*EMEET/SERVE: GERAÇÃO SOB-DEMANDA DE
HIPERDOCUMENTOS MULTIMÍDIA A PARTIR DE INFORMAÇÃO
CAPTURADA EM REUNIÕES DISTRIBUÍDAS*

Pedro Monteiro da Silva Eleutério

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

*Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação de São Carlos
– USP, como parte dos requisitos para a obtenção do título de Mestre em Ciências de
Computação e Matemática Computacional.*

USP – São Carlos
Abril de 2002

Dedicatória

Dedico este trabalho aos meus pais, Danilo e Leonor, pela chance que me deram em poder chegar até aqui, sempre me ensinando e me apoiando em todos os momentos.

Agradecimentos

Inicialmente a Deus, por toda a força e proteção.

A minha orientadora Graça Pimentel, pelo ótimo trabalho realizado.

Ao companheiro Wagner, pelo trabalho conjunto realizado.

Aos amigos e colegas do laboratório Intermídia: Otávio, Orlando, Juninho, Andrea, Alessandra, Toño, Renan e Débora. Especialmente ao Renato e Cláudia, pela ajuda prestada no término deste trabalho.

Ao ICMC/USP, pela oportunidade de realizar este trabalho.

A Radiumsystems.com, pelo suporte oferecido.

Aos meus irmãos Fábio, Danilo e Valéria e parentes mais próximos, por me apoiarem sempre.

A minha namorada Fernanda, pela compreensão demonstrada.

Aos atuais colegas e ex-companheiros da Radiumsystems.com, pelo ótimo ambiente de trabalho proporcionado.

Ao amigo e ex-colega de trabalho Ramon, pelo apoio durante a fase de qualificação deste trabalho.

Ao amigo e colega de república DG, pelas cervejas e tequilas compartilhadas durante todo esse tempo.

A toda a minha turma de Engenharia de Computação '95, pela amizade demonstrada.

Aos meus amigos de Araraquara e de São Carlos, pelo companheirismo e amizade demonstrados.

Ao Pink Floyd, Dire Straits e Alan Parsons pela ótima música proporcionada durante a realização deste e de muitos outros trabalhos.

Ao meu teclado Yamaha, pelas horas de descontração proporcionadas.

As minhas raquetes de tênis e bolas de basquete, pelos jogos e torneios participados.

E a todos que contribuíram direta ou indiretamente para a realização deste trabalho, ou que simplesmente conviveram comigo nestes últimos anos...

Índice

Resumo.....	xii
Abstract.....	xiii
Capítulo 1. Introdução.....	1
1.1. Contexto	1
1.2. Motivação.....	3
1.3. Objetivo.....	3
1.4. Resultados Alcançados.....	3
1.5. Estrutura	4
Capítulo 2. InCA-SERVE.....	5
2.1. Considerações iniciais	5
2.2. <i>eClass</i>	5
2.3. A infra-estrutura InCA	9
2.4. A infra-estrutura SERVE	9
2.5. <i>eMeet</i>	10
2.6. Considerações finais.....	13
Capítulo 3. Computação Ubíqua do ponto de vista de trabalho cooperativo.....	14
3.1. Considerações iniciais	14
3.2. Hipertexto e trabalho cooperativo	14
3.2.1. Origem.....	14
3.2.2. Um <i>framework</i> para classificação (Ellis et al., 1991)	15
3.3. Computação Ubíqua.....	17
3.3.1. Origem.....	17
3.3.2. Experiências com capturas em dispositivos de superfícies interativas	18
3.3.3. Experiências com captura de áudio.....	20
3.3.4. Experiências com captura de vídeo.....	21
3.4. Considerações finais.....	21
Capítulo 4. Captura e apresentação de informação multimídia.....	22
4.1. Considerações iniciais	22
4.2. Sistemas para captura e apresentação	22
4.2.1. Cornell Lecture Browser	22
4.2.2. Notelook.....	27

4.2.3. TELEP	28
4.2.4. MSTE	32
4.2.5. TeamSpace	34
4.3. Técnicas de apresentação e sumarização	36
4.3.1. Apresentação de vídeo	36
4.3.2. Sumarização de apresentações	40
4.4. Fóruns de discussão.....	42
4.5. Considerações finais.....	45
Capítulo 5. <i>eMeet/SERVE</i> : Especificação e Arquitetura	47
5.1. Considerações iniciais	47
5.2. <i>eMeet/SERVE</i> : Especificação, Modelagem e Características	47
5.2.1. Especificação Inicial do <i>eMeet/SERVE</i>	48
5.2.2. Diagramas UML de Estudo de Caso.....	48
5.2.2.1. Gerenciamento de Salas Virtuais	48
5.2.2.2. Gerenciamento de Usuários	49
5.2.2.3. Gerenciamento de Reuniões.....	49
5.2.2.4. Visualizar Reunião	50
5.2.2.5. Envio de Mensagem – Fale Conosco	50
5.2.3. Seções e Características do <i>eMeet/SERVE</i>	51
5.3. Arquitetura LAIAc proposta para sistemas de geração de hiperdocumentos	51
5.3.1. Módulo de dados	53
5.3.2. Módulo de acesso	53
5.3.3. Módulo de idiomas.....	54
5.3.4. Módulo de <i>layout</i>	55
5.3.5. Aplicação.....	56
5.4. O <i>eMeet/SERVE</i> no contexto da arquitetura LAIAc	57
5.4.1. Módulo de dados do <i>eMeet/SERVE</i>	58
5.4.1.1. Modelo Entidade Relacionamento (MER).....	58
5.4.1.2. Dicionário de Dados.....	62
5.4.2. Módulo de acesso do <i>eMeet/SERVE</i>	63
5.4.2.1. Arquivos mestres do módulo de acesso do <i>eMeet/SERVE</i>	65
5.4.2.2. Arquivos específicos do módulo de acesso do <i>eMeet/SERVE</i>	65
5.4.3. Módulo de idiomas do <i>eMeet/SERVE</i>	70

5.4.4. Módulo de <i>layout</i> no <i>eMeet/SERVE</i>	72
5.4.5. A aplicação <i>eMeet/SERVE</i>	74
5.4.5.1. Arquivos Geradores	76
5.4.5.2. Arquivos de Tratamento.....	79
5.5. Considerações finais.....	81
Capítulo 6. <i>eMeet/SERVE</i> : Hiperdocumentos Gerados.....	82
6.1. Considerações iniciais.....	82
6.2. Estrutura e Conteúdo dos Hiperdocumentos.....	83
6.3. Interação.....	88
6.3.1. Botões de ação.....	88
6.3.2. Indicador de ações.....	89
6.3.3. Instruções e Informações de Ajuda	90
6.3.4. Tratamento de Erros	90
6.4. Tratamento de sincronização.....	92
6.4.1. Área de <i>Slides</i>	94
6.4.2. Área de Linha do Tempo.....	96
6.4.3. Área de Vídeo	97
6.4.4. Controlador Central do Processo de Sincronização	99
6.5. Controle de acesso.....	101
6.6. Considerações finais.....	103
Capítulo 7. Conclusões.....	104
7.1. Considerações iniciais.....	104
7.2. Discussão dos resultados.....	104
7.3. Contribuições	105
7.4. Trabalhos Futuros.....	106
7.5. Considerações finais.....	108
Referências Bibliográficas	109
Apêndice A. Manual do Usuário do <i>eMeet/SERVE</i>	114
A.1. Introdução e requisitos	114
A.2. Página Inicial.....	114
A.3. Seção Reuniões	115
A.3.1. Cadastro de novas reuniões	116
A.3.2. Visualização de reuniões.....	117

A.3.2.1. Área de slides	118
A.3.2.2. Linha do tempo.....	119
A.3.2.3. Vídeo	119
A.3.3. Edição e remoção de reuniões já existentes	120
A.4. Seção Salas	120
A.4.1. Cadastro de novas salas	121
A.4.2. Listagem de salas cadastradas	121
A.4.3. Edição e remoção de salas já existentes	122
A.5. Seção Usuários	122
A.5.1. Cadastro de novos usuários	123
A.5.2. Edição e remoção de usuários já existentes	124
A.6. Seção Fale Conosco	124
Apêndice B. Código de geração das tabelas do <i>eMeet/SERVE</i> no MySQL.....	126

Índice de Figuras

Figura 1: Dispositivos do <i>eClass</i> - (A) lousa eletrônica, (B) câmeras de vídeo, (C) microfones, (D) projetor com navegação Web e (E) projetor com informações anteriormente apresentadas na (A).	6
Figura 2: Hiperdocumento multimídia gerado automaticamente pelo <i>eClass</i>	7
Figura 3: <i>Workflow</i> do <i>eClass</i> em relação ao tempo. Com a evolução do <i>eClass</i> , a fase de integração é realizada automaticamente por software (Pimentel & Abowd, 1999). Dessa forma, as informações para fase de acesso são disponibilizadas rapidamente.....	8
Figura 4: A infra-estrutura InCA-SERVE implementada para suporte a reuniões distribuídas. ..	10
Figura 5: Interface do <i>eMeet/InCA</i> dos participantes de uma reunião distribuída. Os <i>slides</i> e os áudios/vídeos dos participantes são visualizados em tempo real, durante a reunião.	11
Figura 6: Exemplo de hiperdocumento multimídia gerado pelo <i>eMeet/SERVE</i> . A área central apresenta a seqüência de <i>slides</i> capturados durante a reunião. Na parte inferior, encontra-se a linha do tempo. O vídeo e seus controles estão abaixo do menu lateral esquerdo.....	12
Figura 7: <i>Framework</i> clássico para classificar sistemas cooperativos.....	16
Figura 8: Dispositivos ubíquos com superfície interativa: (a) caderneta eletrônica e (b) lousa eletrônica.	19
Figura 9: Hiperdocumento de apresentação do Lecture Browser (Mukhopadhyay & Smith, 1999)	24
Figura 10: Tipos de sincronização multimídia (Mukhopadhyay & Smith, 1999).	24
Figura 11: Esquema utilizado no Cornell Lecture Browser para realizar a sincronização de dois vídeos (Mukhopadhyay & Smith, 1999).	25
Figura 12: Interface que é exibida para os expectadores remotos no sistema TELEP.	31
Figura 13: Interface Web do MSTE (He et al., 2000).....	33
Figura 14: Apresentação dos dados de uma reunião no TeamSpace. Na janela maior, é mostrado o <i>applet</i> de visualização contendo as linhas do tempo (no alto) e os painéis segmentados com o conteúdo do hiperdocumento gerado (ao centro). Tais painéis também podem ser apresentados em janelas independentes, mostradas à esquerda.	36
Figura 15: Interface da ferramenta criada por Li et al. (2000) com controles avançados.....	39
Figura 16: Representação de ator, ação e relação em UML.....	48

Figura 17: Diagrama UML de estudo de caso para o gerenciamento de salas virtuais no <i>eMeet/SERVE</i> .	49
Figura 18: Diagrama UML de estudo de caso para o gerenciamento de usuários no <i>eMeet/SERVE</i> .	49
Figura 19: Diagrama UML de estudo de caso para o gerenciamento de reuniões no <i>eMeet/SERVE</i> . A ação "Visualizar Reunião" é composta de outras ações, ilustradas na próxima sub-seção.	50
Figura 20: Diagrama UML de estudo de caso para a visualização de reuniões no <i>eMeet/SERVE</i> .	50
Figura 21: Diagrama UML de estudo de caso para o envio de mensagens aos autores/mantenedores no <i>eMeet/SERVE</i> .	51
Figura 22: Esboço da arquitetura LAIAc proposta para sistemas de geração de hiperdocumentos. Os retângulos tradicionais representam os módulos da arquitetura. Os retângulos com canto dobrado indicam os hiperdocumentos. O retângulo com borda arredondada indica integração com outras aplicações. As setas indicam o fluxo de informação durante o processo de geração dos hiperdocumentos.	52
Figura 23: Arquitetura LAIAc utilizada no desenvolvimento do <i>eMeet/SERVE</i> .	57
Figura 24: Modelo Entidade Relacionamento (MER) da base de dados do <i>eMeet/SERVE</i> . O MER é composto de cinco tabelas, responsáveis em armazenar dados e metadados sobre as reuniões capturadas.	58
Figura 25: Tabela <i>room</i> e seus campos.	59
Figura 26: Tabela <i>meeting_room</i> e seus campos.	59
Figura 27: Tabela <i>meeting</i> e seus campos.	60
Figura 28: Tabela <i>slide</i> e seus campos.	61
Figura 29: Tabela <i>penstroke</i> e seus campos.	62
Figura 30: Detalhamento do módulo de acesso e integração com a aplicação e com o módulo de dados	64
Figura 31: Detalhamento do módulo de idiomas e integração com a aplicação	71
Figura 32: Exemplo de utilização de definições de textos com PHP	71
Figura 33: Funcionamento básico de requisição ao servidor Web com a linguagem PHP	76
Figura 34: Exemplo de um arquivo gerador e seus hiperdocumentos gerados. O arquivo gerador recebe o código da reunião e gera hiperdocumentos de mesma família.	77

Figura 35: Estrutura básica de um arquivo gerador do <i>eMeet/SERVE</i>	77
Figura 36: Hiperdocumento gerado pelo <i>eMeet/SERVE</i> com parte comum destacada. A parte comum representa a barra superior, o logo do <i>eMeet</i> , o menu lateral esquerdo e a caixa/indicação de <i>login</i>	79
Figura 37: Exemplo de utilização de um arquivo de tratamento. (A) envio de dados do formulário, (B) ativação do módulo de acesso e (C) redirecionamento para o hiperdocumento adequado através de um arquivo gerador.	80
Figura 38: Estrutura básica de um arquivo de tratamento do <i>eMeet/SERVE</i>	81
Figura 39: Hiperdocumento gerado: (a) logo do <i>eMeet</i> , (b) menu lateral esquerdo, (c) seção correspondente, (d) botão voltar, (e) instruções aos usuários, (f) opções de ajuda, (g) botões de ação e (h) padrão de cores, fontes e estilos.	83
Figura 40: Hiperdocumento gerado: (f) opções de ajuda, (i) saudação, (j) campos obrigatórios, (k) funcionalidade principal do hiperdocumento, (l) campos de formulários e (m) ordem dos botões de ação.	85
Figura 41: Hiperdocumento gerado: (d) botão voltar, (k) funcionalidade principal do hiperdocumento, (n) controle de vídeo, (o) controle de <i>slides</i> , (p) tempo de apresentação, (q) <i>slide</i> atual, (r) linha do tempo, (s) troca de <i>slide</i> e (t) não utilização de <i>frames</i> e barras de rolagem horizontais.	86
Figura 42: Exemplo de botões visuais, seguidos de indicações textuais, utilizados nos hiperdocumentos gerados pelo <i>eMeet/SERVE</i> para representar ações.	88
Figura 43: Exemplo de indicador de ações nos hiperdocumentos gerados.	89
Figura 44: Ícone de ajuda e um exemplo de ajuda adicional.	90
Figura 45: Exemplo de hiperdocumento com mensagem de erro aos usuários.	91
Figura 46: Área de <i>slides</i> nos hiperdocumentos multimídia. A indicação do <i>slide</i> atual é apresentada na parte inferior central. É possível avançar ou retroceder os <i>slides</i> , através dos botões “Anterior” e “Próximo”, respectivamente.	95
Figura 47: Exemplo de utilização da tag <DIV> do HTML.	96
Figura 48: Área de linha do tempo nos hiperdocumentos multimídia. Existe uma barra vermelha que indica o tempo atual. As demais são verdes e indicam as trocas de <i>slides</i> . Os números azuis indicam tempos da reunião.	96
Figura 49: Área de vídeo durante a apresentação da reunião e os controles <i>back</i> , <i>play</i> e <i>pause</i> , respectivamente.	98
Figura 50: Exemplo de acesso a função <code>getTime()</code> do <i>applet</i> Java através do JavaScript.	98

Figura 51: Função <code>checksync()</code> é chamada assim que o hiperdocumento é carregado no navegador do cliente.	99
Figura 52: Passos da função responsável pela sincronização.	100
Figura 53: Formulário para escolha do nível de controle de acesso de uma reunião.....	101
Figura 54: Hiperdocumento com formulário para cadastro/remoção de participantes em uma reunião.....	102
Figura 55: Página inicial do <i>eMeet/SERVE</i>	115
Figura 56: Página inicial da Seção Reuniões.	116
Figura 57: Campos para cadastro de novas reuniões.	117
Figura 58: Lista de reuniões que o usuário poderá acessar.	118
Figura 59: Janela com informações sobre a reunião.	118
Figura 60: Página inicial da Seção Salas.....	121
Figura 61: Campos para cadastro de uma nova sala.	121
Figura 62: Página de confirmação de remoção de uma reunião.	122
Figura 63: Página inicial da Seção Usuários.....	123
Figura 64: Campos para cadastro de um novo usuário.	123
Figura 65: Campos para envio de mensagens aos autores e mantenedores do <i>eMeet/SERVE</i>	125
Figura 66: Código de criação das tabelas do <i>eMeet/SERVE</i> para MySQL.	127

Índice de Tabelas

Tabela 1: A classificação dos sistemas apresentados dentro do <i>framework</i> proposto por Ellis et al. (1991).	46
Tabela 2: Dicionário de dados do <i>eMeet/SERVE</i>	63

Resumo

A literatura apresenta muitos sistemas que foram desenvolvidos a fim de automatizar o processo de geração de hiperdocumentos multimídia a partir de experiências ao vivo; ambientes educacionais e de reunião são os domínios de aplicação mais comuns. Conforme reportados, os sistemas não contemplam características importantes relativas à evolução de hiperdocumentos Web, tais como a possibilidade de atualizações constantes de conteúdo e de *layout*, e a independência de plataforma. Este trabalho visa a implementação do sistema *eMeet/SERVE*, que realiza a geração automática de hiperdocumentos multimídia a partir de informações capturadas em ambientes de reuniões distribuídas na Web. O *eMeet/SERVE* foi implementado com a utilização da arquitetura LAIAc – proposta neste trabalho – habilitada a suportar a evolução dos hiperdocumentos gerados. As interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE* foram desenvolvidas a partir de padrões e regras propostas na literatura. Além disso, um mecanismo de sincronização foi implementado nos hiperdocumentos a fim de permitir uma reprodução adequada dos elementos multimídia. A arquitetura LAIAc pode ser utilizada por outros sistemas associados à geração automática de hiperdocumentos. Com a realização deste trabalho, o *eMeet/SERVE* está preparado para integração com outros serviços implementados no contexto do projeto maior em que se insere, o Projeto InCA-SERVE, visando a evolução contínua dos hiperdocumentos gerados automaticamente.

Abstract

The literature discusses many software systems developed to automate the process of generating multimedia hyperdocuments from live experiences; educational and meeting environments are the most common application domains. As reported, those systems do not present features considered important towards the evolution of hyperdocuments associated to live experiences, such as support to constant updates of contents or changes in layout as well as independence of platform. The work reported in this document aims at developing the *eMeet/SERVE* application, a system that allows the automatic generation of multimedia hyperdocuments by means of captured information from distributed meetings environments on the Web. The *eMeet/SERVE* system is implemented on top of the LAIAc architecture – also proposed in this work – an architecture enabled to support the evolution of the hyperdocuments. The presentation interfaces of the hyperdocuments have been developed according to state of the art heuristics and guidelines. Moreover, a synchronization mechanism has been embedded in the hyperdocuments to allow the appropriate playback of multimedia objects. The LAIAc architecture itself can be used by other applications associated to the automatic generation of hyperdocuments. The *eMeet/SERVE* is part of the InCA-SERVE Project and, as implemented, it is ready to be integrated with other services implemented in the context of that project; the overall goal is the support to the evolution of the hyperdocuments generated automatically.

Capítulo 1. Introdução

1.1. Contexto

A literatura apresenta vários sistemas desenvolvidos a fim de automatizar o processo de geração de hiperdocumentos a partir de informação capturada em experiências ao vivo. Esse procedimento tem como objetivo realizar a documentação dessas experiências automaticamente, visando (a) deixar os participantes livres da necessidade de realizar a documentação de experiências de forma manual e (b) permitir que as informações capturadas e documentadas, através da geração de hiperdocumentos, possam ser acessadas e reutilizadas posteriormente pela Web¹. Sistemas que geram hiperdocumentos automaticamente a fim de apresentar informações aos usuários são denominados genericamente, neste documento, por *sistemas de geração de hiperdocumentos*.

Porque hiperdocumentos são gerados automaticamente, os problemas clássicos da tecnologia de hipertexto – a sobrecarga cognitiva imposta a autores e a possibilidade de desorientação por parte de leitores (Conklin, 1987; Nielsen, 1993) – são abordados diferentemente. Os participantes da sessão ao vivo são geradores do conteúdo capturado, mas o processo de autoria da estrutura hipertexto é realizado automaticamente depois de modelado pelos desenvolvedores da aplicação. Além disso, as alternativas de navegação disponibilizadas aos usuários normalmente têm bastante consistência, uma vez que estão fortemente vinculadas ao processo de captura.

Um exemplo de sistema de geração de hiperdocumentos é o sistema *eClass*² (Abowd, 1999; Abowd et al., 1998a; Abowd et al., 1998b), que vem sendo desenvolvido no *Georgia Institute of Technology* desde 1995 e conta com a colaboração de pesquisadores do ICMC/USP desde 1998. O *eClass* possui um ambiente de sala de aula instrumentado com dispositivos de captura e de autoria de hiperdocumentos multimídia (Brotherton et al., 1998) através do uso da tecnologia de computação ubíqua. A computação ubíqua é baseada no princípio de que as

¹ Neste texto, a World Wide Web está sendo referenciada por WWW ou simplesmente Web.

