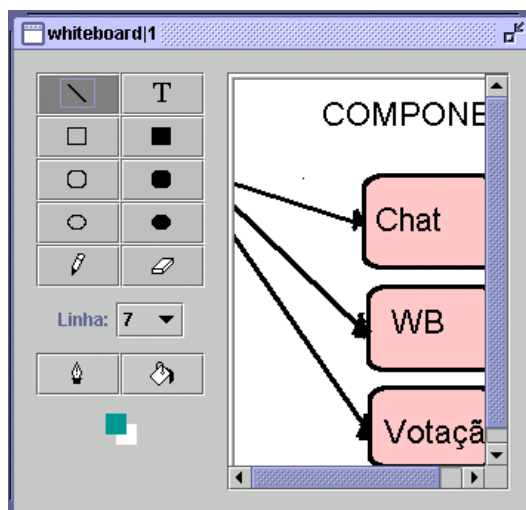


Figura 29: Janela do componente Whiteboard

5.4.4 Votação

O componente Votação tem como requisito permitir que os participantes de uma sessão votem positiva ou negativamente em uma determinada proposta ou escolham uma entre várias opções disponibilizadas (pelo moderador). Os votos são anônimos.

Os votos são registrados no documento através do elemento <vote>, ilustrado na Figura 30.

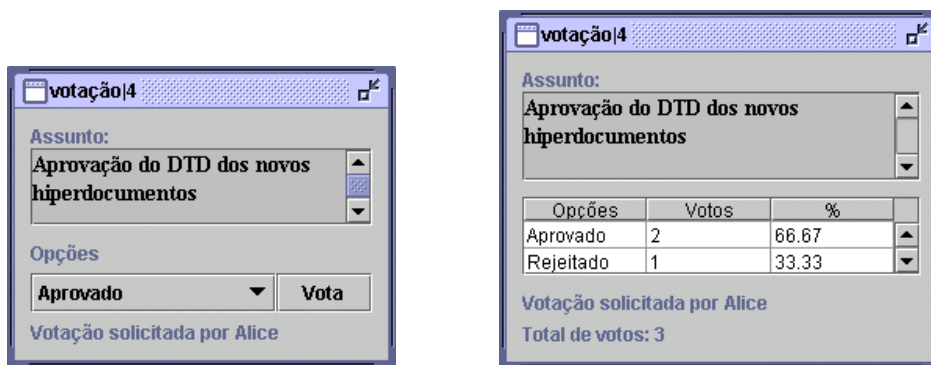
```

<!ELEMENT VOTE (#PCDATA) >
<!ATTLIST VOTE
            COMPID CDATA #REQUIRED
            DATE CDATA #REQUIRED >

```

Figura 30: Elementos do componente Votação

Dada a funcionalidade exibida, uma votação é realizada em dois passos. Quando é iniciada, é apresentada uma tela a cada participante com o tema proposto e as opções de voto possíveis (Figura 31a). Logo após o usuário votar, essa tela é alterada para exibir os resultados da votação (Figura 31b).



(a)

(b)

Figura 31: Componente "Votação"

5.5 TRABALHOS RELACIONADOS

O modelo de componentes utilizado no DocConf também foi adotado por Greenberg e Roseman no ambiente TeamWave [Greenberg & Roseman 97], baseado em Tcl/Tk e no toolkit de groupware Groupkit [Greenberg & Roseman 96]. O TeamWave, desenvolvido na Universidade de Calgary, utiliza a metáfora de salas (análogas às sessões do DocConf), para integrar as atividades e ferramentas de um grupo de trabalho e, mesmo oferecendo um vasto conjunto de componentes ou ferramentas para colaboração, tem como preocupação permitir o desenvolvimento de novos componentes de forma fácil e intuitiva.

Criado pelo NCSA – *National Center for Supercomputing Applications*, Universidade de Illinois, o Habanero [Chabert et al 98] fornece uma plataforma para colaboração através da Internet, especialmente nas áreas de educação e ciências. Apesar de ser um ambiente que implementa inúmeras funcionalidades, o Habanero também permite o desenvolvimento de novas aplicações cooperativas que atendam a necessidades mais específicas. Esse ambiente oferece um vasto conjunto de ferramentas para colaboração e está sendo estendido para permitir sessões assíncronas.

Dourish [98], ao projetar o *toolkit* Prospero, teve como principal preocupação a flexibilidade, ou seja, a gama de aplicações que pode suportar. De acordo com o autor, um *toolkit* deve ser flexível o suficiente para permitir as diversas formas de interação que podem ocorrer entre pessoas, quando essas trabalham em grupo. Para alcançar tal flexibilidade, Prospero utiliza uma nova abordagem arquitetural – *Open Implementation* – onde o programador não apenas

especifica como a aplicação utilizará a infra-estrutura do *toolkit*, como também pode examinar ou modificar a implementação dessa infra-estrutura.

5.6 CONSIDERAÇÕES FINAIS

O DocConf, evolução da aplicação DocChat apresentada no Capítulo 4, procura investigar o uso de hiperdocumentos estruturados no suporte ao trabalho cooperativo, utilizando a Internet como plataforma de comunicação. Para tanto, o ambiente permite diversas formas de interação entre as pessoas de um grupo de trabalho. Cada uma dessas interações foi implementada em um módulo denominado componente. Novos componentes podem ser desenvolvidos de acordo com a necessidade, o que configura o ambiente como extensível.

Ao ser desenvolvido de acordo com a metodologia CSCW-SH, o DocConf tem como requisito registrar todas as informações trocadas durante uma sessão em um documento XML. Os documentos gerados podem depois ser consultados, manipulados ou intercambiados por outros sistemas, sejam eles computacionais ou humanos.