² Anteriormente chamado de *Classroom 2000*.

inovações tecnológicas devem se adaptar ao ambiente natural de usuários (Abowd et al., 1998a). Segundo Weiser (1993), a computação ubíqua tem como objetivo aprimorar o uso da computação ao fazer com que muitos dispositivos computacionais estejam espalhados em um ambiente físico de modo transparente ao usuário. Durante a aula, o sistema é ativado e realiza a captura ao vivo e o armazenamento das informações, que compreendem áudio, vídeo e anotações realizadas pelo professor. Ao término da aula, hiperdocumentos são gerados automaticamente para posterior consulta da informação capturada (Pimentel & Abowd, 1999). Como o material e as anotações do professor estarão disponíveis para posterior acesso, os alunos podem gastar menos tempo nas suas anotações e ficar mais atentos durante a aula (Abowd, 1999).

Ainda dentro do contexto educacional, pesquisadores da Universidade de Cornell, New York, desenvolveram o Cornell Lecture Browser (Mukhopadhyay & Smith, 1999). Esse sistema realiza a sincronização de *slides* com os *streams*³ de áudio/vídeo automaticamente durante o processo de geração de hiperdocumentos.

Pesquisadores da Microsoft também desenvolveram sistemas capazes de realizar a captura e posterior apresentação de informações multimídia através de hiperdocumentos. O TELEP (Jancke et al., 2000) e o MSTE (He et al., 2000) são exemplos de sistemas que podem ser aplicados para o treinamento de funcionários dentro de uma empresa.

Sistemas computacionais de suporte a reuniões foram desenvolvidos com a finalidade de apoiar as constantes reuniões existentes em qualquer tipo de empresa. Reuniões têm por objetivo apresentar, discutir e tomar decisões sobre assuntos genéricos. Naturalmente, os participantes de reuniões estão acostumados a utilizar os materiais tradicionais de uma reunião, como, por exemplo, papel, caneta e lousa. Sistemas computacionais de suporte a reuniões podem utilizar a tecnologia de computação ubíqua (Weiser, 1993). Dessa forma, os participantes de uma reunião podem se reunir da maneira tradicional e, ao mesmo tempo, utilizar um sistema de suporte a captura das informações necessárias, que serão disponibilizadas posteriormente.

Um dos trabalhos pioneiros que visa a captura de reuniões tradicionais foi desenvolvido pelos pesquisadores dos laboratórios da Xerox PARC (Minneman et al., 1995). O Notelook (Chiu et al., 1999) também visa a captura de reuniões tradicionais, gerando hiperdocumentos multimídia que podem ser acessados posteriormente. Pesquisadores do *Georgia Institute of Technology*, em parceria com a IBM, desenvolveram o TeamSpace (Richter et al., 2001), que é

³ *Streams* são fluxos contínuos de informações, geralmente utilizados para a difusão de áudio e vídeo.

um sistema utilizado para captura e apresentação de informações multimídia em reuniões distribuídas.

Alguns sistemas que foram construídos visando o contexto educacional, como o *eClass* e o *MSTE*, também têm sido utilizados no contexto de reuniões e apresentam limitações nesse sentido, uma vez que não foram desenvolvidos para serem utilizados nesse contexto.

1.2. Motivação

Entre os sistemas de captura de informações multimídia a partir de experiências ao vivo reportados na literatura, verifica-se a tendência de disponibilizar tais informações através de hiperdocumentos na Web. Conforme reportados, os sistemas não contemplam características importantes relativas à evolução de hiperdocumentos Web, tais como a possibilidade de atualizações constantes de conteúdo e de *layout*.

1.3. Objetivo

O objetivo deste trabalho aqui reportado foi implementar um sistema de geração automática de hiperdocumentos multimídia a partir de informações capturadas em ambientes de reuniões distribuídas na Web, sistema esse fundamentado em uma arquitetura habilitada a suportar a evolução dos hiperdocumentos gerados.

1.4. Resultados Alcançados

Como resultados alcançados neste trabalho, destacam-se:

1. Definição de uma arquitetura de suporte a sistemas de geração de hiperdocumentos, denominada LAIAc;
2. Implementação do *eMeet/SERVE* para acesso, recuperação e apresentação das informações capturadas em reuniões distribuídas. Esse sistema foi implementado com uso da arquitetura LAIAc;
3. Documentação da utilização de padrões para o desenvolvimento da interface visual (*layout*) dos hiperdocumentos gerados pelo *eMeet/SERVE*;

4. Implementação de um mecanismo de sincronização entre elementos multimídia em hiperdocumentos que leva em conta alguns requisitos importantes de um sistema Web como, por exemplo, a compatibilidade entre diferentes navegadores e o desempenho do mecanismo de sincronização.

A partir dos resultados alcançados por este trabalho, muitos outros trabalhos de melhoria contínua do *eMeet* poderão ser realizados. Além disso, outros sistemas de geração de hiperdocumentos poderão utilizar a arquitetura LAIAc proposta neste trabalho.

1.5. Estrutura

Este capítulo apresentou o contexto no qual este trabalho se insere, a motivação para seu desenvolvimento, objetivos propostos e resultados alcançados. O Capítulo 2 descreve a infraestrutura InCA-SERVE, o sistema *eClass* de forma detalhada e apresenta o sistema *eMeet*. O Capítulo 3 apresenta uma visão geral dos conceitos de computação ubíqua do ponto de vista de trabalho cooperativo. O Capítulo 4 discute e classifica sistemas e técnicas existentes na literatura relacionados ao *eMeet*. O Capítulo 5 apresenta os requisitos e especificações do *eMeet/SERVE*, a arquitetura LAIAc, bem como sua utilização na implementação do *eMeet/SERVE*. O Capítulo 6 discute os fatores de usabilidade e interação das interfaces visuais dos hiperdocumentos gerados pelo *eMeet/SERVE*, bem como o mecanismo de sincronização entre as mídias em hiperdocumentos. Além disso, também são discutidos os níveis de controle de acesso dos usuários aos hiperdocumentos gerados. Por fim, o Capítulo 7 apresenta as conclusões, a discussão dos resultados, as contribuições e os trabalhos futuros.

Capítulo 2. InCA-SERVE

2.1. Considerações iniciais

Este trabalho foi desenvolvido no ICMC/USP no contexto do projeto InCA-SERVE o qual, realizado em parceria com o *College of Computing* do *Georgia Tech* (*CC-Gatech*), é apoiado pelo programa de cooperação internacional entre o PROTEM-CC/CNPq e o CISE/NSF (Pimentel & Abowd, 1999).

O Projeto InCA-SERVE tem como objetivos dar suporte à informação hipermídia evolucionária para os domínios de ensino e de reuniões de grupos de projeto de software (Pimentel & Abowd, 1999).

Cada parte dessa infra-estrutura (InCA e SERVE), além do *eClass* e do *eMeet*, são discutidas a seguir.

2.2. *eClass*

O *eClass* é um projeto que vem sendo desenvolvido desde 1995 no *Georgia Tech* e, desde 1998, conta com a colaboração de pesquisadores do ICMC/USP. O ambiente disponibilizado no projeto explora os conceitos propostos pela computação ubíqua para capturar experiências ao vivo de aulas. Ao término da aula, o *eClass* gera hiperdocumentos que são disponibilizados e acessados através da Web.

A sala de aula do *eClass* é um ambiente instrumentado com hardware e software. Do ponto de vista de hardware, o ambiente *eClass* necessita de uma sala equipada com tecnologias de computação ubíqua, que é um ambiente computacional pouco intrusivo – uma coleção de lousas eletrônicas, projetores, microfones, câmeras de vídeo e computadores em rede, conforme ilustrado na Figura 1.

Do ponto de vista de software, o ambiente *eClass* é um conjunto de aplicações cliente/servidor. Um dos módulos do *eClass*, chamado *Zen*, realiza a captura de uma aula tradicional na sala de aula apresentada na Figura 1. Existem outros módulos de software

responsáveis pelo armazenamento e geração automática de hiperdocumentos baseados na infraestrutura Web (Pimentel et al., 2000). Utilizando codificadores e servidores da Real™, informações de áudio e vídeo são capturadas e disponibilizadas como ligações no hiperdocumento gerado.

Figura 1: Dispositivos do eClass - (A) lousa eletrônica, (B) câmeras de vídeo, (C) microfones, (D) projetor com navegação Web e (E) projetor com informações anteriormente apresentadas na (A).

A Figura 2 apresenta um exemplo de um hiperdocumento gerado automaticamente após uma sessão de captura. A interface utilizada divide a tela em quadros (*frames*) para facilitar a visualização da informação. O *frame* da direita apresenta uma aula como uma seqüência de *slides*: cada *slide* contém as anotações realizadas pelo professor na lousa eletrônica durante a aula. O sistema disponibiliza diferentes interfaces que permitem acesso aos *slides* de uma aula, juntamente com o áudio e/ou vídeo associado. Usuários com conexões rápidas podem utilizar o conjunto de interfaces que apresentam vídeo, enquanto que, para usuários com conexões lentas, é disponibilizado um conjunto de interfaces que utiliza unicamente áudio como mídia contínua associada. O oferecimento de várias outras interfaces foi investigado (Pimentel et al., 2001b), entre elas interfaces que dão acesso a serviços de busca, a lista de aulas de cada curso ou de todos os cursos.

Cada aula é gravada como um único fluxo de áudio ou vídeo, e a interface de acesso disponibiliza diferentes formas para indexação dos eventos importantes ocorridos no decorrer da sessão, de forma a possibilitar acesso randômico ao conteúdo capturado. Por exemplo, acima de cada *slide* mostrado na Figura 2, existe uma ligação para a parte do vídeo associada à sua apresentação, e as anotações feitas pelo instrutor na lousa eletrônica também servem como indexadores para os fluxos de mídia contínua (Pimentel et al., 2001b).

Figura 2: Hiperdocumento multimídia gerado automaticamente pelo *eClass*

O *eClass* tem seu *workflow* baseado em quatro fases (Abowd et al., 1998a), de acordo com a Figura 3:

- Pré-produção: antes da aula, o professor pode realizar a preparação dos *slides*, ou até de hiperdocumentos Web. O professor pode aproveitar-se do uso de imagens (por exemplo, geradas de apresentações PowerPointTM), para preparar o material. Tal material será carregado e exibido na lousa eletrônica do sistema.
- Gravação e captura: quando a aula começa, grava-se o máximo de informações possíveis, como as anotações⁴ realizadas pelo professor na lousa eletrônica, o vídeo e o áudio da sala de aula. A troca de *slides* e as anotações do professor são dados importantes para a sincronização do material montado com o áudio e o vídeo do sistema.
- Pós-produção ou integração: quando a aula termina, todas as informações que foram capturadas na aula são processadas automaticamente, gerando os documentos necessários para a fase de acesso. Isso inclui novas imagens, informações de sincronização, transição entre *slides*, dentre outras. Nas primeiras versões do *eClass*,

⁴ No contexto de uma sessão de captura, o termo anotações é utilizado para fazer referência ao que o professor escreveu na lousa eletrônica, durante a aula, utilizando uma caneta eletrônica. Esse termo é utilizado neste documento.

havia algum trabalho necessário para integrar a captura das anotações do professor com os *streams* de áudio e vídeo da sala de aula. Porém, com a evolução da ferramenta, hoje esse processo é realizado automaticamente pelo próprio *eClass* (Pimentel & Abowd, 1999).

- Acesso: nessa fase, estudantes e professores podem acessar as aulas passadas através de interfaces Web.

Figura 3: *Workflow* do *eClass* em relação ao tempo. Com a evolução do *eClass*, a fase de integração é realizada automaticamente por software (Pimentel & Abowd, 1999). Dessa forma, as informações para fase de acesso são disponibilizadas rapidamente.

Durante a fase de captura, alguns fatores são interessantes e precisam ser ressaltados para um melhor entendimento do sistema. É necessário, por exemplo, que a lousa eletrônica utilizada possua uma grande superfície interativa para que o professor possa ter espaço suficiente para escrever durante a aula tudo o que ele realmente precisa. Outro fator interessante é que cada novo *stream* de informação que é capturado, como mais um canal de vídeo, por exemplo, requer uma intervenção humana para iniciar a gravação. Além disso, deve-se ter certeza de que as mídias estão sincronizadas. Outro fator que é desejável no *eClass* é a captura das atividades dos alunos durante a aula, pois tais informações podem ser utilizadas pelo sistema em implementações futuras (Pimentel & Abowd, 1999).

Já na fase de integração, deve-se definir o tipo de controle que será utilizado para indexar os *streams* de áudio/vídeo com informações capturadas durante a aula, como, por exemplo, a transição entre *slides*. No caso do *eClass*, as anotações do professor na lousa eletrônica são utilizadas para indexação, pois para cada vez que o professor realiza uma anotação, há a geração de uma ligação automática com o *stream* de áudio/vídeo. Porém, deseja-se também que a busca dentro desse material seja feita por outros tipos de *streams*, como, por exemplo, o áudio. Foi desenvolvido um protótipo de uma ferramenta que converte o áudio em fonemas, deixando que

os usuários procurem por palavras-chave para localizar um determinado trecho da aula (Pimentel & Abowd, 1999).

Na fase de acesso, atualmente, os dados estão sendo exibidos através de hiperdocumentos HTML. Há a intenção de utilização do Java, na forma de *applets*, em um futuro próximo para que exista a opção de uma melhor visualização dos dados, como, por exemplo, a animação da própria escrita das anotações de acordo com o áudio/vídeo da aula.

2.3. A infra-estrutura InCA

No contexto das infra-estruturas InCA e SERVE em desenvolvimento, a infra-estrutura InCA – *Infrastructure for Capture and Access* – que é responsável pela captura e integração de atividades interativas genéricas, é a infra-estrutura de mais baixo nível. Além de suportar a captura de atividades realizadas em sessões ao vivo, a InCA deverá disponibilizar um modelo de informação extensível para a captura de experiências interativas, o qual poderá ser utilizado para integrar atividades que ocorrem fora da sessão de captura.

No *eClass*, essa infra-estrutura foi implementada pelos pesquisadores do *Georgia Tech* para suportar o contexto das atividades envolvidas dentro da sala de aula. O *eClass* é regularmente utilizado para captura de reuniões, mas apresenta limitações, uma vez que não foi construído para esse domínio. A infra-estrutura InCA deve corresponder a uma infra-estrutura escalável⁵ e expansível⁶ para ambientes de captura e acesso à informação capturada.

2.4. A infra-estrutura SERVE

Explorando os serviços da InCA, a infra-estrutura SERVE será responsável por armazenar (*Store*), estender (*Extend*), recuperar (*Retrieve*) e visualizar (*Visualize*) informação evolucionária⁷ (*Evolutionary*). Essa infra-estrutura deverá apresentar a informação capturada aos usuários bem como suportar serviços especializados oferecidos fora da sessão de captura.

⁵ Ser uma infra-estrutura escalável significa poder ser utilizada por um maior número de usuários de forma simultânea ou não.

⁶ Ser uma infra-estrutura expansível significa poder receber novas funcionalidades sem necessidade de reformulação da mesma.

⁷ Informações evolucionárias são informações que estão sendo modificadas constantemente através de seu próprio reuso ou de agregação de novas informações.

No *eClass*, essa infra-estrutura foi desenvolvida, mas ainda não apresenta nada relacionado ao suporte a reuniões.

2.5. *eMeet*

O sistema implementado neste trabalho, juntamente com o trabalho de Bila (2002), denominado *eMeet*, apresenta características semelhantes ao *eClass*, porém possui infra-estrutura adequada para realizar o suporte a reuniões distribuídas. Nesse processo inicial, que envolveu a implementação básica do *eMeet*, dois trabalhos de mestrado estiveram envolvidos. A Figura 4 mostra o *eMeet* como parte da infra-estrutura InCA-SERVE e a divisão do trabalho entre os responsáveis pelo projeto.

Figura 4: A infra-estrutura InCA-SERVE implementada para suporte a reuniões distribuídas.

O sistema de captura e transmissão das informações, denominado de *eMeet/InCA*, foi implementado por Wagner da Silva Bila (Bila, 2002). O *eMeet/InCA* captura e armazena as informações que são recuperadas, processadas e apresentadas na forma de hiperdocumentos pelo sistema implementado por este trabalho, denominado de *eMeet/SERVE*. A partir da Figura 4,

conclui-se que a interface principal de comunicação entre as partes InCA e SERVE do *eMeet* é o banco de dados. O banco de dados atualmente utilizado é o MySQL. Os dados poderiam estar armazenados em outros bancos de dados ou utilizando outras formas de armazenagem.

O *eMeet/InCA* (Bila, 2002), ilustrado Figura 5, é o ambiente responsável pela comunicação síncrona entre os participantes durante as reuniões. Com ele, os participantes podem visualizar e ouvir uns aos outros através dos vídeos e áudios transmitidos entre suas estações de trabalho. Durante a reunião, os participantes dispõem de uma lousa eletrônica compartilhada, que pode ser utilizada para a realização de anotações compartilhadas. Após a reunião, o *eMeet/InCA* armazena todas as anotações sob a forma de *slides* e faz a mixagem dos *streams* de áudio/vídeo dos participantes em um único canal⁸. Todas essas informações que foram capturadas durante a realização da reunião e seus respectivos metadados de controle são armazenados em um servidor central.

Figura 5: Interface do *eMeet/InCA* dos participantes de uma reunião distribuída. Os *slides* e os áudios/vídeos dos participantes são visualizados em tempo real, durante a reunião.

A partir das informações capturadas e armazenadas no servidor central pelo *eMeet/InCA*, o *eMeet/SERVE* é o responsável pelo acesso e apresentação das informações capturadas nas reuniões (áudio, vídeo e *slides*). Os dados recuperados do servidor são processados e

⁸ A mixagem dos *streams* de áudio/vídeo de cada participantes em um único canal é feita através do volume do áudio de cada *stream*, através de um algoritmo implementado por Bila (2002).

transformados automaticamente em hiperdocumentos HTML que, através da Web, são enviados sob-demanda para os usuários. Tais hiperdocumentos estão disponíveis após o término da reunião. A Figura 6 ilustra um hiperdocumento multimídia gerado pelo *eMeet/SERVE*. A apresentação das informações capturadas nos hiperdocumentos multimídia é feita de forma sincronizada. Mais informações sobre o *eMeet/SERVE* encontram-se nos Capítulos 5 e 6.

Figura 6: Exemplo de hiperdocumento multimídia gerado pelo *eMeet/SERVE*. A área central apresenta a seqüência de *slides* capturados durante a reunião. Na parte inferior, encontra-se a linha do tempo. O vídeo e seus controles estão abaixo do menu lateral esquerdo.

Os hiperdocumentos gerados pelo *eMeet/SERVE* apresentam interfaces visuais que foram desenvolvidas para permitir boa interação usuário-computador. Regras de usabilidade fornecidas por Netto (2002) serviram como base para o desenvolvimento dessas interfaces. Os aspectos de interação, estrutura e conteúdo dos hiperdocumentos gerados são discutidos no Capítulo 6.

O *eMeet/SERVE* também possui controle de acesso próprio aos hiperdocumentos com informações das reuniões. Usuários ou grupos de usuários distintos podem ter acesso a informações distintas de acordo com o seu perfil de acesso. Isso evita que informações privadas

sejam acessadas por pessoas não autorizadas (mais detalhes sobre o controle de acesso do *eMeet/SERVE* encontram-se na Seção 6.5).

2.6. Considerações finais

O *eMeet*, sistema implementado neste trabalho, em conjunto com outro trabalho de mestrado (Bila, 2002), é um sistema que estende os conceitos do *eClass* (Abowd, 1999; Abowd et al., 1998a; Abowd et al., 1998b), dando suporte à captura, transmissão, armazenamento, recuperação e posterior acesso via Web das informações capturadas durante reuniões distribuídas ou não.

O *eMeet* possui duas partes complementares, denominados *eMeet/InCA* (Bila, 2002) e *eMeet/SERVE*. Um dos objetivos deste trabalho foi a implementação do *eMeet/SERVE* (mais detalhes sobre o *eMeet/SERVE* encontram-se nos Capítulos 5 e 6). Atualmente, a principal interface de comunicação entre o *eMeet/InCA* e o *eMeet/SERVE* é o banco de dados MySQL.

O *eMeet* foi desenvolvido dentro da infra-estrutura InCA-SERVE e poderá ser futuramente estendido por outros trabalhos, dentro do projeto de cooperação internacional entre ProTeM-CC/CNPq/Brasil e NSF/EUA (Pimentel & Abowd, 1999).

Capítulo 3. Computação Ubíqua do ponto de vista de trabalho cooperativo

3.1. Considerações iniciais

A tecnologia de computação ubíqua é utilizada nos mais diversos domínios de aplicação. Em muitas aplicações, infra-estruturas que utilizam tecnologias de computação ubíqua capturam informações para posterior acesso. Hiperdocumentos multimídia estão sendo cada vez mais utilizados para apresentar a informação capturada.

O trabalho cooperativo suportado por computador (CSCW – *Computer-Supported Cooperative Work*) vem sendo explorado em sistemas hipermídia. Ellis et al. (1991) propõem uma classificação para sistemas CSCW, separando-os em quadrantes de acordo com os fatores tempo e lugar. Tal classificação é apresentada neste capítulo, juntamente com alguns estudos da tecnologia de computação ubíqua sob o ponto de vista de trabalho cooperativo.

3.2. Hipertexto e trabalho cooperativo

3.2.1. Origem

Bush (1945) foi o primeiro a propor associação como forma de armazenar e recuperar o conhecimento que é desenvolvido em pesquisas e investigações. Esse mecanismo, que seria parte de um dispositivo denominado Memex, foi o primeiro da história a utilizar o conceito de hipertexto.

Engelbart (1984a; 1984b) foi o primeiro a desenvolver um sistema CSCW, chamado NLS/Augment, que suportava a cooperação síncrona e assíncrona sobre materiais hipermídia. Os usuários do sistema podiam editar o conteúdo em paralelo, e até iniciar uma conferência à distância, compartilhando uma janela entre diversos computadores.

Outro sistema hipertexto importante, o *Knowledge Management System* (KMS) (Akscyn et al., 1988) permitia a edição de *frames* por múltiplos usuários. O KMS fazia controle de edição simultânea através de um controle de concorrência otimizado, detectando conflitos que os usuários poderiam solucionar manualmente.

Ainda antes do surgimento da World Wide Web, um outro sistema a dar suporte à cooperação entre múltiplos usuários na edição de materiais compartilhados foi o Intermedia (Meyrowitz et al., 1986). Esse sistema foi o primeiro a suportar estruturas hipermídia externas aos seus documentos. Além disso, o Intermedia também foi o primeiro a introduzir acesso a anotações, além dos tradicionais acessos à leitura e escrita. Com isso, os usuários que tinham acesso às anotações estavam aptos a criar ligações a documentos que não tinham direito de escrita. Com isso, o conceito de navegação em hiperdocumentos, implementados pelos sistemas hipertexto tradicionais, tornou-se utilizado em larga escala.

O Notecards (Halasz, 1987) foi um outro sistema importante em CSCW, que utilizando o conceito de cartões, chamados de colaboradores ou históricos, davam suporte à coordenação entre os autores que se alternavam na edição de um material (Trigg and Suchman, 1989).

Halasz (1988), em seu importante artigo, levantou o fato de que muitos aspectos de cooperação não podem ser suportados tecnicamente, mas podem ser controlados por convenções sociais que estão fora do escopo dos sistemas hipermídia. Halasz sugeriu que por mais que um sistema cooperativo seja eficiente, devem existir regras e convenções que auxiliem a interação humana, ou seja, permitir que pessoas trabalhem de forma cooperativa vai, normalmente, além das funcionalidades de um sistema.

Engenharia, arquitetura, projeto de um sistema, administração e autoria são alguns exemplos de domínios de aplicação dos sistemas cooperativos, onde pessoas trabalham individualmente, mas contribuem em prol de uma tarefa comum e geralmente produzem muita documentação. Todos esses trabalhos envolvem uma grande comunicação e coordenação.

3.2.2. Um *framework* para classificação (Ellis et al., 1991)

Dentro dos sistemas hipermídia CSCW, os usuários cooperadores podem estar separados por tempo, ou seja, contribuem em horários diferentes. Tais usuários também podem estar separados fisicamente, ou seja, contribuem em lugares diferentes. Com isso, podemos observar o

framework clássico, proposto por Ellis et al. (1991), que caracteriza o suporte computacional dentro dos sistemas CSCW. Tal *framework* está ilustrado na Figura 7.

	Mesmo local	Locais diferentes
Mesmo tempo (síncrono)	1° quadrante	2° quadrante
Tempos diferentes (assíncrono)	3° quadrante	4° quadrante

Figura 7: *Framework* clássico para classificar sistemas cooperativos.

Observando o *framework* clássico na Figura 7, temos os seguintes quadrantes:

- 1° quadrante: usuários que cooperam no sistema estão presentes no mesmo local ao mesmo tempo. Exemplo: reunião de um grupo em uma sala com computadores, onde todo o grupo participa e coopera simultaneamente.
- 2° quadrante: usuários que cooperam no sistema estão em locais diferentes, porém estão trabalhando ao mesmo tempo. Exemplo: um grupo que possui integrantes em cidades diferentes e eles se conectam simultaneamente para trabalhar.
- 3° quadrante: usuários que cooperam no sistema estão presentes no mesmo local, porém em tempos diferentes. Exemplo: cada pessoa de um grupo de trabalho coopera no mesmo local (sala com computadores), porém em horários diferentes.
- 4° quadrante: usuários que cooperam no sistema estão em locais diferentes e ainda trabalham em tempos diferentes. Exemplo: um grupo que possui integrantes em cidades diferentes e eles se conectam em tempos também diferentes para trabalhar.

Esse *framework* pode ser utilizado para classificar as funcionalidades existentes em cada tipo de sistema CSCW, dentro de cada um dos quadrantes. Isso nos diz como o sistema suporta um trabalho cooperativo baseado em materiais compartilhados.

Não é difícil de imaginar um sistema hipermídia que apareça em mais de um quadrante. De fato, muitos sistemas já existentes permitem que os usuários trabalhem de forma cooperativa de diversas maneiras, pertencendo a mais de um quadrante. Por exemplo, o *eClass*, citado anteriormente, pertence ao primeiro, terceiro e quarto quadrantes. O *eClass* pertence ao primeiro quadrante, pois durante a aula, de forma síncrona e local, o sistema realiza a captura das informações. O *eClass* também pertence ao terceiro e quarto quadrantes, pois em tempos

diferentes, grupos em um mesmo local ou em locais diferentes se reúnem para acessar as informações capturadas. Porém, o *eClass* não pertence ao segundo quadrante, pois durante a aula, o sistema tem atuação somente no local da aula e não em locais diferentes.

Outras classificações que podem ser feitas em sistemas CSCW podem ser encontradas na literatura. Um exemplo é a classificação proposta por Grønbaek & Trigg (1999), que leva em conta como usuários coordenam o trabalho em materiais compartilhados.

Para a realização de estudos dos sistemas atuais, a classificação por quadrantes (Ellis et al., 1991) foi adotada para comparar os sistemas estudados. Tais sistemas são discutidos no próximo capítulo.

3.3. Computação Ubíqua

3.3.1. Origem

O objetivo principal da computação ubíqua é permitir que a tecnologia esteja presente dentro do ambiente que cerca o usuário, sem fazer com que o usuário modifique o seu comportamento natural (Abowd et al., 1998a). Sendo assim, o usuário não tem que se deslocar para um ambiente próprio de tecnologia para fazer uso dela.

A tecnologia de computação ubíqua pode ser explorada para gravar situações diárias e fazer com que elas estejam disponíveis para eventuais necessidades. Sendo assim, tal tecnologia pode ser utilizada para gravar momentos diários de uma pessoa, apesar de apenas uma pequena parte dessa informação ser fundamental para a realização de uma determinada tarefa. Uma analogia a esse comportamento pode ser feita, por exemplo, com o *backup* de dados de um servidor, onde os usuários geralmente precisam de apenas um determinado conjunto de arquivos.

A tecnologia de computação ubíqua propõe que muitos computadores possam ser utilizados em ambientes físicos, deixando-os transparentes para os usuários (Weiser, 1993).

Estudos antropológicos sobre a vida das pessoas no trabalho (Lave, 1991⁹; Suchman, 1985¹⁰) *apud* (Weiser, 1993) indicam que o computador é destinado a trabalhos específicos e não integra o dia-a-dia do trabalho de muitas pessoas. Em outras palavras, o computador é visto

⁹ (Lave, 1991) Lave, J. (1991) Situated learning: legitimate peripheral participation. *Cambridge University Press*. Cambridge. New York, N.Y. 1991.

¹⁰ (Suchman, 1985) Suchman, L.A. (1985) Plans and situated actions: the problem of Human-Machine Communication. *Xerox Parc Tech. Rep. Isl-6*, Feb. 1985.

como um “centro de atenções” e não como um outro componente qualquer dentro do ambiente de trabalho normal de uma pessoa. O trabalho das pessoas pode melhorar se o computador deixar de ser um “centro de atenções” e passar a estar inserido no ambiente normal de trabalho, deixando as pessoas se preocuparem com o que elas têm a fazer, sem modificar suas atitudes por causa da presença da tecnologia.

3.3.2. Experiências com capturas em dispositivos de superfícies interativas

Pesquisas relacionadas à tecnologia de computação ubíqua foram iniciadas pelos pesquisadores do Xerox Palo Alto Research Center (PARC) em 1998 para dar suporte a uma nova geração de dispositivos ubíquos onde as pessoas estariam interagindo com centenas de computadores sem fio (*wireless*) interconectados.

Por razões práticas, as pesquisas foram iniciadas com dispositivos de três diferentes tamanhos e que possuem superfície interativa, conforme classificação de Weiser (1993):

- Lousa eletrônica (*Board*): tamanho grande. Pode ser comparada à lousa de escritórios e escolas, ou a um mural.
- Prancheta eletrônica (*Pad*): tamanho médio. Tem tamanho similar aos computadores portáteis atuais. Eles têm o papel de substituir anotações feitas em folhas de papel, por exemplo.
- Caderneta eletrônica (*Tab*): tamanho pequeno. Assemelha-se a um *palmtop* e registra notas individuais. Pode substituir papeis de anotação, como os *post-it*¹¹.

É importante observar que nenhum desses equipamentos são computadores pessoais. Na realidade, são equipamentos que podem substituir outros já utilizados no dia-a-dia das pessoas. Pesquisadores do Xerox PARC acreditam que no futuro, para cada pessoa em um escritório, existirão centenas de cadernetas eletrônicas, dezenas de pranchetas eletrônicas e uma ou mais lousas eletrônicas. A Figura 8 mostra um exemplo de (a) uma caderneta eletrônica e (b) uma lousa eletrônica.

A lousa eletrônica utilizada nas pesquisas realizadas nos laboratórios Xerox PARC é muito semelhante à lousa eletrônica que foi utilizada no *eClass*. A lousa eletrônica desenvolvida no

¹¹ *Post-it* são pequenos papéis adesivos que pessoas utilizam para realizar anotações.

eMeet/InCA é um pouco diferente, pois ainda não é um dispositivo físico. Trata-se de um espaço para realização de anotações implementado no próprio software.

Figura 8: Dispositivos ubíquos com superfície interativa: (a) caderneta eletrônica e (b) lousa eletrônica.

Ainda dentro do centro de pesquisa dos laboratórios Xerox PARC, duas aplicações principais nesse contexto foram desenvolvidas: a localização de pessoas e o projeto¹² compartilhado.

O trabalho de localização de pessoas é realizado através de equipamentos ubíquos acoplados às pessoas e às salas. Um mapa de localização, que representa o ambiente onde as pessoas estão circulando, mostra a imagem de cada uma das pessoas no local exato do ambiente. Esse mapa é atualizado constantemente. Localizar pessoas pode ter grandes vantagens em um ambiente de trabalho, como, por exemplo, a transferência de ligações telefônicas para uma pessoa diretamente na sala onde ela está, agendamento de reuniões de acordo com as pessoas presentes, ou até para ir conversar com uma pessoa em particular (olhando se ela está sozinha no mapa) (Weiser, 1993).

O trabalho de projeto compartilhado é uma outra aplicação que vem sendo intensamente pesquisada dentro dos laboratórios Xerox PARC. Uma das ferramentas mais conhecidas é a Tivoli (Pedersen et al., 1993), que aceita com que duas ou mais pessoas compartilhem a mesma página, utilizando lousas eletrônicas e computadores pessoais em um ambiente de reuniões, e a editem conjuntamente. Tivoli também tem a característica de aceitar entradas (imagens) digitalizadas e de poder imprimir os resultados.

¹² Nesse contexto, o projeto é realizado em uma janela compartilhada por múltiplos usuários para livre edição.

3.3.3. Experiências com captura de áudio

Muito tempo da vida profissional e acadêmica das pessoas é gasto ouvindo e memorizando os eventos que cercam os seres humanos. Porém, quando tentamos nos lembrar de alguma parte dessa informação, nem sempre obtemos os resultados desejados. Isso ocorre porque o ser humano é ineficiente em memorizar os acontecimentos, principalmente quando existem muitos tipos diferentes de informações envolvidas. Portanto, a utilização da computação ubíqua é uma forma eficiente para suprir tal necessidade (Abowd & Mynatt, 2000)¹³.

Alguns trabalhos foram realizados para suprir tal necessidade, dentre eles: o Phone Slave (Schmandt & Arons, 1985) e as ferramentas Xcapture e Listener (Hindus & Schmandt, 1992). Essas ferramentas trabalham com a captura e recuperação de áudio.

O Phone Slave é um sistema que possui uma gravação que pode ser ouvida pelo telefone e que grava as respostas dos usuários. Na verdade, a gravação é uma série de perguntas espaçadas com um certo intervalo de tempo. Os usuários respondem as perguntas e cada resposta é armazenada pelo sistema de maneira semi-estruturada, pois se entende que a informação colhida após uma pergunta do sistema é a resposta daquela mesma pergunta.

A primeira ferramenta de Hindus & Schmandt (1992), chamada Xcapture, implementa um *buffer* circular que guarda informações de áudio recentes com um microfone ubíquo em um ambiente normal de trabalho. A ferramenta foi denominada pelos autores de “memória de curta duração” do usuário. Um cenário que exemplifica a necessidade dessa “memória de curta duração” acontece quando múltiplos autores estão trabalhando juntos para escrever um artigo e um deles diz uma nova maneira de escrever um parágrafo. Após isso, o outro autor diz para ele repetir a frase para que ela seja escrita, pois a frase é “perfeita”. Só que as palavras foram perdidas e o primeiro autor, muito dificilmente, se lembrará exatamente do parágrafo que acabou de dizer. Tal problema também foi apontado por Abowd & Mynatt (2000). Nesse exemplo, se o Xcapture estivesse ativo de forma ubíqua no ambiente e gravando a conversa, os autores poderiam ouvir, editar e salvar o áudio desejado dos últimos minutos, recuperando assim, o parágrafo dito.

A segunda ferramenta, chamada de Listener, captura e grava conversas de telefone. A ferramenta possui dois microfones: um é conectado diretamente na linha do telefone e grava a

¹³ Esse artigo é muito importante no contexto de computação ubíqua, pois faz uma revisão de muitos pontos fundamentais dessa tecnologia.

conversa das duas pessoas; o outro fica próximo ao telefone, gravando somente a voz da pessoa local. Com isso, o Listener consegue distinguir as falas e monta um gráfico, dependente do tempo, com as falas das pessoas representadas por blocos. Ao final da conversa, tudo o que foi dito estará representado por blocos na tela do computador e o usuário poderá editar e salvar as partes mais importantes, podendo inclusive transportá-las para outros aplicativos (desde que sejam compatíveis). Com essa ferramenta, as partes mais importantes das conversas podem ser gravadas e ouvidas a qualquer instante.

3.3.4. Experiências com captura de vídeo

Assim como a captura de áudio, a captura de vídeo pode suprir algumas necessidades do ser humano, uma vez que informação em vídeo também pode ser considerada outra fonte para suprir a “memória de curta duração” dos usuários.

Muitos sistemas, inclusive o *eClass* e o *eMeet*, realizam a captura de vídeo para posterior acesso. Alguns sistemas como o Cornell Lecture Browser (Mukhopadhyay & Smith, 1999), o TELEP (Jancke et al., 2000), o Notelook (Chiu et al., 1999) e o TeamSpace (Richter et al., 2001) realizam a captura de *streams* de vídeo de forma ubíqua. Tais sistemas são discutidos no próximo capítulo.

3.4. Considerações finais

Hiperdocumentos estão sendo cada vez mais utilizados para apresentar informações capturadas em sistemas que podem conter aspectos de tecnologia de computação ubíqua e de cooperação. Para esses novos sistemas, é desejável que as informações capturadas e disponíveis tenham possibilidade de acesso e navegação por parte dos usuários. Por exemplo, em um cenário educacional, ao reverem um ponto de uma aula, os alunos podem não desejar assistir a toda a aula novamente. Por isso, é necessário que existam meios alternativos para que usuários filtrem as informações desejadas. Alguns métodos são apresentados no próximo capítulo.

O *framework* proposto por Ellis et al. (1991), que divide os sistemas CSCW em quatro quadrantes de acordo com o tempo e localização física dos usuários, foi utilizado para classificar alguns sistemas estudados. Tais sistemas são apresentados no próximo capítulo.

Capítulo 4. Captura e apresentação de informação multimídia

4.1. Considerações iniciais

A captura e a apresentação de informação multimídia estão sendo exploradas em muitos sistemas – nem sempre com o uso da tecnologia de computação ubíqua – para o armazenamento e futuro acesso aos dados. Assim como o *eClass* e o *eMeet*, muitos sistemas utilizam o hipertexto para exibir a informação multimídia capturada. Dentre diversos sistemas estudados, alguns possuem similaridades e diferenças interessantes em relação ao *eClass* e ao *eMeet*. Neste capítulo, são apresentados alguns desses sistemas, sendo classificados de acordo com os quadrantes propostos por Ellis et al. (1991), além de outras técnicas estudadas que já fazem parte ou podem ser implementadas futuramente no *eMeet*.

4.2. Sistemas para captura e apresentação

4.2.1. Cornell Lecture Browser

O Cornell Lecture Browser (Mukhopadhyay & Smith, 1999), desenvolvido na Universidade de Cornell, New York, é um sistema que constrói documentos multimídia automaticamente a partir de uma apresentação ao vivo. Esse sistema tem algumas características que podem ser comparadas ao *eClass* e a outros sistemas de geração de hiperdocumentos.

Uma diferença entre o Lecture Browser e o *eClass*, por exemplo, é que o *eClass* é baseado em captura intrusiva (apesar de ser pouco intrusiva), enquanto o Lecture Browser, foi construído para ser baseado em captura passiva. A diferença é que, na captura intrusiva, o palestrante ou apresentador precisa realizar algumas ações específicas para ajudar no processo de captura, enquanto que um sistema de captura passiva opera sem auxílio do palestrante.

Uma semelhança entre o *eClass* e o Lecture Browser é a necessidade de um ambiente estruturado para captura. Ambos necessitam de uma sala contendo equipamentos para o

funcionamento do sistema. Um exemplo de um sistema que funciona em ambientes desestruturados é o Experience On-Demand (Christel et al., 1995), cujo objetivo é capturar experiências pessoais utilizando áudio e vídeo para criar uma agenda digital.

O funcionamento do Lecture Browser possui características interessantes. Duas câmeras de vídeo capturam a apresentação que é digitalizada e codificada no formato MPEG. Uma câmera é responsável por filmar a região que inclui os *slides*, permitindo que seja visualizado tanto o apresentador quanto os *slides* (isso é feito para futura sincronização). A segunda câmera segue os movimentos do palestrante através de um sistema e hardware próprios para isso, deixando o palestrante livre para se deslocar dentro de um espaço previamente delimitado.

Ao final da apresentação, ambos os canais de vídeo são transmitidos através da rede para um servidor, chamado servidor de processamento. Esse servidor também recebe os *slides* que foram apresentados. O servidor faz a combinação entre os vídeos, *slides* e utiliza o HTML para criar uma apresentação que pode ser vista por qualquer navegador conectado à Internet. Como apresentado na Figura 9, os hiperdocumentos HTML mostram os *slides* em sua parte principal, juntamente com um índice, que contém o título do *slide* e que pode ser selecionado pelo usuário a qualquer momento. Na interface também existem botões como próximo e anterior para navegar entre *slides*, além de conter também o vídeo da apresentação. A parte inferior do hiperdocumento contém uma linha do tempo (*timeline*), que indica o andamento da apresentação e permite mais uma forma de interação com o usuário. A linha do tempo mostra a duração de cada *slide*, além de indicar o título de cada um deles quando o usuário posiciona o cursor do mouse sobre cada parte da linha do tempo correspondente.

Uma característica importante é que os dois *streams* de vídeo são mixados em um único canal para dar a ilusão aos usuários de que o vídeo foi editado manualmente. Com isso, consegue-se uma maior atenção dos usuários em relação ao vídeo da apresentação. O *eMeet/InCA* também realiza a mixagem dos *streams* de áudio/vídeo dos participantes em um único canal.

O Cornell Lecture Browser foi proposto a resolver dois problemas clássicos dentro dos inúmeros sistemas que possuem tais características, assim como o *eMeet*: a sincronização dos dados e a edição automática.

A sincronização das mídias do sistema ocorre em pares e, por isso, deve-se tomar cuidado para não propagar erros indesejáveis. Por exemplo, se o erro de sincronização entre A e B é E1

unidades de tempo e entre B e C é E2 unidades de tempo, com certeza temos um erro de sincronização entre A e C de (E1+E2) unidades de tempo.

Figura 9: Hiperdocumento de apresentação do Lecture Browser (Mukhopadhyay & Smith, 1999)

As mídias capturadas, em relação à sincronização, podem ser classificadas como temporais (T) ou não temporais (U). As temporais contêm informações sobre tempo, como um vídeo, por exemplo. Já as não temporais, como os *slides*, não contêm informações sobre o tempo. Sendo assim, a sincronização de um par de mídias pode ser de três tipos, de acordo com a Figura 10.