6

DOCCONF: IMPLEMENTAÇÃO, EXTENSÃO E REUSO

6.1 CONSIDERAÇÕES INICIAIS

De acordo com Paul Dourish [98], uma boa ferramenta de trabalho cooperativo deve possuir flexibilidade suficiente para permitir as diversas formas de interação que podem ocorrer entre membros de um grupo. O DocConf procura alcançar essa flexibilidade permitindo que novos componentes sejam desenvolvidos de forma fácil e direta, a fim de possibilitar outras formas de interação.

Para isto, todos os detalhes que não tratam especificamente da funcionalidade de um componente, como a arquitetura de comunicação, o controle de acesso de usuários, o gerenciamento de sessões entre outros, foram retirados das implementações e incorporados em outros módulos, os quais também podem ser modificados de acordo com a necessidade da aplicação. Dessa forma, o desenvolvedor pode projetar seu componente como se fosse uma aplicação monousuário e se concentrar apenas na funcionalidade almejada.

Esse capítulo apresenta, inicialmente, aspectos da implementação do DocConf em três visões distintas: (1) o ambiente, que controla o acesso de usuários e a manutenção das sessões; (2) a sessão, que gerencia os componentes utilizados para a comunicação entre os participantes e (3) os componentes propriamente ditos. Nas demais seções, são discutidos aspectos da extensão do DocConf, com a apresentação de um roteiro para o desenvolvimento de novos componentes e, baseando-se em um exemplo, como o DocConf pode ser facilmente adaptado para integração a projetos que utilizem outra forma de comunicação ou de gerenciamento de sessões.

6.2 **DOCCONF: IMPLEMENTAÇÃO**

6.2.1 **O Ambiente**

O DocConf foi desenvolvido utilizando a linguagem Java da Sun Microsystems, cujos principais benefícios são a independência de plataforma e o amplo suporte existente na WWW. Para o projeto de interface gráfica, foi utilizada a API Swing, que garante uma interface consistente entre plataformas. A comunicação é feita através de *sockets* TCP/IP estabelecidos entre clientes e servidor.

O DocConf pode ser utilizado como uma aplicação ou como uma Applet (Figura 19). O acesso ao ambiente através de uma Applet evita a necessidade da distribuição de módulos de cliente aos usuários, uma vez que os navegadores da WWW passam, automaticamente, a funcionar como clientes. As Applets, entretanto, impõem severas restrições de segurança, tornando complicada, quando não impossível, a implementação de alguns recursos úteis às ferramentas de trabalho cooperativo.

Uma vez dentro ambiente, o usuário não faz distinção entre as formas de acesso: aplicação ou Applet. As classes do sistema mantêm, porém, uma variável que indica qual a forma utilizada. Essa variável é necessária para permitir o acesso a imagens ou documentos disponibilizados na WWW.

Após o usuário entrar no ambiente, ele recebe a tela de gerenciamento de sessões (Figura 20). Por meio dessa tela, ele pode criar uma nova sessão, entrar em uma sessão ativa ou, ainda, fechar uma sessão, desde que esteja vazia. Essa tela é implementada através da classe `Cliente`. Um objeto dessa classe se comunica diretamente com um objeto `ThreadServidor` no servidor. A Figura 32 mostra a comunicação entre esses objetos. As informações trocadas entre o cliente e o servidor são objetos da classe `Mensagem`, os quais possuem um identificador de tipo de operação a ser realizada (por exemplo, criação, conexão e fechamento de sessão) e os parâmetros necessários a essa operação.

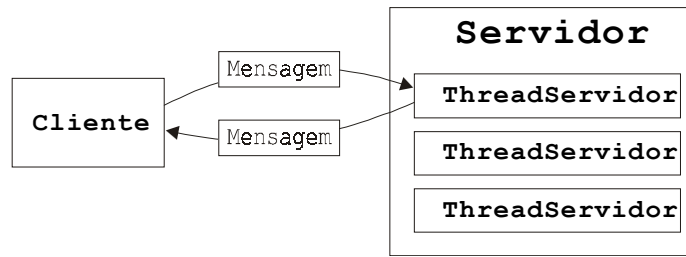


Figura 32: Comunicação entre objetos no cliente e no servidor

6.2.2 A Sessão

Para cada sessão ativa no servidor, existe um objeto da classe `Sessão` correspondente. Esse objeto é responsável pelo gerenciamento das mensagens recebidas dos clientes. Os canais de comunicação entre os clientes e o objeto `Sessão` são mantidos, do lado do servidor, por objetos `ThreadSessão`, que só tratam do recebimento e envio de mensagens (instâncias da classe `MensagemSessão`). O processamento destas é realizado pelo próprio objeto `Sessão`. A Figura 33 ilustra esse processo de comunicação. Do lado do cliente, toda a comunicação do é mantida através de um objeto `ClienteSessão`, que também é responsável pela exibição da janela do cliente (Figura 21).

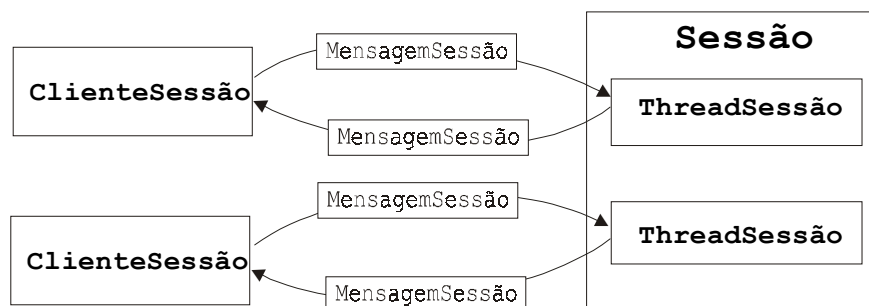


Figura 33: Comunicação entre uma sessão e seus clientes

6.2.3 Os Componentes

As interações entre clientes conectados a uma sessão são feitas exclusivamente através dos componentes. Um componente XYZ (chat, whiteboard, etc.) envolve duas classes: `ClienteComponenteXYZ` e `MensagemComponenteXYZ`. Essas classes implementam as

interfaces³ `ClienteComponenteGenérico` e `MensagemComponenteGenérico` respectivamente, ilustradas na Figura 34 e Figura 35.