	Temporais (T)	Não Temporais (U)
Temporais (T)	TTS	TUS
Não Temporais (U)	TUS	UUS

Figura 10: Tipos de sincronização multimídia (Mukhopadhyay & Smith, 1999).

- Sincronização Temporal - Temporal (TTS): ocorre quando as mídias a serem sincronizadas são temporais. Por exemplo, realizar a sincronização entre dois *streams* de vídeo ou de áudio.
- Sincronização Temporal - Não Temporal (TUS): ocorre quando uma das mídias é do tipo temporal (T) e a outra é do tipo não temporal (U). Por exemplo, realizar a sincronização entre um vídeo e *slides*.

- Sincronização Não Temporal - Não Temporal (UUS): ocorre quando ambas as mídias são do tipo não temporal (U). Por exemplo, realizar a sincronização entre os *slides* e os seus títulos, respectivamente.

A edição automática dos hiperdocumentos a partir das mídias capturadas inicia-se justamente com o processo de sincronização. No caso do Lecture Browser, primeiro há a sincronização dos dois *streams* de vídeo (TTS), depois há a sincronização com os *slides* (e seus títulos) a partir de um vídeo (TUS) e por fim, há a combinação dos dois vídeos para gerar a apresentação final.

A sincronização dos dois vídeos é feita através de uma placa de som cuja saída é conectada ao canal esquerdo de áudio de cada câmera. A placa de som emite um sinal que é responsável por iniciar a sincronização dos *streams* de forma rápida e eficiente. O áudio ambiente é gravado no canal direito de áudio de cada câmera. Tal representação pode ser observada na Figura 11.

Figura 11: Esquema utilizado no Cornell Lecture Browser para realizar a sincronização de dois vídeos (Mukhopadhyay & Smith, 1999).

Já para sincronizar os *slides* com o vídeo, utiliza-se o *stream* de vídeo que é responsável por filmar a região visual dos *slides*. A partir da imagem desses *slides*, há um tratamento em cima dessa imagem e um processamento é realizado para comparar a imagem dos *slides* filmados aos *slides* reais. Com isso, há um acerto de aproximadamente 97% da sincronização (Mukhopadhyay & Smith, 1999).

Por fim, para realizar a combinação dos vídeos em um único *stream*, foi utilizada uma técnica que depende da troca de *slides* e da definição prévia de algumas regras de edição. No caso do Lecture Browser, as regras utilizadas foram:

1. Três segundos antes da troca de *slides*, o vídeo mostrado será o que focaliza os *slides*. Tal vídeo será mostrado até cinco segundos depois da troca;

2. Nenhum segmento de vídeo poderá ter menos que três segundos;
3. Nenhum segmento de vídeo poderá ter mais de vinte e cinco segundos.

Com essas regras definidas, foi construído um algoritmo capaz de mixar os dois *streams* de vídeo em um único canal. Tal algoritmo consiste em quatro passos, que devem ser executados nessa ordem:

1. Atribuir ao vídeo final os segmentos do *stream* de vídeo que filma a região dos *slides*, de acordo com a primeira regra;
2. Atribuir aos outros espaços os segmentos do segundo *stream* de vídeo (o que segue e focaliza o palestrante o tempo todo);
3. Juntar segmentos que sejam menor do que a segunda regra;
4. Quebrar segmentos que sejam maior do que a terceira regra.

É importante observar que o vídeo gerado é armazenado diretamente em formato RealTM. De acordo com a classificação por quadrantes, o Cornell Lecture Browser se encaixa no quarto quadrante por se tratar de um sistema que exibe os dados de uma apresentação capturada, e não há indícios de que grupos locais se reúnem para acessar tal informação sincronamente. Esse sistema também se encaixa no primeiro quadrante, pois de acordo com o tempo de apresentação de cada *slide*, a apresentação das informações capturadas é diferente. Portanto, o sistema está presente durante a fase de captura síncrona e local.

O Lecture Browser possui formas de mixagem de áudio/vídeo e sincronização muito semelhantes ao *eMeet*. O *eMeet/InCA* faz a mixagem das diversas *streams* de áudio/vídeo para uma única *stream*. Essa única *stream* é composta por trechos de todas as outras *streams*, que são escolhidos de acordo com algoritmos que levam em consideração o volume do áudio de cada *stream* (Bila, 2002).

A conversão de *streams* de vídeo em um único canal de forma automática é uma característica importante para uma melhor visualização de reuniões (Mukhopadhyay & Smith, 1999). Portanto, essa técnica também foi aplicada ao *eMeet*.

Já o tratamento de sincronização realizado entre os elementos (vídeo, *slides* e linha do tempo) nos hiperdocumentos gerados pelo *eMeet/SERVE* é discutido no Capítulo 6.

4.2.2. Notelook

Um outro sistema que necessita de um ambiente estruturado, semelhante ao *eClass* e ao Lecture Browser, é o Notelook (Chiu et al., 1999). O Notelook é um sistema integrado a uma sala de conferências equipada com câmeras de vídeo controladas por computador e um projetor de vídeo. As câmeras de vídeo viabilizam a captura da imagem do apresentador e da tomada da sala em geral. O projetor, além de projetar uma apresentação com *slides*, por exemplo, também pode apresentar as imagens geradas pelas câmeras.

Uma das características mais importantes da infra-estrutura do Notelook é que existem computadores portáteis nos quais participantes das reuniões realizam suas anotações. Tais computadores portáteis, através de um programa cliente, recebem todas as informações da apresentação e das imagens das câmeras via transmissão sem fio (*wireless*). Com isso, os computadores pessoais podem circular livremente pela sala de reuniões entre os participantes sem que haja o transtorno de fios. Os clientes recebem os vídeos gerados pelas câmeras, mas não os armazenam devido ao problema de espaço necessário. Para realizar tal tarefa, existem servidores externos à sala de reuniões que recebem todas as informações geradas e realizam o processamento. Todos esses servidores possuem a característica de transmissão sem fio e rodam na plataforma Windows NTTM. Já os clientes, nos computadores portáteis, utilizam a plataforma Windows 95TM. A idéia de utilização de computadores portáteis para a realização de anotações e de transmissão *wireless* é muito boa, porém não foi utilizada na primeira versão do *eMeet*, pois o compartilhamento de anotações no *eMeet/InCA* é feito através de uma lousa eletrônica implementada no próprio software (Bila, 2002).

Observa-se que a arquitetura do Notelook, similarmente ao *eClass*, se preocupa em deixar a maior parte do processamento fora da sala de reuniões, fazendo com que o sistema cliente, que captura as anotações, seja o mais “leve” possível. Essa idéia também é utilizada pelo *eMeet*, uma vez que podem existir mais de uma sala de reuniões, que estarão distribuídas fisicamente. Um dos fatores importantes no processamento é que, para cada canal de vídeo gerado, existe um servidor próprio para realizar o processamento desse canal.

O áudio é capturado através de seis microfones distribuídos na sala, que são combinados para a geração de um único canal de áudio, que é armazenado juntamente com os *streams* de vídeo.

O sistema publica todas as anotações geradas durante a reunião na Web no formato de hiperdocumentos HTML com ligações para a parte de vídeo relacionada. Cada hiperdocumento HTML gerado pelo sistema consiste de três imagens do cliente do Notelook e todas imagens são mapeadas utilizando recursos da linguagem HTML. Tais imagens contêm todas as anotações realizadas no período correspondente da reunião, de acordo com o que foi realizado pelos usuários que fizeram as anotações. Quando um usuário seleciona uma dessas imagens, a parte dos vídeos referentes a essa imagem é recuperada e exibida. Observa-se que a única interação do usuário que está acessando tais informações na Web, é escolher qual imagem que ele gostaria de recuperar. A partir daí, o usuário pode assistir aos vídeos relacionados à imagem escolhida. Portanto, em comparação com o *eClass*, com o Cornell Lecture Browser e com o *eMeet*, a apresentação do Notelook na Web é bastante limitada. É importante observar que todo o vídeo gravado é armazenado e recuperado em uma base de dados central própria para vídeos.

O sistema suporta reuniões locais e ainda não há indícios de que há trabalhos em busca de suporte a reuniões distribuídas. Devido a esses fatores, o Notelook pertence ao primeiro, terceiro e quarto quadrantes dentro à classificação proposta por Ellis et al. (1991). O terceiro quadrante foi considerado nesse sistema por haver indícios de que os participantes da reunião se reúnem para a visualização conjunta das informações capturadas.

4.2.3. TELEP

Em grandes empresas, como a Microsoft, a transmissão de uma apresentação em intranets¹⁴ é cada vez mais comum, quer seja para treinamento dos funcionários, conteúdo informativo, ou mesmo para entretenimento. Seja qual for o tipo de apresentação, curso ou palestra, informações de áudio, vídeo e *slides* podem ser transmitidos simultaneamente para os usuários, desde que a rede já possua banda suficiente para tal e que os computadores pessoais suportem áudio e vídeo em tempo real. Independentemente desses fatores, o importante é que a tecnologia existe e pode ser utilizada no conceito de reuniões distribuídas, como no *eMeet*.

Genericamente, a transmissão de apresentações traz algumas vantagens para os expectadores:

- Eles não precisam se deslocar do ambiente de trabalho. Muitos podem estar trabalhando em outras cidades ou países;

¹⁴ Intranets são sistemas Web restritos a determinados usuários que normalmente fazem parte de empresas, universidades, órgãos públicos e outros estabelecimentos.

- Se eles não estiverem gostando da apresentação, eles podem sair sem perder tempo e sem correr o risco de ofender o palestrante;
- Se parte da apresentação não for interessante, ele pode realizar outra atividade e gerenciar outras tarefas simultaneamente.

Porém, a transmissão também traz desvantagens:

- Com a possibilidade de transmissão remota, o auditório ao vivo poderá ter poucas pessoas e isso poderá desmotivar ou ofender o palestrante;
- Os expectadores remotos não têm a possibilidade de sentir o ambiente de apresentação, interagir com a platéia local ou fazer uma pergunta diretamente ao palestrante;
- A platéia ao vivo pode fazer perguntas e conduzir o foco da apresentação para seus interesses, desmotivando os expectadores remotos.

No caso de reuniões, muitos fatores são semelhantes aos de apresentações e por isso, tais fatores estão sendo discutidos neste trabalho.

O sistema TELEP (Jancke et al., 2000) foi desenvolvido para maximizar os benefícios da transmissão remota e minimizar suas desvantagens. Atualmente, o TELEP está sendo utilizado para a transmissão de seminários e apresentações dentro da Microsoft.

A idéia do sistema é fazer com que os expectadores remotos possam ter maior participação durante a apresentação, além de fazer com que o palestrante sinta uma maior presença do público remoto. Tudo isso tem que ser feito sem que o palestrante se distraia ou se sinta incomodado pela presença da tecnologia do sistema no auditório, ou seja, deve ser o mais ubíquo e natural possível.

No caso dos expectadores remotos, o sistema disponibiliza uma série de funcionalidades interessantes, como:

- O expectador remoto, além de ver o vídeo, os *slides* e o áudio de forma síncrona, pode realizar perguntas diretamente para o palestrante. Isso é fundamental para que o expectador remoto se sinta motivado a assistir à apresentação;
- Os expectadores remotos podem trocar comentários particulares ou públicos através de inúmeras salas de conversação¹⁵ disponíveis pelo sistema;

¹⁵ Salas de conversação ou de “bate-papo” são conhecidas também por *Chat rooms*.

- Há uma fila das perguntas que já foram propostas pelo público remoto para evitar que perguntas repetitivas sejam feitas, atrapalhando o apresentador e o andamento da apresentação;
- Há a possibilidade de transmissão em tempo real do vídeo do expectador remoto para outros expectadores remotos, a fim de haver uma maior interação entre eles. Para isso, o expectador remoto precisa ter uma câmera em seu computador pessoal;
- Há a possibilidade do público remoto enviar sua foto, escolher uma imagem ou mandar a transmissão em tempo real do vídeo que o represente junto ao palestrante e à platéia ao vivo;
- Há a possibilidade de participação em anonimato, no caso de usuários que preferem não se identificar;
- Há a possibilidade de votação remota, onde o público remoto pode se manifestar contra ou a favor de enquetes elaboradas em tempo real pelo palestrante;
- Os expectadores remotos podem observar a platéia local através de câmeras instaladas no auditório.

Para o palestrante perceber a presença dos expectadores remotos, inúmeras funcionalidades foram previstas no sistema:

- Há um telão que projeta a foto, imagem ou vídeo de grande parte do público remoto. Com isso, o palestrante pode perceber a presença do público remoto. Nesse mesmo telão, pode haver a indicação do número de expectadores remotos que estão assistindo à apresentação;
- O palestrante pode receber uma pergunta da platéia remota quando quiser, e essa pergunta é exibida no telão. A cada nova pergunta é emitido um som para avisar ao palestrante que uma pergunta está sendo feita;
- O palestrante pode contar com votos do público remoto em caso de uma votação ou enquete.

Muitas dessas características aplicadas a apresentações também são desejáveis em uma reunião. A formação de enquetes e sistemas de votação, a visualização dos participantes da reunião, a possibilidade de levantamento de perguntas por qualquer um dos participantes e a troca de

informações em tempo real, através de salas de conversação, são exemplos de funcionalidades que já existem ou podem ser implementadas futuramente no *eMeet*.

Na realidade, a arquitetura do sistema TELEP é dividida em dois sistemas paralelos. Um é responsável por enviar as mídias (áudio, vídeo e *slides*) para os expectadores remotos, e outro é responsável pela interação entre o público remoto e o palestrante.

O codificador de vídeo foi desenvolvido para consumir o menor ciclo de processamento possível. As transmissões das mídias são baseadas no Microsoft Windows Media ServerTM, com a utilização de transmissão *multicasting*¹⁶ para transportar os dados de forma eficiente.

O decodificador de vídeo recebe os *frames* de vídeo *multicasting* e realiza a descompressão, exibindo-os em tempo real. O processamento é feito de forma a ser o mais leve possível, para que o servidor possa suportar cerca de trinta ou quarenta vídeos sem saturar o processador. Caso haja uma lentidão no processamento dos *frames* de vídeo, o *frame rate*¹⁷ é alterado.

Figura 12: Interface que é exibida para os expectadores remotos no sistema TELEP¹⁸.

A interface da apresentação para o público remoto é dividida praticamente em duas partes, como pode ser observado na Figura 12. A maior delas, à direita, exibe o vídeo e os *slides* que estão sendo apresentados pelo palestrante. O *slide* muda automaticamente quando o palestrante faz uma transição. A parte da esquerda exibe a representação de todos os outros usuários remotos que estão assistindo à palestra, além de exibir a fila de perguntas que já foram escritas e serão

¹⁶ A transmissão *multicasting* evita que dados, durante a transmissão pela rede, sejam duplicados desnecessariamente.

¹⁷ O *frame rate* é a taxa de *frames* por unidade de tempo (normalmente em segundos) em que o vídeo é exibido.

¹⁸ A Figura 12 encontra-se no endereço http://research.microsoft.com/coet/telep_page.asp.

enviadas para o palestrante. Há opções de abrir uma sala de conversação com um ou mais expectadores remotos.

O telão que exhibe a representação (imagem, vídeo ou foto) dos expectadores remotos ao palestrante é dividido de forma a mostrar o maior número de expectadores possível. Quando um expectador remoto faz uma pergunta, ele muda de cor automaticamente, e a pergunta aparece no telão em uma nova janela.

Testes realizados para medir a satisfação dos usuários mostraram que, tanto os expectadores locais e remotos quanto os palestrantes, demonstraram interesse pelo sistema.

Por não haver a disponibilização desse sistema para futuro acesso na Web, esse sistema pertence ao primeiro e segundo quadrantes, dentro da classificação proposta.

4.2.4. MSTe

Cada vez mais freqüentes, apresentações ao vivo têm foco na platéia local e, ao mesmo tempo, são transmitidas para expectadores remotos e ainda gravadas para os usuários acessarem a apresentação pela Web em um momento posterior.

Com as melhorias da largura de banda de rede, processamento e tecnologia de compressão, a rotina de uso do vídeo sob-demanda está cada vez mais comum no trabalho, em casa e nos sistemas de educação. Entretenimento e educação são os dois principais usos do vídeo sob-demanda (He et al., 2000).

Diferentemente do sistema TELEP, o MSTe (Microsoft Technical Education) (He et al., 2000) é mais voltado para a produção sob-demanda de cursos e posterior acesso desse conteúdo pela Web.

O MSTe é um sistema que fornece treinamento técnico para funcionários corporativos. O sistema possui uma série de cursos de desenvolvimento de software, métodos de testes e gerenciamento de programas, além de uma série de seminários sobre a empresa. Tais cursos podem ser acessados pela Web a qualquer momento, não necessitando que os usuários estejam sincronamente conectados para ver uma apresentação ao vivo, como no caso do TELEP.

O MSTe possui uma interface Web para o acesso ao material, como pode ser vista na Figura 13. Os funcionários podem localizar um determinado assunto usando tabelas de índices ou a busca por palavras-chave.

Figura 13: Interface Web do MSTE (He et al., 2000).

A interface possui três *frames*: um é responsável pela visualização do vídeo, outro pela tabela de índices e o principal para a exibição dos *slides*. Os *slides* são sincronizados com o vídeo, ou seja, quando o apresentador vai para um novo *slide*, a transição para o próximo *slide* é feita de forma automática. Quando um usuário seleciona um índice, a apresentação salta para o ponto correspondente, tanto no vídeo quanto nos próprios *slides*.

A produção do vídeo com a tabela de índices e a sincronização dos *slides* requer um grande esforço da equipe que está preparando o material, pois toda a sincronização é feita de forma manual. Porém, o material fica muito mais interessante e completo para os usuários que irão acessar o material posteriormente pela Web. Todo esse trabalho de montagem e sincronização é justificado pelo número de funcionários que acessam tal material, segundo medições realizadas pela própria Microsoft.

Ainda nessas mesmas medições, alguns pontos interessantes podem ser destacados sobre o comportamento dos usuários que acessam as apresentações pela Web (He et al., 2000):

- O tempo de permanência de um usuário em uma apresentação pode ser verificado logo no início. Estudos mostram que mais da metade dos usuários deixam a apresentação em menos de quatro minutos. Porém, a maioria dos que continuam assistindo chegam até o final;

- No início da apresentação de cada *slide* há um maior número de usuários do que no seu decorrer. Isso se deve ao fato de que os usuários passam rapidamente pelos *slides* para verificar se são de seus interesses.

Observando esse comportamento dos usuários, He et al. (2000) concluem que os resultados encontrados por Morkes & Nielsen (1997) e Nielsen (1995) aplicados à leitura de texto na Web, onde eles dizem que a navegação é mais uma “olhada superficial” do que leituras completas em si, se aplicam também a vídeos e apresentações. Além disso, Morkes & Nielsen (1997) e Nielsen (1995) recomendam que cada parágrafo escrito na Web seja limitado a uma idéia. Similarmente, os apresentadores devem limitar cada *slide* em um ponto chave (He et al., 2000).

Outro fator interessante dos materiais e apresentações sob-demanda é a possibilidade de haver outras referências semelhantes como uma URL, por exemplo. Assim, os usuários que desejarem complementar um ponto em questão, podem instantaneamente buscar outras informações.

Os palestrantes, participantes das reuniões e criadores de cursos, podem despendar tempo extra na produção do material sob-demanda pois, na maioria das vezes, o público que utilizará tal material poderá ser muito maior do que a platéia de uma apresentação local. Além disso, há a possibilidade de automatização de grande parte desse processo, que é uma das características do *eMeet*. Esse é um fator motivante para o crescimento desse tipo de material na Web.

A interação e cooperação sobre material digitalizado podem ser facilitadas por tecnologias que permitam com que as pessoas criem e compartilhem anotações que estarão ligadas a trechos da apresentação. Sendo assim, há um maior papel cooperativo entre os expectadores sob-demanda.

O MSTe se encaixa no primeiro e quarto quadrantes da classificação adotada, pois ele é focado na captura local e síncrona e posterior apresentação do conteúdo capturado na Web.

4.2.5. TeamSpace

O TeamSpace (Richter et al., 2001) é um ambiente que tem como objetivos principais suportar o trabalho cooperativo e gerenciar os processos de compartilhamento de informação, comunicação e coordenação entre grupos de trabalho. O TeamSpace pode atuar em atividades síncronas e assíncronas dos grupos de trabalho. Assim, o TeamSpace inclui entre suas funcionalidades o

suporte ao gerenciamento de reuniões distribuídas, que inclui a preparação de novas reuniões, a captura dos dados da reunião e o posterior acesso a esses dados.

Durante o processo de preparação de novas reuniões, usuários podem reservar salas de reuniões disponíveis e incluir usuários, que foram previamente cadastrados no sistema, para participarem da reunião. Automaticamente, o TeamSpace envia *e-mails* a todos os participantes cadastrados, indicando o local e a data da realização da reunião. Além disso, é possível inserir uma sequência de *slides*, preparada anteriormente, que será exibida durante a reunião. O processo de preparação é realizado pela interface principal do TeamSpace, que é uma interface Web.

A captura dos dados da reunião é feita através de um programa cliente, que é iniciado a partir da interface Web do TeamSpace. O programa cliente é executado durante a reunião pelos participantes, que podem estar distribuídos fisicamente. Os participantes podem visualizar, editar e realizar suas próprias anotações, bem como visualizar e anotar sobre os *slides* apresentados, que foram inseridos durante a fase de preparação da reunião. Esses eventos são capturados juntamente com o áudio e vídeo de cada participante da reunião, se os mesmos estiverem utilizando câmeras e microfones. Assim como no *eMeet*, os participantes podem visualizar uns aos outros durante a reunião, através dos vídeos capturados, transmitidos e apresentados em tempo real.

A visualização das informações capturadas pode ser feita através de hiperdocumentos, que são gerados automaticamente após as reuniões, como no *eMeet*. As informações capturadas das reuniões (apresentações de *slides*, anotações, itens de discussão e áudio/vídeo) são apresentadas por um *applet* na forma de painéis separados, como um fichário, ou em janelas independentes, conforme ilustrado na Figura 14. Durante a visualização das informações capturadas, os usuários podem escolher quais eventos desejam visualizar e realizar buscas por determinadas partes da reunião.

O TeamSpace possui muitas características em comum com o *eMeet*, como a captura e apresentação síncrona de mídias contínuas, como áudio e vídeo, durante as reuniões (que podem ser distribuídas) e a visualização de elementos, como linha do tempo, *slides* e áudio/vídeo capturados, de forma sincronizada, nos hiperdocumentos gerados após as reuniões.

A diferença principal em termos de implementação na visualização das informações capturadas é que o TeamSpace utiliza *applets* para controlar todos os elementos a serem visualizados, enquanto que o *eMeet/SERVE*, utiliza *applets* somente para a reprodução dos áudios/vídeos capturados. Os demais elementos no *eMeet/SERVE*, como linha do tempo e *slides*,

são apresentados e sincronizados através de funções desenvolvidas na linguagem JavaScript e HTML (mais detalhes encontram-se na Seção 6.4).

Figura 14: Apresentação dos dados de uma reunião no TeamSpace. Na janela maior, é mostrado o *applet* de visualização contendo as linhas do tempo (no alto) e os painéis segmentados com o conteúdo do hiperdocumento gerado (ao centro). Tais painéis também podem ser apresentados em janelas independentes, mostradas à esquerda¹⁹.

Em termos de funcionalidades, o TeamSpace possui algumas funcionalidades a mais do que o *eMeet*, tais como agendamento e reserva de salas de reuniões, envio de *e-mails* aos participantes e mecanismos de busca por trechos de reuniões mais completos.

Por possuir suporte síncrono e assíncrono em reuniões distribuídas, o TeamSpace pertence aos quatro quadrantes da classificação proposta.

4.3. Técnicas de apresentação e sumarização

4.3.1. Apresentação de vídeo

Além das técnicas de recuperação e apresentação dos dados contidos nos sistemas estudados, outras técnicas discutidas na literatura têm relação com o *eMeet*. Os dados apresentados para os usuários precisam conter mecanismos que facilitem, ao máximo, a busca da informação desejada. Fazer essa busca em mídias contínuas e temporais, tais como o vídeo ou o áudio, traz muitos benefícios para os usuários, pois eles podem despendar menos tempo na procura de determinados trechos dessas mídias (Li et al., 2000).

¹⁹ A Figura 14 encontra-se no endereço <http://www.research.ibm.com/teamspace/projectdescription.html>.

Dois aspectos são fundamentais para os usuários quando eles precisam extrair informações de uma mídia contínua, como o vídeo. São elas: a possibilidade e habilidade de gastar menos tempo do que em uma apresentação ao vivo, e o sentimento de estar sob-controle para achar e ver todo o conteúdo que ele deseja (Li et al., 2000).

Um dos primeiros meios para criação de conteúdo e distribuição foi o vídeo. Porém, o modo como pessoas assistem a vídeos não mudou significativamente desde a invenção do videocassete (VCR) por volta dos anos 70 e 80. O VCR permite a possibilidade de se assistir a vídeo sob-demanda com as funções de *pause*, *fast-forward* e *rewind*²⁰.

A procura por uma determinada parte do vídeo no VCR é possível, porém leva-se um determinado tempo por causa do modo de acesso linear imposto pela fita magnética. Já no vídeo digital utilizado atualmente pela Internet, a procura pode ser feita instantaneamente. Além disso, o *replay* instantâneo passou a ser possível também. Outra grande vantagem é a possibilidade de aumentar a velocidade de reprodução até um certo limite, sem que a legibilidade e o entendimento sejam perdidos, permitindo com que as pessoas gastem menos tempo para assistir ao vídeo.

Apesar dos grandes interesses comerciais sobre o vídeo, o foco principal das ferramentas líderes de mercado²¹ tem sido a criação de conteúdo e não a visualização ou facilidade de navegação por essas mídias (Li et al., 2000).

A ferramenta²² criada por Li et al. (2000) para introduzir novos conceitos na navegação de vídeos foi baseada estendendo o Windows Media PlayerTM e possui dois modos de exibição, listados a seguir, que o usuário pode optar:

- Controle básico de navegação: corresponde às funções básicas de qualquer *player*²³ de vídeo. Elas incluem *play*, *pause*, *fast-forward*, procura, “ir para o início” e “ir para o fim” do vídeo. Durante o *fast-forward*, o áudio não é tocado e a procura é feita através de uma linha de tempo na própria interface. Não há a tradicional opção de *rewind*, devido às limitações do Windows Media PlayerTM (Li et al., 2000).

²⁰ Essas funções são tradicionais de um VCR e significam, respectivamente, pausar, avançar rapidamente e voltar em uma determinada parte de um vídeo.

²¹ Li. et al. (2000) citam que o Real PlayerTM, o QuickTimeTM e o Windows Media PlayerTM são as ferramentas líderes do mercado de *players* de vídeo.

²² O nome da ferramenta criada não foi citado em nenhum momento.

²³ *Player* pode ser traduzido como tocador de vídeo, porém o nome *player* é mais utilizado na área.

- Controle avançado de navegação: além das funções contidas no controle básico de navegação, nesse modo de exibição estão incluídas as funcionalidades de controle de velocidade, índice textual, índice visual e controles de deslocamento, descritas a seguir:
 - Controles de Velocidade
 - Compressão de Tempo (TC): utiliza técnicas de processamento que aumentam a velocidade do vídeo, sem perder a sintonia do áudio.
 - Remoção de Pausas (PR): detecta pausas e silêncio na fala contínua e remove o segmento do vídeo correspondente.
 - Índice textual
 - Tabela de conteúdo (TOC): é uma lista dos conteúdos pré-existentes. Ao escolher um índice, o vídeo vai diretamente para a parte correspondente.
 - Anotações: usuários podem realizar as suas próprias anotações relacionadas a uma posição do vídeo. Outros usuários poderão acessá-las através do índice de anotações.
 - Índice visual
 - “Shot Boundary Frames” (SB): são *snapshots*²⁴ de segmento dos vídeos, onde os usuários podem ver o que o vídeo mostra em um determinado momento. Quando o usuário seleciona um SB, é mostrada exatamente aquela parte do vídeo.
 - Marcadores de linha do tempo: de acordo com a parte do vídeo que está sendo exibida, cores diferentes nos índices textuais irão aparecer para indicar em que parte o vídeo se encontra. Tal indicação também acontece nos *snapshots* dos SB.
 - Controles de deslocamento
 - Retroceder: os usuários poderão voltar o vídeo segundo um intervalo de tempo definido (5 ou 10 segundos), um índice no TOC, uma anotação ou um *snapshot* do SB com apenas uma operação de seleção.

- Avançar: semelhante ao retroceder, porém o vídeo vai para frente ao invés de voltar.

A interface utilizada exibe todas essas funcionalidades, como apresentada na Figura 15.

Figura 15: Interface da ferramenta criada por Li et al. (2000) com controles avançados.

O objetivo principal de Li et al. (2000) é fazer com que os usuários gastem menos tempo para ver um vídeo digital e tenham controle sobre o que estão procurando. Para medir se as funcionalidades da ferramenta no modo de controle avançado de navegação são realmente úteis, foi realizada uma pesquisa na qual participantes tiveram 30 minutos para navegar por um vídeo que poderia durar de 45 minutos a 1 hora. Existiam seis tipos de vídeo, e ao final dos testes, cada participante respondia a um questionário que perguntava, dentre outras coisas, qual a estratégia adotada para percorrer o vídeo. Os seis tipos de vídeo propostos foram: aulas, conferências, esportes, shows, notícias e viagens. Em cada um desses tipos, estratégias claras de como percorrer o vídeo foram adotadas pelos participantes.

Depois de analisar os questionários, Li et al. (2000) concluíram que os vídeos podem ser classificados basicamente em três tipos:

- Vídeos com informação centralizada na mídia áudio: vídeos como aulas e conferências contêm a maioria de seus conteúdos no canal de áudio e, usualmente,

²⁴ *Snapshots* são imagens tiradas a partir da reprodução um vídeo em um determinado momento.

possuem uma pequena atividade visual. Para esses casos, tabelas de conteúdos e índices de anotações são os mais importantes. É nessa categoria que se enquadra o *eClass* e o *eMeet*. Portanto, a disponibilização de tabelas de conteúdos e índices de anotação são aspectos fundamentais para o acesso aos dados.

- Vídeos com informação centralizada na mídia vídeo: vídeos como esportes e viagens contêm a maioria de seus conteúdos no canal de vídeo. Para esses casos, praticamente todos os tipos de índices e funcionalidades são úteis, pois os usuários podem fazer a combinação dessas funcionalidades para procurar o que desejam.
- Vídeos com narração ou entretenimento: vídeos como novelas ou shows são afetados quando os usuários tentam minimizar o tempo gasto. Nesse caso, a maioria das tentativas de minimizar o tempo gasto pode acarretar perda de conteúdo. Ratificando essa conclusão, um dos participantes disse: “Eu raramente vou querer ver um show com a velocidade aumentada”.

Tanto o *eClass* como o *eMeet* possuem semelhanças com uma conferência e, portanto, os conteúdos desses sistemas estão centralizados no áudio. De fato, a abordagem de gravar áudio de alta qualidade e vídeo de baixa qualidade foi adotada no *eClass* (Abowd, 1999) e no *eMeet* (Bila, 2002) por esse mesmo motivo. A versão atual do *eMeet* não apresenta tabelas de conteúdos e índices para buscas e anotações. Esses são alguns dos trabalhos futuros a serem realizados.

4.3.2. Sumarização de apresentações

Uma outra técnica que pode ser utilizada para aumentar a velocidade de visualização de mídias como áudio, vídeo e *slides* em apresentações é a sumarização. Uma sumarização deve corresponder a uma versão curta das apresentações, contendo as partes mais importantes da mesma. O desafio é conseguir gerar resumos²⁵ de forma automática, isto é, sem que haja a necessidade de uma pessoa selecionar e editar manualmente o material.

Segundo He et al. (1999), para que o resumo de uma apresentação seja realmente eficaz, ele deve apresentar as seguintes características:

²⁵ A sumarização também é citada neste texto como “resumo”.

- Concisão: qualquer segmento que é selecionado para entrar no resumo deve conter somente a informação necessária. Além disso, esse segmento deve ser o mais curto possível.
- Cobertura: as partes escolhidas para o resumo devem cobrir todos os pontos-chaves da apresentação.
- Contexto: a sequência de apresentação dos segmentos escolhidos deve manter o contexto da apresentação.
- Coerência: a passagem entre os segmentos deve ser o mais natural possível.

He et al. (1999) consideram que quatro principais aspectos podem ser utilizados para a produção de resumos automáticos. São eles:

- Canal de vídeo: a movimentação e os gestos dos palestrantes podem ser tratados para indicar importância em certos pontos da apresentação.
- Canal de áudio: características tais como o *pitch*²⁶, pausa e intensidade dada à fala podem estar associados aos aspectos mais importantes da apresentação.
- Ações do palestrante: o tempo gasto em um *slide* e a própria passagem de um *slide* para o outro representam uma informação importante na escolha dos segmentos que estarão no resumo.
- Ações dos usuários finais (que assistiram à apresentação): usando estatísticas de acesso de usuários que já assistiram à apresentação, pode-se saber quais foram as partes mais e menos acessadas. Com isso, pode-se montar um resumo adequado às características daqueles usuários.

Dentro dos três últimos aspectos²⁷, He et al. (1999) construíram três algoritmos que fazem o resumo automático de uma apresentação.

O primeiro algoritmo levou em conta somente a transição entre *slides*. Portanto, um segmento de duração apropriada logo após uma mudança de *slides* iria para o resumo. Tal técnica considera que as informações mais importantes estão logo no início de cada *slide*.

²⁶ A variação do *pitch* nada mais é do que a variação de tons (notas) musicais, que pode ser automaticamente detectada.

²⁷ O canal de vídeo foi ignorado para a produção de resumos automáticos no trabalho de He et al. (1999), pois os vídeos apresentavam uma baixa resolução.

O segundo algoritmo levou em conta somente o *pitch* do canal de áudio, de acordo com um algoritmo proposto por Arons (1994²⁸) *apud* (He et al., 1999). Arons utilizava segmentos de aproximadamente oito segundos para compor seus resumos. Porém, o segundo algoritmo utilizou segmentos de cerca de quinze segundos para não haver muitas mudanças entre um segmento e outro, e tentar minimizar a perda de coerência do vídeo resumido.

Já o terceiro algoritmo levou em conta o *pitch* do canal áudio, a transição entre *slides* e as informações de acesso dos usuários que já assistiram ao vídeo.

Depois de um teste realizado com cerca de vinte e quatro pessoas de mesmo nível de conhecimento, constatou-se que os três algoritmos propostos tiveram desempenhos semelhantes. Porém, o resumo realizado por um autor de forma manual, que também entrou nos testes, foi o que obteve uma maior aprovação. Apesar disso, constatou-se que os usuários, utilizando um resumo automático, atingiram cerca de 60% do nível de conhecimento da apresentação de usuários que utilizaram um resumo editado pelo autor de forma manual (He et al., 1999).

O resumo automático das reuniões pode ser implementado futuramente no *eMeet* para as pessoas que perderam toda ou parte da reunião e precisam revê-la de forma rápida. Essa funcionalidade poderá ser incluída no *eMeet* em trabalhos futuros, por se tratar de uma funcionalidade que envolve pesquisas específicas para esse assunto.

4.4. Fóruns de discussão

Além de mecanismos que os hiperdocumentos Web podem oferecer para que os usuários busquem as informações necessárias, é desejável também que esses hiperdocumentos sejam acessados em um primeiro momento e que eles continuem sendo acessados, constituindo “comunidades virtuais”. Tais comunidades têm o objetivo de debater um assunto comum e resolver as dúvidas mais frequentes sobre aquele assunto, estabelecendo um canal de comunicação entre os participantes.

Nesse contexto, uma das mais óbvias ferramentas de comunicação é o fórum de discussão. Existem inúmeros *sites* na WWW hoje que possuem tal ferramenta. Entretanto, é interessante notar que (a) a maioria é utilizada sem muita frequência, (b) o fórum é utilizado juntamente com

²⁸ (Arons, 1994) Arons, B. (1994). Pitch-based emphasis detection for segmenting speech recordings. *Proceedings of International Conference On Spoken Language Processing*, vol. 4, p. 1931-1934. 1994.

outras funções, como salas de conversação, e (c) os *newsgroups*²⁹, que são considerados como a “velha ferramenta de discussão” ainda estão sendo muito utilizados (Maurer et al., 1999).

Maurer et al. (1999) fazem uma análise e propõem parâmetros que devem ser utilizados para que um fórum de discussão na Web obtenha sucesso. Eles observaram que os *newsgroups*, apesar de antigos, ainda fazem mais sucesso que os fóruns de discussão, e que o sucesso dos *newsgroups* se deve por cinco principais razões: (a) todos os usuários estão concentrados no mesmo lugar, enquanto os fóruns estão distribuídos em centenas de *sites* WWW, (b) muitas funcionalidades dos *newsgroups* ainda não são encontradas nos fóruns (por exemplo, o aparecimento automático de novas mensagens enviadas para todos os usuários), (c) o acesso aos *newsgroups* são muito mais rápidos que o acesso aos fóruns, (d) os fóruns de discussão devem ser vistos em um contexto maior, como uma ferramenta de cooperação (*General Cooperation Tools* - GCT) e (e) muitos fenômenos psico-sociais são problemas quando uma nova ferramenta de comunicação surge, ou seja, há resistências naturais a mudanças por parte dos usuários.

Acredita-se que, devido à maior flexibilidade da WWW, os fóruns de discussão irão, eventualmente, substituir os *newsgroups*, principalmente se eles forem utilizados como ferramentas de cooperação (GCT).

Estudos realizados por Maurer et al. (1999) mostram que os fóruns de discussão devem conter parâmetros tecnológicos e não-tecnológicos para obter sucesso. Alguns parâmetros tecnológicos importantes são:

- Os fóruns devem ser rápidos e fáceis de usar;
- A opção de contribuição anônima deve existir;
- As contribuições mais recentes deverão estar facilmente visíveis para o usuário;
- Devem existir mecanismos de notificação de quando uma certa contribuição é dada;
- Os fóruns devem conter um mecanismo de busca por palavras-chave, autores e títulos.

Para cada funcionalidade que é acrescida ao sistema, existe o aumento da sobrecarga cognitiva (Conklin, 1987; Maurer et al., 1999) imposta aos usuários, principalmente pela interface do

²⁹ *Newsgroups* são grupos de discussão onde os participantes recebem as mensagens discutidas e enviam novas idéias em seu próprio *e-mail*.

sistema que, cada vez mais, ficará mais complexa. Para tentar contornar esse problema, três propostas podem ser utilizadas:

1. Manter a interface e a funcionalidade o mais simples possível;
2. Contemplar diversos modos de interface na ferramenta: iniciantes, intermediários e avançados;
3. Dar a opção de um menu para que o usuário escolha entre controles básicos e controles avançados.

No caso de parâmetros não-tecnológicos, o principal para o sucesso do fórum é que exista um líder ou um grupo de líderes. Isso não significa que o fórum passa a ser moderado, mas sim que uma ou mais pessoas devem se sentir responsáveis em manter o fórum atualizado e acessado pelos usuários. Não é importante que os usuários do fórum saibam da existência de um líder, pois o importante é que haja participação e cooperação.

São papéis do(s) líder(es) do fórum:

- Ler todas as contribuições e ter certeza de que todas as questões foram respondidas, mesmo quando a resposta for: “Essa é uma questão interessante. Eu também tenho interesse na resposta”;
- Ter certeza de que as questões foram respondidas de uma forma útil em um certo estágio. Os líderes podem e devem mandar mensagens para algumas pessoas que eles desconfiam que saibam a(s) resposta(s);
- A combinação de fórum e *e-mail* é essencial. Como no item anterior, os líderes devem utilizar seus *e-mails* para convidar participantes, sem que os usuários do fórum saibam que isso foi feito;
- O fórum deve criar um canal entre os participantes e o próprio fórum. Por exemplo, caso um participante fique muito tempo sem mandar mensagens, os líderes podem enviar-lhe uma mensagem tentando trazê-lo de volta à discussão;
- Um fórum deve ter seu foco voltado a um tema próprio. Um ou outro comentário pessoal pode ser feito, mas os líderes devem evitar que o objetivo do fórum seja alterado para troca de mensagens pessoais.

Como discutido anteriormente, o fórum deve ser visto como uma ferramenta de cooperação genérica (GCT). Para isso, é necessário dar ao usuário a possibilidade de gerenciar o seu próprio

fórum, permitindo que ele se sinta livre para criar um espaço de discussão de um tema próprio e dando permissão de escrita aos usuários que ele quiser. Além disso, deve haver espaço para troca de documentos, por exemplo. Assim, os fóruns atuais podem ser utilizados para um trabalho cooperativo que pode melhorar a comunicação entre os participantes.

Dentro do contexto do *eClass*, há a utilização de uma ferramenta de apoio, chamada *CoWeb* (Guzdial, 1999). A *CoWeb* é uma ferramenta muito semelhante aos fóruns de discussão e permite que os alunos de uma determinada disciplina, juntamente com o professor, troquem mensagens e informações, além de poderem tirar dúvidas.

A *CoWeb* já tem sua utilização consolidada juntamente com o *eClass* através de ligações criadas automaticamente entre os dois sistemas (Abowd et al., 1999). As ligações automáticas permitem que alunos e professores possam encontrar conteúdos relacionados entre as duas ferramentas, complementando a inserção de ligações de forma manual (Pimentel et al., 2001a). A *CoTeia* (Arruda Jr. & Pimentel, 2001) é uma ferramenta similar à *Coweb* e foi desenvolvida no ICMC/USP a partir dos conceitos propostos por Guzdial (1999).

Um sistema de fórum de discussão também é desejável para o *eMeet*, onde os participantes das reuniões podem formar uma comunidade para debater o assunto. Tal funcionalidade ainda não existe no *eMeet* e poderá ser realizada em trabalhos futuros. Uma sugestão para trabalho futuro seria a integração entre a *CoTeia* e o *eMeet/SERVE*.

4.5. Considerações finais

Muitos sistemas e técnicas de captura e apresentação foram explorados no desenvolvimento do *eMeet*, tanto na parte de captura (*eMeet/InCA*), quanto na parte de apresentação (*eMeet/SERVE*). Outros ainda podem ser explorados em trabalhos futuros. Por exemplo, a utilização de conceitos de usabilidade nas interfaces projetadas e as diversas possibilidades de navegação pelos dados capturados são características que foram consideradas pelo *eMeet/SERVE*. Além disso, a formação de comunidades para discussão do conteúdo capturado, através de fóruns de discussão, é um fator interessante que pode ser implementado em trabalhos futuros.