A classe `ClienteComponenteXYZ` é utilizada como interface com o usuário. Ela é responsável pelo desenho da tela, pela leitura de entradas de teclado e mouse e pela exibição das mensagens vindas do servidor. A classe `MensagemComponenteXYZ` é utilizada para encapsular as informações produzidas por um componente `XYZ`, que serão enviadas aos demais usuários.

```
public interface ClienteComponenteGenerico {  
  
    public void recebe( Object objeto, String horas );  
    public String tipo();  
    public boolean ativa();  
    public void desativa( boolean h );  
  
}
```

Figura 34: Interface `ClienteComponenteGenérico`

```
import java.util.Calendar;  
  
public interface MensagemComponenteGenerico {  
  
    public String registra( Calendar dataAtual );  
    public String processa();  
  
}
```

Figura 35: Interface `MensagemComponenteGenérico`

Apesar de as interações entre usuários serem feitas exclusivamente através dos componentes, estes não possuem um canal de comunicação com o servidor. Suas mensagens são encapsuladas dentro de um objeto da classe `MensagemSessão`, que é enviado ao servidor pelo objeto `ClienteSessão` (discutido na seção anterior).

Os componentes são implementados exclusivamente do lado do cliente. Assim, o objeto `Sessão`, implementado no servidor, não reconhece o conteúdo das mensagens, apenas as repassa aos clientes. Mas ele executa dois métodos de cada mensagem - `processa()` e `registra()` - especificados na interface `MensagemComponenteGenérico`. O primeiro método é utilizado para algum processamento especial. Já o segundo, serve para registrar a mensagem no

³ Uma **interface** é utilizada em Java para descrever os métodos que uma determina classe deve implementar. Todos

documento associado à sessão. Mesmo que nenhum processamento ou registro seja necessário, esses objetos devem, obrigatoriamente, existir.

Quando o cliente de uma sessão – um objeto da classe `ClienteSessão` – recebe uma mensagem, ele a passa ao componente correto. A lista de componentes ativos é mantida por esse objeto em uma tabela hash. Como as mensagens carregam um identificador de componente, este pode ser localizado facilmente na lista. O componente que receber a mensagem fica encarregado de realizar todo o processamento necessário.

6.3 DOCCONF: EXTENSÃO COM NOVOS COMPONENTES

Na maioria dos trabalhos relacionados, os autores afirmam que o sucesso de um ambiente de trabalho cooperativo está na possibilidade de desenvolvimento de novos módulos que implementem novas formas de interação entre os membros do grupo [Greenberg & Roseman 97; Chabert et al 98; Dourish 98].

O desenvolvimento de outros componentes através de DocConf envolve duas etapas: especificação e implementação. A etapa de especificação, abordada no capítulo anterior, deve ser feita de acordo com a metodologia CSCW-SH. Já a de implementação, deve resultar nas seguintes classes:

1. Mensagem - Deve-se especificar quais os tipos dos dados da mensagem trocada entre os clientes deste componente. Para o componente Chat, por exemplo, são trocadas Strings. Já para o componente Votação, podem ser trocados números inteiros, que indiquem o índice do voto no vetor de votos. Outras informações, como o identificador do componente e o apelido do remetente, devem ser obrigatoriamente incluídas na mensagem. Esta deve implementar as interfaces `MensagemComponenteGenérico` e `Serializable`, sendo a última a responsável pela serialização do objeto, para que ele possa ser enviado através de um *socket* TCP/IP.
2. Configuração - Deve-se projetar uma classe que seja responsável por obter, do usuário ou do sistema, as configurações necessárias ao componente. Estas devem retornar uma tabela hash

os métodos descritos na interface são abstratos.

contendo os pares de nome e valor de cada um dos parâmetros. Caso o componente não necessite de configuração, não há necessidade da implementação desta classe –cria-se apenas uma tabela hash vazia.

3. Interface e Processamento - Finalmente, cria-se a classe responsável pela interface e processamento das mensagens recebidas e que deve implementar a interface `ClienteComponenteGenérico`. Para permitir que o componente seja exibido em uma janela interna à da sessão, essa classe deve estender a classe `JInternalFrame`, do pacote `Swing`. Finalmente, os seguintes métodos básicos devem ser implementados:

- `Construtor` Ajusta as variáveis dos componentes com base nos parâmetros recebidos e exibe a tela inicial.
- `actionPerformed()` Interpreta os eventos dos elementos da tela, como botões, listas e barras de rolagem. As mensagens normalmente são geradas e enviadas após esses eventos.
- `recebe()` Recebe as mensagens do servidor e as processa/exibe.
- `tipo()` Informa o tipo do componente (chat, whiteboard, etc.).
- `ativa()` Ativa o componente.
- `desativa()` Desativa o componente.

Componentes mais sofisticados podem necessitar de outros métodos ou classes.

6.4 DOCCONF: REUSO

O `StudyConf` [Macedo et al 99b] é um ambiente cooperativo que visa a apoiar o estudo e a discussão em grupo de hiperdocumentos didáticos, disponibilizados na WWW. O ambiente utiliza tecnologias como HTML, JavaScript, Java e JDBC para monitorar as atividades dos usuários e tem sua modelagem conceitual baseada na técnica MCLAI — Modelagem Conceitual e Lógica de Aplicações para a Internet [Macedo et al 99a].