Os sistemas apresentados foram classificados de acordo com o *framework* proposto por Ellis et al.(1991). O resultado dessa classificação, que também inclui o *eMeet*, é apresentada na Tabela 1.

Tabela 1: A classificação dos sistemas apresentados dentro do *framework* proposto por Ellis et al. (1991).

1° Quadrante Cornell Lecture Browser, Notelook, TELEP, MSTE, TeamSpace, <i>eClass</i> , <i>eMeet (InCA)</i>	2° Quadrante TELEP, TeamSpace, <i>eMeet (InCA)</i>
3° Quadrante Notelook, TeamSpace, <i>eClass</i> , <i>eMeet (SERVE)</i>	4° Quadrante Cornell Lecture Browser, Notelook, MSTe, TeamSpace, <i>eClass</i> , <i>eMeet (SERVE)</i>

Como apresentado na Tabela 1, o *eMeet* possui características correspondentes aos quatro quadrantes dessa classificação. Observa-se que o *eMeet*, durante a reunião (forma síncrona - InCA), atende aos grupos locais (primeiro quadrante) e distribuídos (segundo quadrante). Depois da reunião, quando as informações estiverem disponíveis na Web (forma assíncrona - SERVE), o *eMeet* também atende aos grupos locais (terceiro quadrante) e distribuídos (quarto quadrante). Essa foi mais uma motivação para a realização deste trabalho, pois o *eMeet* atende a grupos locais e distribuídos, de forma síncrona e assíncrona.

Observa-se que o TeamSpace (Richter et al., 2001) também pertence aos quatro quadrantes da classificação realizada e possui características similares ao *eMeet*. Porém, o TeamSpace e o *eMeet* possuem diferenças em termos de implementação e de funcionalidades, como discutido na Seção 4.2.5.

Conforme reportados, os sistemas apresentados neste capítulo não utilizaram uma arquitetura Web para desenvolvimento. Além disso, as interfaces dos hiperdocumentos gerados por esses sistemas não foram desenvolvidas a partir de regras, heurísticas ou padrões de usabilidade discutidos na literatura. O *eMeet/SERVE* utilizou uma arquitetura proposta em seu desenvolvimento (mais detalhes no próximo capítulo) e as interface dos hiperdocumentos gerados foram desenvolvidas a partir de padrões de usabilidade, discutidos no Capítulo 6.

Capítulo 5. : Especificação e Arquitetura

5.1. Considerações iniciais

Sistemas de geração de hiperdocumentos, como o *eMeet/SERVE*, possuem uma série de necessidades comuns, como, por exemplo, a constante necessidade de atualização de conteúdo e de *layout*. Tais necessidades nem sempre são previstas por sistemas responsáveis em realizar esse tipo de tarefa. Além disso, a instalação e a manutenção desses sistemas, muitas vezes, são realizadas com alterações no código-fonte da aplicação, demandando muito esforço por parte dos desenvolvedores (Pressman, 2000).

A especificação e análise de requisitos do *eMeet/SERVE* foi realizada utilizando conceitos baseados em UML (*Unified Modeling Language*) (Rosenberg & Scott, 1999). Somente alguns passos da especificação UML foram utilizados na modelagem do *eMeet/SERVE*.

O *eMeet/SERVE* foi construído a partir de uma arquitetura voltada a sistemas de geração de hiperdocumentos, que prevê funcionalidades como: integração com outras APIs, independência de base de dados, suporte a múltiplos idiomas, *layout* customizável e organização interna para melhor instalação e manutenção dos sistemas, dentre outras (Eleutério & Pimentel, 2001). Tal arquitetura, denominada LAIAc (*Layout, Acesso, Idiomas e Aplicação customizáveis*), tem seus conceitos básicos apresentados neste capítulo. Além disso, a especificação, a análise de requisitos e a forma como o *eMeet/SERVE* utilizou a arquitetura LAIAc também são discutidas.

5.2. *eMeet/SERVE*: Especificação, Modelagem e Características

Durante a fase de análise de requisitos³⁰ do *eMeet/SERVE*, parte da especificação UML (Rosenberg & Scott, 1999) foi utilizada. A especificação inicial, os diagramas UML de estudo de caso e as seções e características do *eMeet/SERVE* estão ilustrados a seguir.

³⁰ A fase de análise de requisitos é uma das primeiras fases propostas pela Engenharia de Software para o desenvolvimento de novos softwares (Pressman, 2000).

5.2.1. Especificação Inicial do *eMeet/SERVE*

O *eMeet/SERVE* foi desenvolvido a partir da seguinte especificação inicial:

- O *eMeet/SERVE* deve ser uma aplicação Web construída a partir de tecnologias que possuam distribuição e utilização gratuitas. Porém, o *eMeet/SERVE* deve ser compatível com outras tecnologias;
- O *eMeet/SERVE* deve gerenciar o cadastro e remoção de reuniões, salas virtuais e usuários. No caso de usuários, o *eMeet/SERVE* deve possuir integração com o servidor de usuários já desenvolvido;
- Os hiperdocumentos de apresentação das informações capturadas devem contemplar elementos como áudio/vídeo, linha do tempo e sequência de *slides*. Tais elementos devem ser sincronizados entre eles e possuírem interação com os usuários;
- O *eMeet/SERVE* deve ser construído utilizando-se o conceito de APIs, a fim de que seus serviços possam ser reutilizados por outras aplicações.

5.2.2. Diagramas UML de Estudo de Caso.

Os diagramas de estudos de caso são uma sequência de ações que um ator, que é normalmente um usuário, realiza dentro do sistema a fim de atingir um objetivo em particular. A relação entre os atores e as ações, e entre ações, são representadas por uma seta. As relações nunca podem ocorrer entre atores, uma vez que atores são elementos externos ao sistema. As representações de ator, ação e sua relação, na especificação UML, estão ilustrados na Figura 16.

Figura 16: Representação de ator, ação e relação em UML.

Para a modelagem do *eMeet/SERVE*, cinco diagramas de estudo de caso foram realizados, discutidos a seguir.

5.2.2.1. Gerenciamento de Salas Virtuais

Conforme especificação inicial, o *eMeet/SERVE* deve contemplar o gerenciamento de salas virtuais. Isso inclui o cadastro, edição e remoção de salas. As salas são utilizadas para relacionar participantes em uma reunião (mais detalhes encontram-se na Seção 6.5.). A

Figura 17 mostra o diagrama de estudos de caso para o gerenciamento de salas. Usuários poderão criar salas, visualizar a lista de salas criadas, editar dados de salas e remover salas, desde que tenham permissão para realizar tais operações.

Figura 17: Diagrama UML de estudo de caso para o gerenciamento de salas virtuais no *eMeet/SERVE*.

5.2.2.2. Gerenciamento de Usuários

O *eMeet/SERVE* também deve contemplar o gerenciamento de usuários, com integração com o Servidor de Usuários já existente. Isso inclui o cadastro, edição e remoção de usuários. Os usuários podem se tornar participantes de reuniões (mais detalhes encontram-se na Seção 6.5.). A Figura 18 mostra o diagrama de estudos de caso para o gerenciamento de usuários. Usuários poderão criar novos usuários, editar dados de usuários e remover usuários, desde que tenham permissão para realizar tais operações.

Figura 18: Diagrama UML de estudo de caso para o gerenciamento de usuários no *eMeet/SERVE*.

5.2.2.3. Gerenciamento de Reuniões

O gerenciamento de reuniões é outra funcionalidade que o *eMeet/SERVE* deve contemplar. Isso inclui o cadastro, visualização, edição e remoção de reuniões. A Figura 19 mostra o diagrama de estudos de caso para o gerenciamento de reuniões. A ação “Visualizar Reunião” é composta de outras ações e está discutida na próxima sub-seção. Usuários poderão criar reuniões, visualizar a lista de reuniões permitidas de acordo com seus perfis (mais detalhes encontram-se na Seção 6.5.), editar dados de reuniões e remover reuniões, desde que tenham permissão para realizar tais operações.

Figura 19: Diagrama UML de estudo de caso para o gerenciamento de reuniões no *eMeet/SERVE*. A ação "Visualizar Reunião" é composta de outras ações, ilustradas na próxima sub-seção.

5.2.2.4. Visualizar Reunião

A visualização de informações nos hiperdocumentos gerados das reuniões distribuídas capturadas é o principal elemento de interação entre usuários e o *eMeet/SERVE*. A Figura 20 mostra o diagrama de estudos de caso para essa interação. Para visualizar a reunião desejada é necessário que usuários tenham permissão de acesso nessa reunião (mais detalhes encontram-se na Seção 6.5.). Enquanto visualizam as informações das reuniões distribuídas capturadas, usuários poderão interagir com a reprodução de vídeo, com os *slides* e com a linha do tempo (mais detalhes encontram-se na Seção 6.4.).

Figura 20: Diagrama UML de estudo de caso para a visualização de reuniões no *eMeet/SERVE*.

5.2.2.5. Envio de Mensagem – Fale Conosco

Usuários também poderão enviar mensagens de contato para os autores/mantenedores do *eMeet/SERVE*. A Figura 21 mostra o diagrama de estudo de caso para esta funcionalidade.

Figura 21: Diagrama UML de estudo de caso para o envio de mensagens aos autores/mantenedores no *eMeet/SERVE*.

5.2.3. Seções e Características do *eMeet/SERVE*

A partir dos diagramas de estudo de caso, concluiu-se que o *eMeet/SERVE* necessitaria de quatro seções independentes, além do hiperdocumento inicial, que é responsável pela apresentação do sistema.

As quatro seções independentes e suas funcionalidades básicas são:

1. Seção Reuniões: responsável pela criação, edição, visualização e remoção de reuniões no *eMeet/SERVE*. A visualização das informações capturadas das reuniões de forma sincronizada é a principal funcionalidade do *eMeet/SERVE*.
2. Seção Salas: responsável pela criação, edição e remoção de salas virtuais no *eMeet/SERVE*.
3. Seção Usuários: responsável pela criação, edição e remoção de usuários no *eMeet/SERVE*, com integração com o Servidor de Usuários.
4. Seção Fale Conosco: responsável pelo envio de mensagens de usuários aos autores/mantenedores do *eMeet/SERVE*.

Pode-se observar, na Figura 6, que o menu lateral esquerdo é composto por cinco ligações, responsáveis em fazer a ligação de hiperdocumentos para as quatro seções descritas, mais o hiperdocumento inicial (mais detalhes sobre a utilização do *eMeet/SERVE* e suas seções encontram-se no Apêndice A – Manual do Usuário do *eMeet/SERVE*).

5.3. Arquitetura LAIAc proposta para sistemas de geração de hiperdocumentos

Para o desenvolvimento do *eMeet/SERVE* foi proposta uma arquitetura, denominada LAIAc (*Layout, Acesso, Idioma e Aplicação* customizáveis). A arquitetura LAIAc pode ser utilizada por

sistemas de geração de hiperdocumentos, bem como por outros sistemas que necessitem de uma melhor organização estrutural.

Tal arquitetura, ilustrada na Figura 22, se utilizada, garante aos sistemas uma série de vantagens durante a fase de programação e, principalmente, durante a fase de manutenção do sistema. Além disso, serviços implementados em um sistema que utilize a arquitetura LAIAc podem ser reutilizados naturalmente por outros.

A arquitetura básica é composta de cinco módulos principais: aplicação, módulo de *layout*, módulo de idiomas, módulo de acesso e módulo de dados. A aplicação é a parte responsável por gerar os hiperdocumentos. Para isso, ela recupera as definições e os dados dos outros módulos e realiza o processamento. A integração com outras aplicações deve ser realizada pelo módulo de acesso.

Figura 22: Esboço da arquitetura LAIAc proposta para sistemas de geração de hiperdocumentos. Os retângulos tradicionais representam os módulos da arquitetura. Os retângulos com canto dobrado indicam os hiperdocumentos. O retângulo com borda arredondada indica integração com outras aplicações. As setas indicam o fluxo de informação durante o processo de geração dos hiperdocumentos.

Essa arquitetura básica pode ser implementada independente da linguagem de programação utilizada, uma vez que ela é composta de módulos que podem ser implementados por praticamente qualquer linguagem de programação.

O objetivo principal da arquitetura LAIAc é minimizar a necessidade de reprogramação das aplicações, devido a alterações rotineiras de estrutura, armazenamento, serviços, idiomas e algumas definições de *layout*, deixando o sistema preparado para a fase de manutenção e aumentando a flexibilidade da aplicação. A principal desvantagem de utilizar tal arquitetura é que os desenvolvedores despenderão um tempo maior durante a implementação dos sistemas, pois devem respeitar algumas regras durante o desenvolvimento de cada módulo da arquitetura. Os módulos e regras da arquitetura LAIAc são explicados a seguir.

5.3.1. Módulo de dados

Os dados de um sistema podem estar armazenados em base de dados, arquivos, planilhas, dentre outros. O módulo de dados é responsável por armazenar todas as informações utilizadas pelo sistema.

Este é o módulo de mais baixo nível do sistema, onde as informações estão armazenadas fisicamente em disco, fitas ou qualquer outro dispositivo de dados.

Recomenda-se a utilização de pelo menos uma base de dados para gerenciar a inclusão e remoção de informações com segurança. No caso de ser utilizada uma base de dados relacional, esse acesso pode ser feito através da linguagem SQL. A linguagem SQL, em sua forma básica, é interpretada por qualquer base de dados relacional existente. Desenvolvedores e administradores de base de dados podem realizar operações em qualquer base de dados relacional utilizando comandos comuns da linguagem SQL.

Se o banco de dados utilizado não tiver suporte a arquivos multimídia, recomenda-se que informações como vídeos e imagens sejam armazenados em arquivos, ou seja, estejam fisicamente armazenados na estrutura de diretórios do sistema operacional. Ao armazenar informações em arquivos, é importante verificar as permissões dos diretórios que serão utilizados. Apesar de estarem fora do banco de dados, recomenda-se que os nomes e os caminhos completos dos diretórios dos arquivos estejam armazenados na base de dados, a fim de manter uma melhor consistência das informações.

O módulo de dados é o que exige um menor esforço dos desenvolvedores, uma vez que grande parte dos mecanismos de gerenciamento dos dados já está implementada pelas bases de dados e sistemas operacionais. Cabe ao desenvolvedor realizar um bom projeto sobre como armazenar os dados do sistema.

É importante salientar que na arquitetura LAIAc proposta, somente o módulo de acesso pode trocar dados com o módulo de dados; a aplicação não pode acessar o módulo de dados diretamente.

5.3.2. Módulo de acesso

O módulo de acesso é a camada intermediária entre a aplicação e o módulo de dados. O módulo de acesso é responsável por receber da aplicação as chamadas de armazenamento e recuperação de informações e acessar o módulo de dados de forma correta e segura, realizando a operação desejada.

Os serviços requeridos pela aplicação, em termos de armazenamento e recuperação de dados, devem estar implementados neste módulo. Um exemplo seria o cadastro de usuários no sistema, no qual a aplicação chamaria o módulo de acesso dizendo que gostaria de armazenar um novo usuário no sistema. A partir daí, o módulo de acesso verifica onde tal informação deverá ser armazenada e acessa o módulo de dados para realizar a operação. O módulo de acesso retorna para a aplicação se a operação foi realizada com sucesso ou não.

No exemplo citado, é possível verificar que a aplicação não tem controle sobre onde os dados de um usuário estão armazenados fisicamente. Essa é a grande vantagem da utilização de um módulo de acesso, pois o local de armazenamento pode ser trocado sem que haja necessidade de reescrita de parte do código da aplicação.

Todas as funções que acessam os dados devem ser implementadas neste módulo como serviços. Assim, outras aplicações podem utilizar as funções do módulo de acesso, o que promove a reutilização dos serviços implementados. Além disso, o módulo de acesso poderá utilizar serviços disponíveis de outras aplicações.

O módulo de acesso deve controlar o acesso a informações que estejam armazenadas no módulo de dados, podendo inclusive acessar APIs já desenvolvidas de outras aplicações. O importante é que a aplicação sempre ativará o módulo de acesso para tratar das atividades de armazenamento e recuperação de dados e de integração com outras aplicações, independentemente de como elas são realizadas. Dessa maneira, a aplicação não precisa ser reprogramada caso venha a ocorrer qualquer alteração com esses serviços.

A desvantagem da utilização de um módulo de acesso é que os desenvolvedores despenderão um tempo maior durante a implementação, pois deverão implementar serviços com algoritmos genéricos que possam ser reusados por outros sistemas.

5.3.3. Módulo de idiomas

As definições de idiomas são fundamentais quando sistemas devem conter suporte a múltiplos idiomas. Além disso, textos que serão exibidos aos usuários podem ser trocados em um único lugar, facilitando a manutenção do sistema.

O módulo de idiomas é responsável por conter textos que serão exibidos nos hiperdocumentos aos usuários, gerados pela aplicação. Quando a aplicação precisa gerar um hiperdocumento que contenha textos, a aplicação acessa o módulo de idiomas e obtém o texto de acordo com o idioma desejado.

Todos os textos utilizados na geração de hiperdocumentos pelo sistema devem estar definidos neste módulo, ou seja, a aplicação não pode conter textos em seu código-fonte.

Textos podem estar definidos em diversos idiomas, como português, inglês e espanhol, por exemplo. Esta é a grande vantagem do módulo de idiomas. A aplicação pode gerar um mesmo hiperdocumento em idiomas diferentes, sem que haja necessidade de reescrita do código da aplicação.

Outra vantagem do módulo de idiomas é que novos idiomas podem ser incluídos posteriormente sem modificação da aplicação. Basta criar uma nova definição dos textos no idioma desejado e acrescentá-lo neste módulo. A aplicação utilizará esta nova definição se o módulo de idiomas estiver configurado para fornecer o novo idioma criado.

Assim como no módulo de acesso, a desvantagem da utilização de um módulo de idiomas é que os desenvolvedores despenderão um tempo maior durante a implementação do sistema, pois será necessário separar todos os textos da aplicação em um módulo à parte.

5.3.4. Módulo de *layout*

O *layout* é uma das características dos hiperdocumentos que mais necessitam de mudanças com o tempo. Muitas vezes, desenvolvedores precisam realizar diversas mudanças na aplicação a fim de alterar o visual dos hiperdocumentos gerados.

O módulo de *layout* é responsável por configurar algumas definições de *layout* comuns aos hiperdocumentos gerados. Hiperdocumentos possuem características comuns com respeito ao *layout*, como cor de fundo, tamanho de fonte, tipo de fonte, cor de fonte, cor de tabela, tamanho de tabela, entre outras. Tais características não devem ser definidas na aplicação e sim, neste módulo.

Ao gerar o *layout* de hiperdocumentos, a aplicação deve consultar o módulo de *layout* e obter as definições desejadas. Portanto, pode-se mudar as cores e fontes de um hiperdocumento, por exemplo, apenas alterando a configuração do módulo de *layout*. Com isso, os hiperdocumentos gerados podem se “adaptar” ao ambiente visual onde eles forem utilizados, sem que haja reprogramação do código da aplicação.

É importante observar que imagens não são controladas por este módulo. Portanto, ao alterar o *layout* de hiperdocumentos é necessário verificar se algumas imagens devem ser editadas e ter suas cores e fontes trocadas manualmente. Apesar desse fator, não há necessidade de reprogramação da aplicação, pois a edição das imagens existentes pode ser feita independentemente da aplicação.

5.3.5. Aplicação

A aplicação é a parte principal do sistema, que é responsável por reunir os dados necessários e realizar o processamento para a geração de hiperdocumentos.

Algumas regras devem ser seguidas durante a programação da aplicação, para que as vantagens propostas pela arquitetura sejam obtidas. São elas:

- Proibido o acesso direto aos dados: a aplicação jamais deve acessar o módulo de dados diretamente. Isso inclui acesso à base de dados, que pode ser a principal fonte de dados do sistema. Quando a aplicação necessitar de operações que envolvam armazenamento e recuperação de dados, a aplicação deve fazer uma chamada ao módulo de acesso.
- Proibido o uso de textos e literais³¹: a aplicação não pode conter textos e literais em seu código-fonte. Todos os textos devem estar contidos no módulo de idiomas. Quando a aplicação necessitar da definição de um texto, ela deve buscá-lo no módulo de idiomas, que retornará o texto no idioma configurado. As literais utilizadas podem estar em um arquivo de configuração da aplicação.
- Proibido a definição de cores, fontes e tamanhos: assim como os textos e dados, a aplicação não deve definir cores e fontes para gerar hiperdocumentos. As definições devem estar no módulo de *layout*.
- Uma aplicação não deve acessar diretamente outras aplicações: a aplicação deve utilizar o seu próprio módulo de acesso ou, se existir, o módulo de acesso de outras aplicações. Uma aplicação jamais deverá referenciar outra aplicação diretamente, sem que haja, pelo menos, uma camada (ou módulo) intermediária(o).

Além dessas regras, a aplicação deve conter suas próprias configurações. Tais configurações devem estar em um arquivo de configuração próprio, conforme ilustra a Figura 22. Essas configurações devem conter aspectos mutáveis da aplicação, como, por exemplo, o *e-mail* de contato do *webmaster*³². Tal atributo pode mudar freqüentemente e não é desejado que haja uma busca nos arquivos da aplicação a fim de alterar tal valor. Definições desse tipo devem estar, portanto, em arquivos de configuração.