O `StudyConf` mantém um controle sobre um conjunto de disciplinas cada uma dividida em tópicos e sobre os alunos nelas matriculados. Os tópicos são compostos por hiperdocumentos, que devem ser estudados pelos alunos. Opcionalmente, o aluno só pode avançar para outro

tópico quando tiver concluído questionários associados ao tópico corrente. A Figura 36 mostra o diagrama de classes do StudyConf.

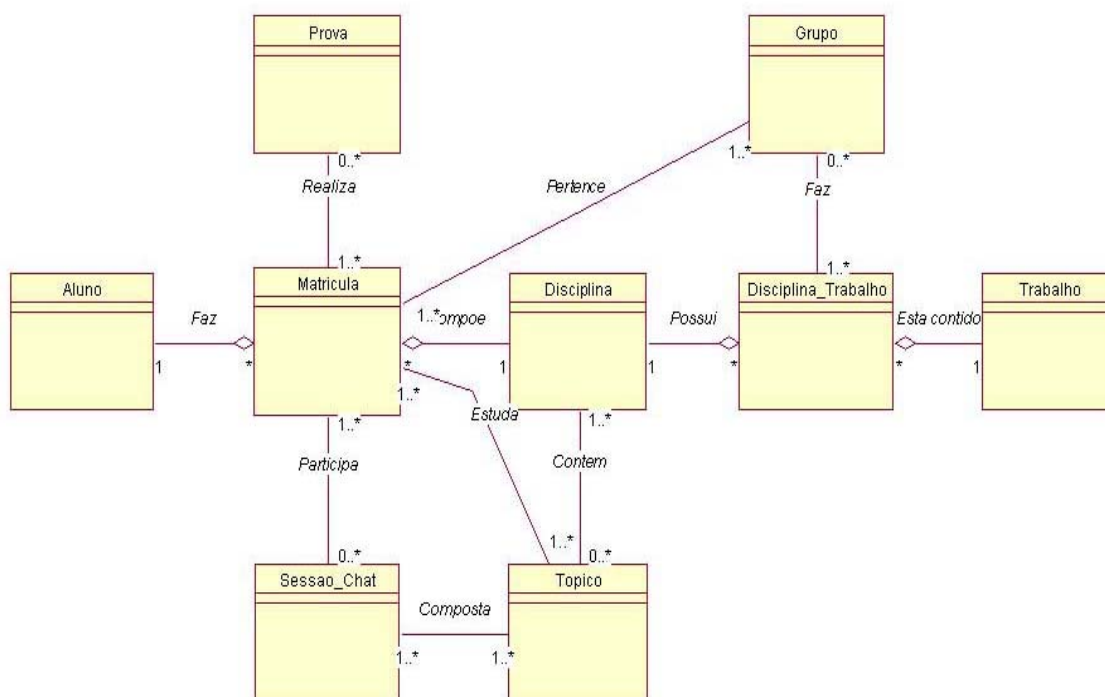


Figura 36: Diagrama de classes do StudyConf [Macedo et al 99a]

Para permitir a interação entre os alunos que estudam um determinado tópico, o StudyConf utiliza o DocConf (Figura 37). Entretanto, como os alunos só podem entrar em sessões vinculadas aos tópicos que estudam, todo o gerenciamento das sessões é feito pelo próprio StudyConf. Para isso, necessitou-se apenas da adaptação das classes `Servidor` e `Cliente` do DocConf. Todo o restante do DocConf foi reutilizado na íntegra. Os documentos gerados pelas discussões dos alunos sobre um tópico ficam armazenados no servidor e disponíveis para consulta ou processamento.

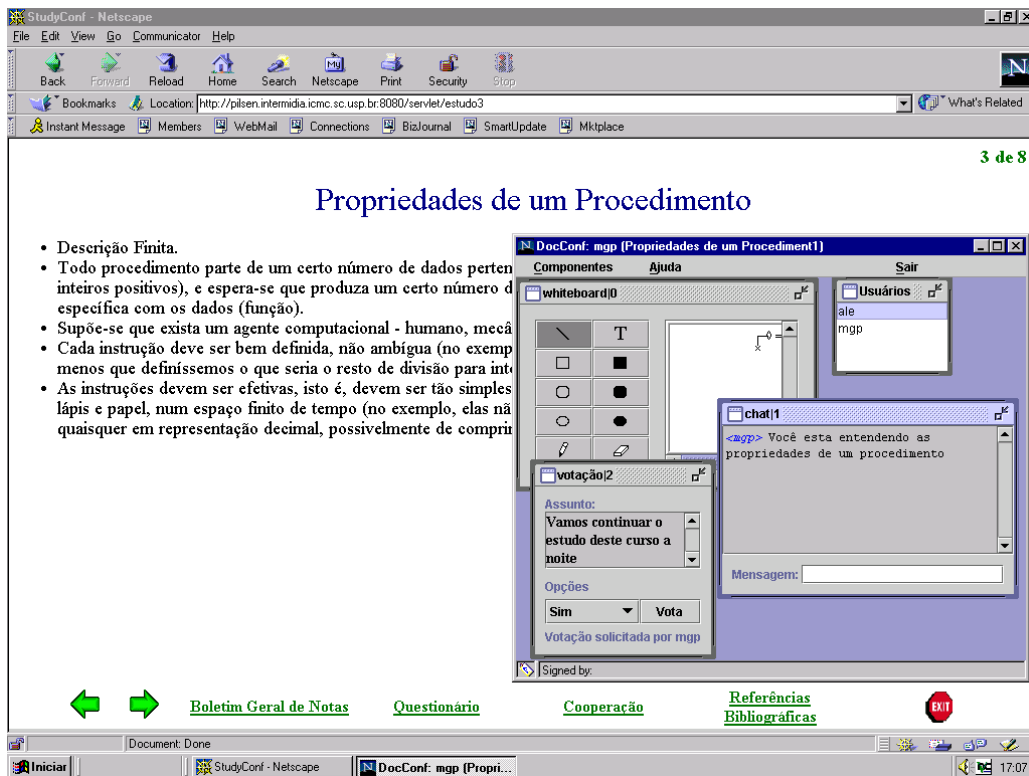


Figura 37: Exemplo de utilização do DocConf no StudyConf [Macedo et al 99b]

6.5 CONSIDERAÇÕES FINAIS

Um dos principais requisitos do DocConf é ser extensível, isto é, permitir que tipos de interação entre os participantes de uma sessão, que não tenham sido previstos durante seu desenvolvimento, possam ser criados a qualquer momento. Assim, durante a implementação do DocConf, procurou-se isolar dos componentes todo o código responsável por questões como comunicação e segurança.

A metodologia CSCW-SH, adotada pelo DocConf, procura facilitar e orientar o desenvolvimento de novos componentes. Estes devem tratar apenas a interface com o usuário e a geração/processamento de informações. Para isso, deverão implementar algumas classes e métodos especificados neste capítulo.

A incorporação do DocConf pelo StudyConf mostra que o ambiente também pode ser facilmente adaptado para integração a projetos que utilizem outra forma de comunicação ou de gerenciamento de sessões.

7

CONCLUSÕES

7.1 MOTIVAÇÃO

Muito se tem discutido acerca da utilização de hiperdocumentos como forma de estruturação e armazenamento de informações em aplicações de CSCW [Nielsen 90; Streitz et al 91; Fuchs 96; Borges 95]. Características inerentes aos hiperdocumentos, como a facilidade de criação e combinação de módulos de informação, permitem a sua utilização tanto na troca de mensagens entre os participantes quanto no armazenamento da memória da sessão.

No contexto de CSCW, inúmeros pesquisadores sugerem que a WWW pode ser utilizada como plataforma para aplicações colaborativas [Bentley et al 95; Busbach et al 96; Dix 96; Walther 96]. Apesar de garantir independência entre plataformas e uma interface consistente, a arquitetura atual da WWW não é suficiente para suportar a colaboração entre agentes humanos e/ou computacionais independentes.

O ambiente apresentado nesta dissertação, o DocConf, procura fornecer uma infra-estrutura com os recursos necessários para o suporte à colaboração no contexto da WWW, além de investigar o uso de hiperdocumentos XML no registro das comunicações.

O DocConf foi desenvolvido de acordo com a metodologia CSCW-SH, cuja criação foi motivada pela necessidade de apoio à construção de aplicações colaborativas que utilizam hiperdocumentos para formalizar a estrutura das sessões.

7.2 CONTRIBUIÇÕES

A pesquisa bibliográfica reportada nesta dissertação contribuiu diretamente para a produção de uma nota didática do ICMC intitulada "*AMBIENTES COOPERATIVOS: TENDÊNCIAS E EXEMPLOS*" [Pimentel et al 99a]. Tal recurso é útil para os muitos alunos deste instituto, que

podem utilizá-lo como significativa fonte de referência nessa área em constante desenvolvimento.

A pesquisa bibliográfica também levou à implementação de vários protótipos de sistemas de comunicação, os quais, por sua vez, motivaram a criação da metodologia CSCW-SH a subsequente implementação do DocChat, construído para validá-la. A metodologia foi tema do artigo "*HIPERDOCUMENTOS ESTRUTURADOS NO SUPORTE AO TRABALHO COOPERATIVO EM SISTEMAS ABERTOS DISTRIBUÍDOS*", apresentado no SEMISH'98 [Pimentel et al 98].

A evolução do trabalho levou à implementação do ambiente DocConf, desenvolvido a partir do DocChat. Com essa implementação, puderam-se testar e analisar as diferentes formas de comunicação disponibilizadas pela linguagem Java. Essa análise resultou na opção pela utilização de *sockets*, que são independentes de quaisquer outros módulos e garantem uma taxa de transmissão aceitável.

Outra contribuição deste trabalho foi a utilização do DocConf no ambiente StudyConf, um ambiente de apoio ao aprendizado que dá suporte sessões de trabalho colaborativo. O DocConf supriu a necessidade do StudyConf na realização de sessões de discussão entre seus usuários.

Os aspectos inovadores relacionados ao registro das sessões de trabalho com o uso de documentos XML são tema do artigo "*REGISTERING WEB-BASED CONFERENCING WITH STRUCTURED XML DOCUMENTS*", a ser apresentado na WebNet 99 [Pimentel et al 99b].

7.3 TRABALHOS FUTUROS

Uma tarefa importante correspondente à execução de experimentos para avaliação da interação do usuário com o ambiente DocConf. Para isso, informações valiosas podem ser obtidas a partir da utilização do DocConf no ambiente StudyConf.

No contexto da metodologia CSCW-SH, um trabalho que pode ser desenvolvido é a extensão da metodologia para que essa inclua técnicas de avaliação da qualidade dos produtos intermediários obtidos em cada passo. Outro trabalho a ser desenvolvido é a pesquisa de outras metodologias

que apóiem o desenvolvimento de aplicações colaborativas para comparação com a CSCW-SH, objetivando o aperfeiçoamento dessa.

Também se pode estudar o uso conjunto da metodologia CSCW-SH com a MCLAi – Modelagem Conceitual e Lógica de Aplicações para a Internet. Essa modelagem visa à especificação da estrutura dos documentos com o objetivo de auxiliar o processamento dos mesmos para armazenamento das informações em um banco de dados.

Como continuidade natural da implementação do ambiente DocConf, pode ser realizado o desenvolvimento de componentes que permitam novos tipos de interação entre os usuários, como por exemplo:

- Gerador de gráficos (barra, linhas, etc.);
- Acesso a uma base de dados;
- Visualizador de objetos 3D e VRML;
- Navegador HTML cooperativo e
- Editor de textos cooperativo.