Para garantir que os módulos da arquitetura serão utilizados pelos arquivos da aplicação, recomenda-se a criação de arquivos auxiliares que realizem a inclusão de todos os módulos da

³¹ Literais são expressões e/ou variáveis com valores definidos.

³² *Webmaster* normalmente é a pessoa mantedora do *site* responsável por receber críticas e sugestões dos usuários.

arquitetura. Dessa forma, todos os arquivos da aplicação devem incluir esses arquivos auxiliares e estarão incluindo automaticamente os módulos da arquitetura LAIAc.

Sistemas que contemplem a arquitetura LAIAc proposta, como é o caso do *eMeet/SERVE*, terão as seguintes vantagens:

- Independência de base de dados e dispositivos de armazenamento;
- Suporte a múltiplos idiomas;
- *Layout* configurável em alguns aspectos comuns;
- Integração com reutilização dos serviços por outras aplicações;
- Preparação para manutenção e expansão.

5.4. O *eMeet/SERVE* no contexto da arquitetura LAIAc

Por se tratar de um sistema de geração de hiperdocumentos, o *eMeet/SERVE* foi implementado segundo a arquitetura LAIAc. Além de implementar todas os módulos existentes na arquitetura, por possuir características próprias, o *eMeet/SERVE* estendeu alguns módulos. A linguagem PHP³³ foi utilizada para implementar o *eMeet/SERVE*.

A arquitetura do *eMeet/SERVE* está ilustrada na Figura 23. Pode-se observar que a arquitetura do *eMeet/SERVE* respeita a arquitetura LAIAc, ilustrada na Figura 22.

Figura 23: Arquitetura LAIAc utilizada no desenvolvimento do *eMeet/SERVE*.

³³ Mais informações sobre a linguagem PHP encontram-se no endereço <http://www.php.net>.

Cada módulo do *eMeet/SERVE*, dentro da arquitetura proposta, está detalhado a seguir.

5.4.1. Módulo de dados do *eMeet/SERVE*

O módulo de dados do *eMeet/SERVE* é composto por tabelas em banco de dados e arquivos em diretórios do sistema operacional. No banco de dados são armazenadas todas as informações do sistema, como, por exemplo, quais reuniões ocorreram e quem participou. Em arquivos são armazenadas as imagens dos *slides*, bem como os vídeos (mídias). Os nomes desses arquivos também estão armazenados no banco de dados, para uma melhor consistência entre os dados. Qualquer banco de dados existente pode ser utilizado pelo *eMeet/SERVE*, desde que as funções do módulo de acesso correspondente sejam implementadas.

O banco de dados escolhido foi o MySQL, principalmente por ser um banco de dados de livre utilização e apresentar desempenho satisfatório. O modelo entidade relacionamento (MER) e as tabelas criadas estão detalhados a seguir. No Apêndice B encontra-se o código utilizado para criação das tabelas do *eMeet/SERVE* no banco de dados MySQL.

5.4.1.1. Modelo Entidade Relacionamento (MER)

A Figura 24 mostra o Modelo Entidade Relacionamento (MER) da base de dados utilizada pelo *eMeet/SERVE*.

Figura 24: Modelo Entidade Relacionamento (MER) da base de dados do *eMeet/SERVE*. O MER é composto de cinco tabelas, responsáveis em armazenar dados e metadados sobre as reuniões capturadas.

Para a construção do MER, uma regra foi proposta para nomeação das colunas. Tal regra consiste de que o nome de cada campo deve iniciar com as três primeiras letras do nome de sua tabela. Assim sendo, todos os nomes das colunas da tabela “room” devem iniciar com “roo”. Além disso, as demais letras que constituem o nome da coluna, devem ser agrupadas de três em três. Por exemplo, o complemento “nam” da coluna “roonam” é a abreviação em três letras da palavra “name”. Os campos que forem chaves estrangeiras devem manter seus nomes originais.

O *eMeet/SERVE* possui cinco tabelas que fazem o armazenamento de dados e metadados da aplicação. Tais tabelas são discutidas a seguir.

Tabela room

A tabela `room` é a responsável pelo armazenamento das informações das salas de reunião. Tais salas são utilizadas para indicar a localização física dos participantes das reuniões.

Figura 25: Tabela `room` e seus campos.

Os campos da tabela `room`, ilustrados na Figura 25, são:

- `roocod`: código numérico da sala de reunião. É a chave primária da tabela. O valor desse campo é incrementado automaticamente assim que uma nova sala é inserida no sistema.
- `roonam`: nome da sala. O nome da sala é fundamental para que usuários possam encontrar uma sala no sistema, sem precisar do código.
- `rooadd`: endereço da sala. Este campo é utilizado para armazenar o endereço da sala em questão, além de outras informações que melhor detalhem a sala.

Tabela meeting_room

A tabela `meeting_room` é a responsável pelo armazenamento da relação entre as salas de reunião, representadas pela tabela `room`, as reuniões, representadas pela tabela `meeting` e os usuários do sistema. A tabela `meeting_room` é utilizada para indicar que um usuário participou de uma reunião em uma sala.

Figura 26: Tabela `meeting_room` e seus campos.

Os campos da tabela `meeting_room`, ilustrados na Figura 26, são:

- `mtgcod`: código numérico da reunião. É uma chave estrangeira referente à tabela `meeting`.
- `roocod`: código numérico da sala de reunião. É uma chave estrangeira referente à tabela `room`.
- `usrcod`: código numérico do usuário.

Os três campos acima compõem a chave primária da tabela.

O valor do campo `usrcod` é obtido através da integração do *eMeet/SERVE* com o servidor de usuários, que é o responsável por tratar e armazenar todas as operações como cadastro e remoção de usuários. Portanto, o *eMeet/SERVE* está integrado com esse serviço e obtém o valor de `usrcod` a partir do servidor de usuários (maiores informações encontram-se na Seção 5.4.2.2).

Tabela meeting

A tabela `meeting` é a responsável pelo armazenamento das informações das reuniões do *eMeet/SERVE*. Todas as reuniões existentes no sistema são armazenadas nesta tabela, que é a principal da base de dados.

meeting	
mtgcod:	NUMBER
mtgnam:	VARCHAR2(100)
mtgvid:	VARCHAR2(40)
mtgstadat:	DATE
mtgenddat:	DATE
grpcod:	NUMBER
mtgdes:	VARCHAR2(255)

Figura 27: Tabela `meeting` e seus campos.

Os campos da tabela `meeting`, ilustrados na Figura 27, são:

- `mtgcod`: código numérico da reunião. É a chave primária da tabela. O valor desse campo é incrementado automaticamente assim que uma nova reunião é inserida no sistema. Este campo é exportado para diversas outras tabelas do sistema, como chave estrangeira.
- `mtgnam`: nome da reunião. O nome da reunião é fundamental para que usuários possam encontrar uma reunião no sistema, sem precisar do código.
- `mtgvid`: nome do arquivo de vídeo da reunião. Este campo contém o nome do arquivo de vídeo que foi gerado para essa reunião, podendo conter também o diretório onde está armazenado.
- `mtgstadat`: data/hora de início da reunião. Este campo marca o tempo exato do início da reunião, com precisão de segundos. Este valor é importantíssimo para efeitos de sincronia entre os *slides* e o vídeo da reunião.
- `mtgenddat`: data/hora de término da reunião. Este campo marca o tempo exato do término da reunião, com precisão de segundos.
- `grpcod`: código de grupo da reunião. Este código é utilizado para controlar o acesso às informações da reunião, segundo o grupo de permissões armazenadas. Existem

quatro tipos de controle de acesso aos dados das reuniões, que variam desde acesso livre até restrito ao administrador do sistema (maiores informações na Seção 6.5).

- `mtgdes`: descrição da reunião. Este campo é utilizado para melhor descrever a reunião que foi armazenada. Pode conter inclusive informações sobre a pauta ou a ata da reunião.

Tabela slide

A tabela `slide` é a responsável pelo armazenamento das informações dos *slides* das reuniões. Os *slides* são um dos componentes capturados das reuniões, assim como o vídeo. Esta tabela deve ser preenchida durante a fase de captura, pelo *eMeet/InCA*.

Figura 28: Tabela `slide` e seus campos.

Os campos da tabela `slide`, ilustrados na Figura 28, são:

- `slicod`: código numérico do *slide* de uma reunião. Este valor inicia-se com 1 a cada nova reunião e é incrementado sequencialmente. Este campo também indica a ordem que os *slides* foram percorridos no decorrer de uma reunião.
- `mtgcod`: código numérico da reunião. É uma chave estrangeira referente à tabela `meeting`.
- `sligif`: nome do arquivo do *slide*. Este campo contém o nome do arquivo de imagem que foi gerado para um *slide* de uma reunião, podendo conter o diretório onde está armazenado.
- `slitst`: *timestamp* do *slide*. Este campo armazena o *timestamp* exato que o *slide* foi mostrado durante a reunião; é por esse valor que os *slides* serão sincronizados com o vídeo, durante a exibição da reunião.

Os dois primeiros campos acima compõem a chave primária da tabela.

Tabela penstroke

A tabela `penstroke` é a responsável pelo armazenamento de cada anotação feita nos *slides*. Tais anotações, chamadas de *penstrokes*, são armazenadas para que possa haver a ligação direta entre as anotações e a parte da apresentação correspondente.

penstroke	
pencod:	NUMBER
slicod:	NUMBER
mtgcod:	NUMBER
pengif:	VARCHAR2(40)
pentst:	NUMBER
pentopx:	NUMBER
pentopy:	NUMBER
penbotx:	NUMBER
penboty:	NUMBER

Figura 29: Tabela *penstroke* e seus campos.

Os campos da tabela *penstroke*, ilustrados na Figura 29, são:

- *pencod*: código numérico da anotação. Indica a ordem de anotações que foram feitas em cada *slide* de uma reunião.
- *slicod*: código numérico do *slide* de uma reunião. Juntamente com o campo *mtgcod* é uma chave estrangeira da tabela *slide*.
- *mtgcod*: código numérico da reunião.
- *pengif*: nome do arquivo de imagem da anotação. O nome da imagem referente à anotação é armazenado neste campo, que pode conter o diretório onde está armazenado.
- *pentst*: *timestamp* da anotação. Este campo armazena o *timestamp* exato de quando a anotação foi realizada durante a reunião. É por esse valor que as anotações são sincronizadas com a apresentação da reunião.
- *pentopx*: posição da anotação. Coordenada superior esquerda de um retângulo.
- *pentopy*: posição da anotação. Coordenada superior direita de um retângulo.
- *penbotx*: posição da anotação. Coordenada inferior esquerda de um retângulo.
- *penboty*: posição da anotação. Coordenada inferior direita de um retângulo.

Os três primeiros campos acima compõem a chave primária da tabela.

Os quatro últimos campos constituem as coordenadas de um retângulo, a fim de marcar a posição da anotação dentro de um *slide*. Esses valores são utilizados para implementar a ligação das anotações com determinadas partes da apresentação da reunião.

5.4.1.2. Dicionário de Dados

Para permitir uma consulta rápida sobre a disposição das tabelas e seus campos, foi elaborado o dicionário de dados da base de dados do *eMeet/SERVE*. O dicionário de dados é fundamental para que outros programadores possam realizar a manutenção e extensão do sistema. O dicionário de dados está representado pela Tabela 2. Na Tabela 2, “PK” e “FK” indicam se o

campo é uma chave primária ou chave estrangeira, respectivamente. “Null” indica que o campo pode ser vazio e “Not Null” indica que o campo deve ser obrigatoriamente preenchido com algum valor.

Tabela 2: Dicionário de dados do *eMeet/SERVE*.

Nome da Tabela	Comentário	Nome da coluna	Tipo de dado	Null	Comentário	PK	FK
ROOM	Armazena as salas do sistema.	roocod	NUMBER	NOT NULL	Código da sala	Sim	Não
		roonam	VARCHAR2(100)	NOT NULL	Nome da sala	Não	Não
		rooaddd	VARCHAR2(255)	NULL	Endereço da sala	Não	Não
MEETING_ROOM	Armazena a relação reunião, sala e usuário.	mtgcod	NUMBER	NOT NULL	Código da reunião	Sim	Sim
		roocod	NUMBER	NOT NULL	Código da sala	Sim	Sim
		usrcod	NUMBER	NOT NULL	Código do usuário	Sim	Não
MEETING	Armazena as reuniões do sistema	mtgcod	NUMBER	NOT NULL	Código da reunião	Sim	Não
		mtgnam	VARCHAR2(100)	NOT NULL	Nome da reunião	Não	Não
		mtgvid	VARCHAR2(40)	NULL	Vídeo da reunião	Não	Não
		mtgstadat	DATETIME	NULL	Data inicial da reunião	Não	Não
		mtgenddat	DATETIME	NULL	Data final da reunião	Não	Não
		grpcod	NUMBER	NULL	Controle de acesso	Não	Não
		mtgdes	VARCHAR2(255)	NULL	Descrição da reunião	Não	Não
SLIDE	Armazena os slides das reuniões.	slicod	NUMBER	NOT NULL	Código do <i>slide</i>	Sim	Não
		mtgcod	NUMBER	NOT NULL	Código da reunião	Sim	Sim
		sligif	VARCHAR2(40)	NULL	Arquivo do <i>slide</i>	Não	Não
		slitst	NUMBER	NULL	<i>Timestamp</i> do <i>slide</i>	Não	Não
PENSTROKE	Armazena as anotações nos slides.	pencod	NUMBER	NOT NULL	Código da anotação	Sim	Não
		slicod	NUMBER	NOT NULL	Código do <i>slide</i>	Sim	Sim
		mtgcod	NUMBER	NOT NULL	Código da reunião	Sim	Sim
		pengif	VARCHAR2(40)	NULL	Arquivo da anotação	Não	Não
		pentst	NUMBER	NULL	<i>Timestamp</i> anotação	Não	Não
		pentopx	NUMBER	NULL	Posição top x	Não	Não
		pentopy	NUMBER	NULL	Posição top y	Não	Não
		penbotx	NUMBER	NULL	Posição bot x	Não	Não
		penboty	NUMBER	NULL	Posição bot y	Não	Não

5.4.2. Módulo de acesso do *eMeet/SERVE*

O *eMeet/SERVE* foi desenvolvido para ser adaptável para qualquer base de dados existente, graças ao módulo intermediário (módulo de dados) entre a aplicação e a base de dados, que pode ser substituído de acordo com a base de dados desejada.

Isso significa que não há interação direta entre a aplicação e a base de dados, conforme definida pela arquitetura LAIAc, apresentada na Figura 22. A aplicação faz chamadas somente para o módulo de acesso, que é o responsável por se comunicar com o módulo de dados ou com outras aplicações, e retornar os valores para a aplicação. Este módulo intermediário, chamado de módulo de acesso, faz com que todas as funções de acesso à base de dados sejam reaproveitadas pela aplicação, havendo um maior reuso do código.

Mais detalhadamente, no *eMeet/SERVE* o módulo de acesso é composto por arquivos mestres e arquivos específicos, conforme ilustrado na Figura 30.

Arquivos mestres são arquivos ativados pela aplicação, sendo comuns para qualquer base de dados e dificilmente alterados pelos desenvolvedores. Sendo assim, aplicações sempre realizam chamadas a arquivos mestres para acessar a base de dados desejada, e como tais arquivos dificilmente mudam, há uma garantia de que não haverá necessidade de reescrita da aplicação. Os arquivos mestres incluem os arquivos específicos, de acordo com a base de dados desejada.

Os arquivos específicos são os responsáveis em se comunicar diretamente com a base de dados. Existe um arquivo específico para cada base de dados cujo módulo de acesso seja compatível. Caso deseja-se ampliar o módulo de acesso para dar suporte a mais uma base de dados, o único trabalho a ser realizado é criar um arquivo específico no módulo de acesso para a base de dados desejada. Por definição, o arquivo específico deve ter o nome da base de dados correspondente.

Figura 30: Detalhamento do módulo de acesso e integração com a aplicação e com o módulo de dados

De acordo com a Figura 30, observa-se que o *eMeet/SERVE* pode ser instalado em diversas bases de dados, de acordo com a configuração desejada. Tudo isso é feito sem que haja alterações no código-fonte da aplicação, garantindo a independência de base de dados para a aplicação.

Atualmente, o módulo de dados do *eMeet/SERVE* possui apenas o arquivo específico para a base de dados MySQL. Porém, ele pode ser adaptado para funcionar em outras bases de dados.

Segundo a Figura 23, o módulo de acesso é composto por três sub-módulos: o módulo de acesso às informações sobre as sessões das reuniões, o módulo de acesso às mídias e o módulo de validação de usuários. Tais sub-módulos são implementados através de arquivos específicos, que são ativados a partir de arquivos mestres. Tais arquivos são discutidos a seguir.

5.4.2.1. Arquivos mestres do módulo de acesso do eMeet/SERVE

Dois arquivos mestres foram criados no *eMeet/SERVE*. Tais arquivos são incluídos em todos os arquivos da aplicação e estão armazenados em:

`emeet/includes/db/header.php` e `emeet/includes/db/footer.php`

O `header.php` é o arquivo responsável por abrir a conexão com uma determinada base de dados, incluir o arquivo específico de funções para essa base de dados, além de incluir chamadas para utilização dos serviços oferecidos pelo servidor de usuários. Esse arquivo é o responsável por incluir os arquivos específicos `mysql.php` e `apiuser.php`, presentes no mesmo diretório (`emeet/includes/db/`).

O `footer.php` é o arquivo responsável por fechar as conexões abertas pelo `header.php`.

5.4.2.2. Arquivos específicos do módulo de acesso do eMeet/SERVE.

O arquivo `apiuser.php` é o responsável por realizar a comunicação com os serviços do servidor de usuários. Dessa maneira, quando o *eMeet/SERVE* precisa de uma operação com usuários, como, por exemplo, a autenticação de usuários no sistema, esse arquivo é o responsável por acessar os serviços disponíveis no servidor de usuários e retornar os dados à aplicação.

As funções de comunicação implementadas nesse arquivo, utilizadas pelo *eMeet/SERVE*, são:

- `new_user($usnam, $usreml, $usrlog, $usrpwd)`: cria um novo usuário no sistema. Recebe o nome, *e-mail*, *login* e senha (encriptografada através de md5³⁴) e retorna o código do usuário criado (`usrcod`). Quando um novo usuário é criado a partir do *eMeet/SERVE*, ele automaticamente pertence a um grupo específico. Esse grupo padrão é configurado no arquivo de configuração da aplicação do *eMeet/SERVE*.
- `select_all_users()`: seleciona todos os usuários do sistema. Não recebe nenhum parâmetro e retorna uma matriz com todas as informações de todos os usuários, código do usuário, nome do usuário e *login* do usuário, através dos parâmetros `usrcod`, `usnam` e `usrlog`.

³⁴ O md5 é um algoritmo responsável por encriptografar dados de forma *one-way*, ou seja, dados que são encriptografados com o md5 não podem ser descriptografados. O md5 é geralmente utilizado para armazenar informações confidenciais, como senhas. Mais informações encontram-se no endereço <http://www.php.net>.

- `select_user_by_cod($usrcod)`: seleciona as informações de um usuário através do código do mesmo. Recebe o código do usuário como parâmetro e retorna um vetor com todas as informações do usuário desejado, como código do usuário, nome do usuário e *login* do usuário, através dos parâmetros `usrcod`, `usrcod` e `usrcod`.
- `update_user($usrcod, $usrcod, $usrcod, $usrcod, $usrcod)`: atualiza os dados de um usuário. Recebe o código do usuário, o nome do usuário, *login* do usuário e senha (encriptografada através de md5) do usuário como parâmetros e retorna `true` ou `false` dependendo se a atualização foi executada com sucesso ou não.
- `remove_user($usrcod)`: remove um usuário do sistema. Recebe o código do usuário. Se o usuário estiver participando de uma reunião, ele exibe uma mensagem que não é possível removê-lo. Retorna `true` ou `false` dependendo se a remoção foi executada com sucesso ou não.
- `autentic_user($usrcod, $usrcod)`: faz a autenticação de um usuário no sistema. Recebe o *login* e a senha (encriptografada através de md5) como parâmetros e retorna o código do usuário (`usrcod`), se a autenticação for bem sucedida, ou `false` (zero), se a autenticação não foi válida.

O arquivo `mysql.php` é o responsável por conter todas as funções do *eMeet/SERVE* de acesso a base de dados MySQL, que está presente no módulo de dados.

As funções implementadas estão descritas a seguir.

- `new_room($roonam, $rooadd)`: cria uma sala no sistema. Recebe o nome e endereço da sala como parâmetros e retorna o código da sala criada, através do parâmetro `roocod`.
- `select_all_rooms()`: seleciona todas as salas do sistema. Não recebe nenhum parâmetro e retorna uma matriz com todas as informações de todas as salas, código da sala, nome da sala e endereço da sala, através dos parâmetros `roocod`, `roonam` e `rooadd`.
- `select_room_by_cod($roocod)`: seleciona as informações de uma sala através do código da mesma. Recebe o código da sala como parâmetro e retorna um vetor com todas as informações da sala selecionada, como código da sala, nome da sala e endereço da sala, através dos parâmetros `roocod`, `roonam` e `rooadd`.