Outra proposta de trabalho futuro é a implementação de visões de janelas de outros usuários. Essas visões poderão permitir que um usuário veja exatamente o que outro está vendo. Todos os eventos ocorridos na janela original serão, instantaneamente, reproduzidos na outra janela. Através desse recurso, um usuário poderia, por exemplo, acompanhar as alterações que outro estaria fazendo em um documento, ao invés de ver somente o resultado dessas alterações. O treinamento à distância de usuários para o uso do ambiente também se beneficiaria com esse recurso, pois eles poderiam acompanhar as ações do treinador, em suas próprias telas.

Apesar do DocConf possuir todos os módulos de comunicação necessários, ele utiliza uma arquitetura de comunicação centralizada, onde o servidor é responsável pela transmissão de todas as mensagens. Uma proposta de trabalho futuro é a implementação de uma arquitetura replicada. A implementação de ambas arquiteturas é sugerida por [Greenberg & Roseman 98].

7.4 CONSIDERAÇÕES FINAIS

O trabalho reportado nesta dissertação alcançou seus objetivos e resultou nas contribuições descritas. Entre os pontos positivos apresentados estão a revisão bibliográfica das áreas de Hipermídia e CSCW gerada, a especificação da metodologia CSCW-SH e a implementação do ambiente DocConf. O trabalho realizado abriu caminho para a realização de novas tarefas associadas à investigação do uso integrado das tecnologias de Hipermídia e CSCW.

REFERÊNCIAS BIBLIOGRÁFICAS

- [Adler 97] Adler, Sharo; Berglund, Anders; Clark, James; Cseri, Istvan; Grosso, Paul; Marsh, Jonathan; Nicol, Gavin; Paoli, Jean; Schach, David; Thompson, Henry S.; Wilson, Chris. *A PROPOSAL FOR EXTENSIBLE STYLE LANGUAGE (XSL)*. W3C Note. Setembro 1997.
<http://www.w3.org/TR/NOTE-XSL.html>
- [Amaral et al 97] Amaral, Vinícius; Grala, Anderson; Heuser, Carlos A.; Lima, José V. *UMA ARQUITETURA ABERTA PARA A INTEGRAÇÃO DE SISTEMAS DE GERÊNCIA DE DOCUMENTOS E SISTEMAS DE GERÊNCIA DE WORKFLOW*. III Workshop sobre Sistemas Hipermedia. pp. 131-142. Maio 1997.
- [Bacelo & Becker 97] Bacelo, Ana Paulo; Becker, Karin. *UMA FERRAMENTA DE APOIO À DISCUSSÃO E DELIBERAÇÃO EM GRUPO*. III Workshop sobre Sistemas Hipermedia. pp. 119-130. Maio 1997.
- [Bair 89] Bair, James. *SUPPORTING COOPERATIVE WORK WITH COMPUTERS: ADDRESSING MEETING MANIA*. IEEE Computer. Nº 4. pp 208-217. Abril 1989.
- [Bentley et al 95] Bentley, Richard; Horstmann, Thilo; Sikkel, Klaas; Trevor, Jonathan. *SUPPORTING COLLABORATIVE INFORMATION SHARING WITH THE WORLD-WIDE-WEB: THE BSCW SHARED WORKSPACE*. 4th International World Wide Web Conference Proceedings. Dezembro 1995.
- [Borges & Araújo 94] Borges, Marcos; Araújo, Renata. *QUORUM - UM SSDG PARA O DESENVOLVIMENTO DE SOFTWARE*. VIII Brazilian Symposium on Software Engineering, 1994.

- [Borges et al 95] Borges, Marcos; Campos, Ma. Luiza; Cavalcanti, Ma. Cláudia Reis. *SUPORTE POR COMPUTADOR AO TRABALHO COOPERATIVO*. XV Congresso da Sociedade Brasileira de Computação. 1995.
- [Bosak 97] Bosak, John. *XML, JAVA, AND THE FUTURE OF THE WEB*. World Wide Web Journal: XML Principles, Tools and Techniques. Vol 2. Nº 4. pp 219-227. Setembro 1997.
- [Bray et al 98] Bray, Tim; Paoli, Jean; Sperberg-McQueen, C. M. *EXTENSIBLE MARKUP LANGUAGE (XML) 1.0*. W3C Recommendation. Fevereiro 1998.
<http://www.w3.org/TR/1998/REC-xml-199980210.html>
- [Busbach et al 96] Busbach, Uwe; Kerr, David; Sikkel, Klaas; *FOREWORD*. ERCIM Workshop on CSCW and the Web Proceedings. Fevereiro 1996.
- [Chabert et al 98] Chabert, Annie; Grossman, Eddie; Jackson, Larry; Pietrowicz, Stephen; Seguin, Chris. *JAVA OBJECT SHARING IN HABANERO*. Communications of the ACM. Vol. 41. Nº 6. pp 69-76. Junho 1998.
- [Crow et al 97] Crow, D.; Parsowith, S.; Bowden Wise, G. [com Paul Dourish, Saul Greenberg, Jonathan Grudin e Yvonne Rogers]. *STUDENTS: THE EVOLUTION OF CSCW – PAST, PRESENT AND FUTURE DEVELOPMENTS*. ACM SIGCHI Bulletin, Vol. 29. Nº 2. Abril 1997.
- [Culshaw et al 97] Culshaw, Stuart; Leventhal, Michael; Maloney, Murray. *XML AND CSS*. World Wide Web Journal: XML Principles, Tools and Techniques. Vol 2. Nº 4. pp 109-118. Setembro 1997.
- [Daily et al 96] Daily, Bonnie; Whatley, Art; Ash, Steven; Steiner, Robert. *THE EFFECTS OF A GROUP DECISION SUPPORT SYSTEM ON CULTURALLY DIVERSE AND CULTURALLY HOMOGENEOUS GROUP DECISION MAKING*. Information & Management. Nº 30. pp 281-289. 1996.

- [DeRose & Durand 94] DeRose, Steven; Durand, David. *MAKING HYPERMEDIA WORK – A USER’S GUIDE TO HYTIME*. Kluwer Academic Publishers. 1994.
- [Dix 96] Dix, Alan. *CHALLENGES AND PERSPECTIVES FOR COOPERATIVE WORK ON THE WEB*. ERCIM Workshop on CSCW and the Web Proceedings, Fevereiro 1996.
- [Dourish 98] Dourish, Paul. *USING METALEVEL TECHNIQUES IN A FLEXIBLE TOOLKIT FOR CSCW APPLICATIONS*. ACM Transactions on Computer-Human Interaction, Vol. 5. No.2. pp 109-155. Junho 1998.
- [Ellis et al 91] Ellis, Clarence; Gibbs, Simon; Rein, Gail. *GROUPWARE: SOME ISSUES AND EXPERIENCES*. Communications of the ACM. Vol. 34. N° 12. pp 38-58. Janeiro 1991.
- [Ensor et al 90] Ensor, Bob; Crowley, Terry; Kraut, Bob; Rein, Gail; Sproull, Lee. *HOW CAN WE MAKE GROUPWARE PRATICAL?* CHI'90 Conference Proceedings. pp 87-89. Abril 1990.
- [Fuchs 96] Fuchs, Matthew. *LET'S TALK: EXTENDING THE WEB TO SUPPORT COLLABORATION*. Proceedings of the 5TH Workshops on Enabling Technologies. Junho 1996.
- [Fujitsu & TechnoTeacher 95] Fujitsu Open Systems Solutions; TechnoTeacher, Inc.. *HYTIME APPLICATION DEVELOPMENT GUIDE*. Maio 1995.
<http://www.phxdata.com/hytime.html>
- [Goldfarb 90] Goldfarb, Charles F. *THE SGML HANDBOOK*. Oxford University Press, Inc. 1990.
- [Greenberg & Roseman 96] Greenberg, Saul; Roseman, Mark. *BUILDING REAL-TIME GROUPWARE WITH GROUPKIT, A GROUPWARE TOOLKIT*. ACM Transactions on Computer-Human Interaction. Vol. 3. N° 1. Março 1996.

- [Greenberg & Roseman 97] Greenberg, Saul; Roseman, Mark. *SIMPLIFYING COMPONENT DEVELOPMENT IN NA INTEGRATED GROUPWARE ENVIRONMENT*. 10th ACM Symposium on User Interface Software and Technology - UIST'97. pp. 64-72. Outubro 1997.
- [Greenberg & Roseman 98] Greenberg, Saul; Roseman, Mark. *GROUPWARE TOOLKITS FOR SYNCHRONOUS WORK* in M. Beaudouin-Lafon, Ed., Computer-Supported Cooperative Work, Trends in Software Series, John Wiley & Sons. 1998.
- [Grudin 91] Grudin, Jonathan. *CSCW*. Communications of the ACM. Vol. 34. N° 12. pp 30-34. Dezembro 1991.
- [Grudin 94] Grudin, Jonathan. *CSCW: HISTORY AND FOCUS*. IEEE Computer. N° 5. pp 19-26. Maio 1994.
- [Halasz 88] Halasz, Frank G. *REFLECTIONS ON NOTECARDS: SEVEN ISSUES FOR THE NEXT GENERATION OF HYPERMEDIA SYSTEMS*. Communications of the ACM. Vol. 31. N° 7. Pp 836-852. Julho 1988.
- [Herwijnen 94] Herwijnen, Eric Van. *PRATICAL SGML*. Kluwer Academic Publishers. 1994.
- [Hills 97] Hills, Mellanie. *INTRANET AS GROUPWARE*. John Wiley & Sons. 1997.
- [Hoschka et al 98] Hoschka, Phillipp; Bugaj, Stephan; Bulterman, Dick; Hardman, Lynda; Jansen, Jack; Lanphier, Rob; Layaida, Nabil; Mash, Jonathan; Rao, Anup; Rutledge, Lloyd, Kate, Warner ten; Ossenbruggen, Jacco van; Vernick, Michael; Yu, Jin. *SYNCHRONIZED MULTIMEDIA INTEGRATION LANGUAGE (SMIL)*. W3C Working Draft. Fevereiro 1998.
<http://www.w3.org/TR/WD-smil>

- [Ishii & Miyake 91] Ishii, Hiroshi; Miyake, Naomi. *TOWARD AN OPEN SHARED WORKSPACE: COMPUTER AND VIDEO FUSION APPROACH TO TEAMWORKSTATION*. Communications of the ACM. Vol. 24. N° 12. pp. 36-50. Dezembro 1991.
- [ISO 86] ISO/IEC *Standard Generalized Markup Language — SGML* 1986:8879.
- [ISO 92] ISO/IEC *Hypermedia/Time-Based Structuring Language — HyTime*. 1992:10744.
- [Johansen 88] Johansen, Robert. *GROUPWARE: COMPUTER SUPPORT FOR BUSINESS TEAMS*. Series in Communication Technology and Society, The Free Press, 1988.
- [Johnson 99] Johnson, Mark. *XML FOR THE ABSOLUTE BEGINNER*. Java World. Abril 1999.
<http://www.javaworld.com/javaworld/jw-04-1999/jw-04-xml-p.html>
- [Johnson-Lenz 80] Johnson-Lenz, Peter; Johnson-Lenz, Trudy. *GROUPWARE: THE EMERGING ART OF ORCHESTRATING COLLECTIVE INTELLIGENCE*. World Future Society's First Global Conference on the Future. 1980.
- [Lubich 95] Lubich, Hannes P. *TOWARDS A CSCW FRAMEWORK FOR SCIENTIFIC COOPERATION IN EUROPE*. Lecture Notes on Computer Science 889. 1995.
- [Mace et al 98] Mace, Scott; Flohr, Udo; Dobson, Rick; Graham, Tony. *TECENDO UM REDE MELHOR*. Byte Brasil. N° 78. pp 76-84. Março 1998.
- [Macedo et al 99a] Macedo, Alessandra; Pires, Daniel; Pimentel, Maria da Graça; Fortes, Renata. *MCLAi: MODELAGEM CONCEITUAL E LÓGICA DE APLICAÇÕES PARA A INTERNET*. Artigo submetido ao SBES'99.

- [Macedo et al 99b] Macedo, Alessandra; Pimentel, Maria da Graça; Fortes, Renata. *STUDYCONF: INFRA-ESTRUTURA DE SUPORTE AO APRENDIZADO COOPERATIVO NA WWW*. Revista Brasileira de Informática na Educação (RBIE). Setembro 1999.
- [Miner & Ion 97] Miner, Robert; Ion, Patrick. *HTML-MATH - MATHEMATICAL MARKUP LANGUAGE WORKING DRAFT*. World Wide Web Journal: XML Principles, Tools and Techniques. Vol 2. Nº 4. pp 83-89. Setembro 1997.
- [Morgenthal 99] Morgenthal, JP. *PORTABLE DATA / PORTABLE CODE: XML & JAVA TECHNOLOGIES*. Java World. Março 1999.
<http://java.sun.com/xml/ncfocus.html>
- [Nielsen 90] Nielsen, Jacob. *HYPERTEXT AND HYPERMEDIA*. Academic Press. 1990.
- [Oliveira & Soares 96] Oliveira, J. C.; Soares, L. F. G. *TVS - UM SISTEMA DE VIDEOCONFERÊNCIA COM DOCUMENTOS COMPARTILHADOS - UMA VISÃO GERAL*. II Workshop sobre Sistemas Hiperfídia. pp. 79-88. Maio 1996.
- [Paoli et al 97] Paoli, Jean; Schach, David; Lovett, Chris; Layman, Andrew; Cseri, Istvan. *BUILDING XML PARSERS FOR MICROSOFT'S IE4*. World Wide Web Journal: XML Principles, Tools and Techniques. Vol 2. Nº 4. pp 187-195. Setembro 1997.
- [Pimentel et al 98] Pimentel, Maria da Graça; Teixeira, César; Kutova, Marcos; Macedo, Alessandra; Fortes, Renata. *HIPERDOCUMENTOS ESTRUTURADOS NO SUPORTE AO TRABALHO COOPERATIVO EM SISTEMAS ABERTOS DISTRIBUÍDOS*. XXV Seminário Integrado de Software e Hardware - SEMISH. pp 158-173. Agosto 1998.

- [Pimentel et al 99a] Pimentel, Maria da Graça; Kutova, Marcos; Macedo, Alessandra. *AMBIENTES COOPERATIVOS: TENDÊNCIAS E EXEMPLOS*. Nota Didática do ICMC nº 35. Fevereiro 1999.
- [Pimentel et al 99b] Pimentel, Maria da Graça; Kutova, Marcos; Teixeira, César; Fortes, Renata. *REGISTERING WEB-BASED CONFERENCING WITH XML DOCUMENTS*. Artigo aceito para a WebNet 99.
- [Poltrock & Grudin 96] Poltrock, Steven; Grudin, Jonathan, *GROUPWARE AND WORKFLOW: A SURVEY OF SYSTEMS AND BEHAVIORAL ISSUES*. Tutorial in CHI'96 Proceedings. Vancouver, Canada. Abril 1996.
<http://www.acm.org/sigchi/chi96/>
- [Raggett et al 97] Raggett, Dave; Hors, Arnaud Le; Jacobs, Ian. *HTML 4.0 SPECIFICATION*. W3C Recommendation. Dezembro 1997 .
<http://www.w3.org/TR/REC-html40-971218.html>
- [Santos et al 96] Santos, Antônio C.; Meira, Fábio; Nakamura, Adriano; Scholten, Andrea; Alves, Alexandre; *SACE-CSCW: MÉTAFORAS DE SUAS INTERFACES*. II Workshop sobre Sistemas Hipermídia. pp. 109-118. Fortaleza. Maio 96.
- [Streitz et al 91] Streitz, Norbert. *THE ROLE OF HYPERTEXT FOR CSCW APPLICATIONS*. ACM Hypertext'91 Panel. Panel moderator. pp. 369-375. 1991.
- [Trindade & Pimentel 97] Trindade, Ciro; Pimentel, Maria da Graça. *ESPECIFICAÇÃO HYTIME DE ESTRUTURAS HIPERTEXTO*. III Workshop sobre Sistemas Hipermídia. pp. 213-225. Maio 97.
- [W3C 97] World Wide Web Consortium. *W3C ARQUITETCTURE DOMAIN - EXTENDED MARKUP LANGUAGE (XML)*.
<http://www.w3.org/Math/Activity.html>

- [W3C 98] World Wide Web Consortium. *W3C ACTIVITY: MATH*.
<http://www.w3.org/Math/Activity.html>
- [Walsh 97] Walsh, Norman. *A GUIDE TO XML*. World Wide Web Journal: XML Principles, Tools and Techniques. Vol 2. N° 4. pp 29-56. Setembro 1997.
- [Walther 96] Walther, M. *SUPPORTING DEVELOPMENT OF SYNCHRONOUS COLLABORATION TOOLS ON THE WEB WITH GROCO*. Proceedings of the ERCIM Workshop on CSCW and the Web. Fevereiro 1996.
<http://orgwis.gmd.de/projects/proceedings/groco.html>
- [Watson 92] Watson, Dennis. *BRIEF HISTORY OF DOCUMENT MARKUP*. Circular 1086. Florida Cooperative Extension Service. University of Florida. Novembro 1992.