- `update_room($roocod, $roonam, $rooadd)`: atualiza os dados de uma sala. Recebe o código da sala, o nome e endereço da sala como parâmetros e retorna `true` ou `false` dependendo se a atualização foi executada com sucesso ou não.
- `remove_room($roocod)`: remove uma sala do sistema. Recebe o código da sala. Se a sala estiver sendo utilizada em uma reunião, ele retorna uma mensagem que não é possível removê-la. Retorna `true` ou `false` dependendo se a remoção foi executada com sucesso ou não.
- `new_meeting($mtgnam, $mtgvid, $mtgstadat, $mtgenddat, $grpcod, $mtgdes)`: cria uma nova reunião no sistema. Recebe como parâmetros o nome da reunião, o nome do arquivo de vídeo, a data de início da reunião, a data de término da reunião, o código de controle de acesso e a descrição da reunião. Retorna o código da reunião criada (`mtgcod`).
- `select_all_meetings()`: seleciona todas as reuniões do sistema. Não recebe nenhum parâmetro e retorna uma matriz com todas as informações de todas as reuniões, código da reunião, nome da reunião, nome do arquivo de vídeo da reunião, data de início da reunião, data de término da reunião, código de controle de acesso e descrição da reunião, através dos parâmetros `mtgcod`, `mtgvid`, `mtgstadat`, `mtgenddat`, `grpcod` e `mtgdes`.
- `select_all_captured_meetings()`: seleciona todas as reuniões do sistema cujas informações já foram capturadas. Para o sistema, uma reunião encontra-se no *status* de capturada se a data inicial e a data final encontram-se preenchidas, ou seja, se elas estiverem nulas o sistema entende que a reunião ainda não aconteceu. Não recebe nenhum parâmetro e retorna uma matriz com todas as informações das reuniões capturadas, código da reunião, nome da reunião, nome do arquivo de vídeo da reunião, data de início da reunião, data de término da reunião, código de controle de acesso e descrição da reunião, através dos parâmetros `mtgcod`, `mtgvid`, `mtgstadat`, `mtgenddat`, `grpcod` e `mtgdes`.
- `select_captured_meetings_by_user($usrcod)`: seleciona todas as reuniões capturadas para os quais o usuário tem permissão de acesso. Recebe o código do usuário e retorna uma matriz com todas as informações das reuniões capturadas e para os quais o usuário tem permissão, código da reunião, nome da reunião, nome do arquivo de vídeo da reunião, data de início da reunião, data de término da reunião, código de controle de

acesso e descrição da reunião, através dos parâmetros `mtgcod`, `mtgvid`, `mtgstadat`, `mtgenddat`, `grpcod` e `mtgdes`.

- `check_user_permission_in_meeting($usrcod, $mtgcod)`: verifica se um usuário tem permissão de acesso às informações de uma reunião. Recebe o código do usuário e o código da reunião. Retorna `true` ou `false` dependendo das permissões da reunião sobre o usuário.
- `select_meeting_by_cod($mtgcod)`: seleciona as informações de uma reunião do sistema, através de seu código. Recebe o código da reunião e retorna um vetor com todas as informações de reunião selecionada, código da reunião, nome da reunião, nome do arquivo de vídeo da reunião, data de início da reunião, data de término da reunião, código de controle de acesso e descrição da reunião, através dos parâmetros `mtgcod`, `mtgvid`, `mtgstadat`, `mtgenddat`, `grpcod` e `mtgdes`.
- `update_meeting($mtgcod, $mtgnam, $mtgvid, $mtgstadat, $mtgenddat, $grpcod, $mtgdes)`: atualiza os dados de uma reunião. Recebe o código da reunião, o nome da reunião, o nome do arquivo de vídeo da reunião, a data inicial da reunião, a data final da reunião, o código de controle de acesso e a descrição da reunião como parâmetros e retorna `true` ou `false` dependendo se a atualização foi executada com sucesso ou não.
- `remove_meeting($mtgcod)`: remove uma reunião do sistema. Recebe o código da reunião. Diferentemente das outras remoções, a remoção de uma reunião remove todas as informações de *slides*, participantes e *penstrokes* da reunião sem verificação. Isso é feito, pois a reunião é o elemento principal do sistema e se o usuário realmente deseja remover uma reunião, o sistema deve fazê-lo. Retorna `true` ou `false` dependendo se a remoção foi executada com sucesso ou não.
- `new_meeting_room($mtgcod, $roocod, $usrcod)`: cria um relacionamento reunião, sala e usuário. Recebe o código da reunião, o código da sala e o código do usuário como parâmetros. A função faz a verificação de violação de chave primária e retorna `true` ou `false` dependendo se a inserção foi executada com sucesso ou não.
- `select_all_meeting_rooms_by_mtgcod($mtgcod)`: seleciona todos os relacionamentos de salas e usuários em uma reunião, através do código da reunião. Recebe o código da reunião como parâmetro e retorna uma matriz com todas os

relacionamentos dessa reunião, código da reunião, código da sala e código do usuário através dos parâmetros `mtgcod`, `roocod` e `usrcod`.

- `check_user_in_meeting($usrcod, $mtgcod)`: verifica se um usuário participou de uma reunião. Recebe o código do usuário e o código da reunião e retorna `true` ou `false` dependendo do resultado da busca do usuário na reunião.
- `remove_meeting_room($mtgcod, $roocod, $usrcod)`: remove um relacionamento reunião, sala e usuário do sistema. Recebe o código da reunião, o código da sala e o código do usuário como parâmetros. Retorna `true` ou `false` dependendo se a remoção foi executada com sucesso ou não.
- `new_slide($slicod, $mtgcod, $sligif, $slitst)`: cria um novo *slide* de uma reunião no sistema. Recebe como parâmetros o código do *slide*, o código da reunião, o nome da imagem do *slide* e o *timestamp* do *slide*. Retorna `true` ou `false` dependendo se a inserção foi executada com sucesso ou não.
- `new_slide_by_mtgcod($mtgcod, $sligif, $slitst)`: cria um novo *slide* de uma reunião com incremento automático do código do *slide*. Recebe como parâmetros o código da reunião, o nome da imagem do *slide* e o *timestamp* do *slide*. Retorna o código do *slide* inserido (`slicod`) ou `false` dependendo se a inserção foi executada com sucesso ou não.
- `select_slides_by_mtgcod($mtgcod)`: seleciona as informações de todos os *slides* de uma reunião no sistema. Recebe o código da reunião e retorna uma matriz com todas as informações de todos os *slides* da reunião selecionada, código do *slide*, código da reunião, nome da imagem do *slide* e *timestamp* do *slide*, através dos parâmetros `slicod`, `mtgcod`, `sligif` e `slitst`.
- `new_penstroke($pencod, $slicod, $mtgcod, $pengif, $pentst, $pentopx, $pentopy, $penbotx, $penboty)`: cria um novo *penstroke* de uma *slide* de uma reunião no sistema. Recebe como parâmetros o código do *penstroke*, o código do *slide*, o código da reunião, o nome da imagem do *penstroke*, o *timestamp* do *penstroke*, a posição inicial superior, a posição final superior, a posição inicial inferior e a posição final inferior do *penstroke*. Retorna `true` ou `false` dependendo se a inserção foi executada com sucesso ou não.

- `new_penstroke_by_slicod_mtgcod($slicod, $mtgcod, $pengif, $pentst, $pentopx, $pentopy, $penbotx, $penboty)`: cria um novo *penstroke* de um *slide* de uma reunião com incremento automático do código do *penstroke*. Recebe como parâmetros o código do *slide*, o código da reunião, o nome da imagem do *penstroke*, o *timestamp* do *penstroke*, a posição inicial superior, a posição final superior, a posição inicial inferior e a posição final inferior do *penstroke*. Retorna `true` ou `false` dependendo se a inserção foi executada com sucesso ou não.
- `select_penstrokes_by_mtgcod($mtgcod)`: seleciona as informações de todos *penstrokes* dos *slides* de uma reunião no sistema. Recebe o código da reunião e retorna uma matriz com todas as informações de todos os *penstroke* dos *slides* da reunião selecionada, código do *penstroke*, código do *slide*, o código da reunião, o nome da imagem do *penstroke*, o *timestamp* do *penstroke*, a posição inicial superior, a posição final superior, a posição inicial inferior e a posição final inferior do *penstroke*, através dos campos `pencod`, `slicod`, `mtgcod`, `pengif`, `pentst`, `pentopx`, `pentopy`, `penbotx`, `penboty`.

Todas as funções implementadas no módulo de acesso podem ser utilizadas por qualquer outro sistema. Para isso, outros sistemas devem incluir os arquivos mestres do módulo de acesso do *eMeet/SERVE* configurados para utilizar a base de dados MySQL. É possível criar outros arquivos específicos que acessem outras bases de dados existentes, como Oracle, Informix, Postgress e Microsoft SQL Server. Para isso, basta criar um arquivo específico para a base de dados desejada contendo todas as funções descritas e trocar a configuração dos arquivos mestres `header.php` e `footer.php`. Com isso, o *eMeet/SERVE* passará a utilizar outras bases de dados naturalmente, sem que haja reprogramação da aplicação.

5.4.3. Módulo de idiomas do *eMeet/SERVE*

A utilização de múltiplos idiomas foi um dos requisitos iniciais do *eMeet/SERVE*. Por essa razão, o módulo de idiomas segue os padrões da arquitetura proposta. Os idiomas português e inglês foram implementados na versão inicial do *eMeet/SERVE*. A Figura 31 mostra o atual módulo de idiomas do *eMeet/SERVE*.

O funcionamento do módulo de idiomas é bastante simples. Um arquivo principal faz a inclusão do arquivo específico do idioma desejado. Se o *eMeet/SERVE* for configurado para

gerar hiperdocumentos em português, os textos do arquivo de português serão lidos pela camada de idiomas e retornados à aplicação.

Figura 31: Detalhamento do módulo de idiomas e integração com a aplicação

Para estender o sistema, criando um novo idioma, basta criar um arquivo específico para o idioma desejado, com base nos arquivos de inglês e português existentes. É importante que as definições de textos de todos os arquivos sejam as mesmas, variando apenas o valor dessas definições. Desta forma, quando a aplicação precisar do título do hiperdocumento, por exemplo, ela pede ao módulo de idiomas o valor da constante `PAGE_TITLE`. O módulo de idiomas procura o valor dessa constante no arquivo de idioma configurado e retorna o valor da constante à aplicação. A partir daí, a aplicação gera o hiperdocumento cujo título tem o valor retornado pela camada de idiomas.

O arquivo principal, bem como os específicos de cada idioma, encontram-se em:

`emeet/includes/lang/language.php` (arquivo principal)

`emeet/includes/lang/pt.php` (arquivo de português)

`emeet/includes/lang/en.php` (arquivo de inglês)

A definição de textos nos arquivos de idiomas é feita, na linguagem PHP, com o seguinte comando:

```
define(PAGE_TITLE, "Título da página");
```

Todos os arquivos da aplicação devem incluir o arquivo principal de idiomas. Dessa forma, a aplicação pode utilizar diretamente a constante definida, como no exemplo da Figura 32.

```
<html>
<head>
<title>
<?php echo PAGE_TITLE; ?>
</title>
</head>
```

Figura 32: Exemplo de utilização de definições de textos com PHP.

No trecho de código ilustrado na Figura 32, a cadeia de caracteres “Título da página” será exibida como título do hiperdocumento, se o módulo de idiomas estiver configurado para português, por exemplo.

5.4.4. Módulo de *layout* no *eMeet/SERVE*

A mudança de *layout* é uma das alterações mais requisitadas por usuários de hiperdocumentos. Sendo assim, o *eMeet/SERVE* foi construído para que seus hiperdocumentos sejam customizados, sem que haja reprogramação da aplicação.

O módulo de *layout* implementado no *eMeet/SERVE* é muito simples, pois consiste basicamente de um único arquivo que contém as definições necessárias. Apesar de ser um único arquivo, este módulo é muito importante para o sistema, uma vez que ele centraliza todos os aspectos de *layout* dos hiperdocumentos gerados. As definições do módulo de *layout* encontram-se no arquivo:

```
emeet/include/layout/layout.php
```

Os aspectos contemplados pelo módulo de *layout* do *eMeet/SERVE*, que podem ser trocados para realizar mudanças visuais nos hiperdocumentos gerados, são:

- Cor de fundo: a cor de fundo dos hiperdocumentos que serão gerados pode ser trocada. Para isso, usuários devem alterar o valor da definição `COLOR_BACK` do arquivo `layout.php`.
- Cores de fontes e ligações: os textos e ligações dos hiperdocumentos possuem diversas cores. Todas elas podem ser trocadas, alterando-se as definições `COLOR_1`, `COLOR_2`, `COLOR_3`, `COLOR_4` do arquivo `layout.php`.
- Tamanho das fontes: os tamanhos das fontes dos textos e das ligações podem ser alterados. Para isso, usuários devem trocar as definições `SIZE_1`, `SIZE_2` e `SIZE_3` do arquivo `layout.php`.
- Tipo das fontes: os tipos de fontes também podem ser customizadas através das definições `FACE_1` e `FACE_2` do arquivo `layout.php`.
- Cores de indicadores de ação: ao inserir um usuário, por exemplo, o sistema informa que “O usuário foi inserido com sucesso”. Tais mensagens, chamadas de

mensagens de ação, também podem ter suas cores alteradas. Para isso, deve-se alterar a definição `COLOR_MESSAGE` do arquivo `layout.php`.

- Textos de ajuda instantânea: os textos de ajuda on-line, que são apresentados em uma tabela quando usuários passam o mouse sobre o ícone '?', em diversos hiperdocumentos do *eMeet/SERVE* também podem ser alterados através das definições `COLOR_HELP_BORDER`, `COLOR_HELP_BACKGROUND`, `COLOR_HELP_TEXT` e `WIDTH_HELP_DIV` do arquivo `layout.php`.
- Tamanho do menu lateral esquerdo: o tamanho do menu lateral esquerdo também pode ser customizado, através da definição `LEFT_MENU_SIZE` do arquivo `layout.php`.
- Tamanho da área central: assim como o menu lateral esquerdo, o tamanho da área central dos hiperdocumentos também pode ser customizado, através da definição `CENTER_TABLE_SIZE` do arquivo `layout.php`.
- Tamanho da caixa de *login*: a caixa de *login*, que fica abaixo do menu lateral esquerdo, também pode ter o seu tamanho customizado. Para isso, é necessário alterar o valor da definição `LOGIN_TABLE_SIZE` do arquivo `layout.php`.
- Tamanho da tabela de vídeo: durante a apresentação das informações das reuniões, o vídeo é exibido dentro de uma tabela. O tamanho dessa tabela pode ser customizado através da definição `VIDEO_TABLE_SIZE` do arquivo `layout.php`.
- Tamanho dos *slides*: durante a apresentação das informações das reuniões, o tamanho dos *slides* apresentados pode ser customizado através dos defines `SLIDE_TABLE_SIZE` e `SLIDE_TABLE_SIZE_HEIGHT` do arquivo `layout.php`. Com isso, pode-se definir a largura e a altura dos *slides*, respectivamente.
- Posição vertical dos *slides* e da linha do tempo: ainda na apresentação das informações das reuniões, a posição vertical (altura) dos *slides* e da linha do tempo podem ser customizadas, através das definições `SLIDE_TOP_POSITION` e `TIME_LINE_TOP_POSITION` do arquivo `layout.php`.
- Tamanho da linha do tempo: a linha do tempo pode ter seu tamanho horizontal configurável. Para alterá-lo, usuários devem informar as posições (*pixels*) horizontais do início, um terço, dois terços e final da linha do tempo. Isso é feito através das

definições `TIME_LINE_START`, `TIME_LINE_1_3`, `TIME_LINE_2_3` e `TIME_LINE_END` do arquivo `layout.php`.

Os hiperdocumentos gerados pelo *eMeet/SERVE* têm todas essas definições de *layout* configuradas em um único arquivo. Sendo assim, não há necessidade de reprogramação da aplicação para trocar as principais características de *layout* dos hiperdocumentos gerados. Isso garante um menor esforço por parte dos desenvolvedores durante a fase de manutenção do *eMeet/SERVE*, uma vez que a troca parcial do *layout* dos hiperdocumentos a serem gerados pode ser realizada por pessoas/usuários leigos em programação.

Caso haja necessidade de troca de outros parâmetros de *layout* não especificados no módulo de *layout*, será necessário que desenvolvedores alterem a aplicação.

5.4.5. A aplicação *eMeet/SERVE*

A aplicação *eMeet/SERVE* é a responsável por gerar os hiperdocumentos a partir das informações e definições armazenadas por outros módulos.

As configurações genéricas do *eMeet/SERVE* estão em um arquivo a parte (`config.php`), assim como as definições de *layout* do módulo de *layout*. Tais configurações são parâmetros que necessitam de mudanças durante ou após a instalação da aplicação, como, por exemplo, a especificação do domínio do *site*. O arquivo que contém as configurações do *eMeet/SERVE* está localizado em:

```
emeet/include/config/config.php
```

Os parâmetros configuráveis do *eMeet/SERVE* são:

- Domínio do *site*: contém o nome do domínio em que o *eMeet/SERVE* está ou será instalado. Exemplo: `www.emeet.com`. Para alterar o domínio do *site*, usuários devem modificar a definição `SITE_DOMAIN` do arquivo `config.php`.
- Diretório de imagens do sistema: é possível modificar o local (diretório) onde as imagens do *eMeet/SERVE* estão armazenadas. Para isso, deve-se modificar a definição `IMAGE_PATH` do arquivo `config.php`. A especificação do diretório é feita em relação ao domínio do *site*.
- Diretório de vídeos: assim como o diretório de imagens, o local onde estarão armazenados os vídeos capturados das reuniões pode ser configurado. Isso é feito

através da definição `VIDEO_PATH` do arquivo `config.php`. A especificação do diretório é feita em relação ao domínio do *site*.

- Diretório de *slides*: assim como o diretório de vídeos, o local onde estarão armazenados os *slides* capturados das reuniões pode ser configurado. A definição `SLIDE_PATH` do arquivo `config.php` é a responsável por essa configuração. A especificação do diretório é feita em relação ao domínio do *site*.
- Diretório do hiperdocumento inicial do *site*: para configurar o diretório relativo do hiperdocumento inicial do *site*, deve-se alterar o valor da definição `HOME_PATH` do arquivo `config.php`. Assim como os outros diretórios, a especificação do diretório é feita em relação ao domínio do *site* e conter o arquivo `index.php`, que é o hiperdocumento inicial do *site*.
- *E-mail* de contato dos autores/mantenedores: no *eMeet/SERVE* pode-se enviar mensagens aos autores e mantenedores do sistema. A definição `AUTHORS_EMAIL` do arquivo `config.php` é a responsável por conter os *e-mails* dos autores. É para esses *e-mails* que o sistema enviará as mensagens de contato, da Seção Fale Conosco do *eMeet/SERVE*.
- Código do administrador: no *eMeet/SERVE* existe um administrador geral do sistema, que tem acesso a todas as seções do sistema, cujo código deve estar indicado na definição `ADMINISTRATOR_COD` do arquivo `config.php`. O código do administrador é o identificador de um usuário do servidor de usuários.
- Tempo de atualização: durante a apresentação dos dados de uma reunião, há um intervalo de tempo de sincronização da linha do tempo com a exibição do vídeo (maiores detalhes na Seção 6.4 deste documento). Esse intervalo é configurado através da definição `SECONDS_DELAY` do arquivo `config.php`. O valor contido nesta definição deve ser em milissegundos.

Todas configurações existentes são fundamentais para a instalação e manutenção do *eMeet/SERVE*. Devido à existência desse arquivo de configuração, os responsáveis pelo sistema podem alterar tais configurações a qualquer momento, sem que haja necessidade de reprogramação da aplicação. Além disso, o sistema torna-se muito mais flexível e de fácil instalação/manutenção.

Além do arquivo de configuração, a aplicação é composta por diversos arquivos organizados de forma a facilitar a criação de novos hiperdocumentos. Os arquivos da aplicação estão divididos entre arquivos geradores e arquivos de tratamento.

O funcionamento dos arquivos geradores e dos arquivos de tratamento da aplicação, explicados a seguir, está de acordo com o funcionamento básico das requisições e chamadas de um sistema Web em geral, e de um sistema que utiliza a linguagem PHP para a geração dinâmica de hiperdocumentos, em particular.

A Figura 33 ilustra como funciona basicamente uma requisição de um usuário ao servidor Web, que utiliza a linguagem PHP. O usuário, utilizando um navegador, realiza a requisição de um arquivo, que pode ser PHP ou não, ao servidor Web. O servidor Web, internamente, verifica se o arquivo requerido é válido e, se for, o executa. A saída do arquivo após execução é enviada de volta ao cliente. Normalmente, tal saída é um hiperdocumento. Porém, pode ser necessário que o arquivo original realize chamadas a outros arquivos internos do servidor, antes de gerar uma saída ao usuário. Portanto, nem sempre a saída recebida pelo cliente é a saída da execução do arquivo requisitado. Esse processamento é feito de forma transparente ao usuário.

Figura 33: Funcionamento básico de requisição ao servidor Web com a linguagem PHP.

Apesar de utilizar a linguagem PHP, o sistema de requisição explicado na Figura 33 é o mesmo com outras linguagens como JSP, especificada pela Sun e ASP, da Microsoft.

5.4.5.1. Arquivos Geradores

Arquivos geradores são arquivos que geram hiperdocumentos para usuários do *eMeet/SERVE*, ou seja, geram ao menos, uma saída ao usuário. Naturalmente, arquivos geradores geram como saída um hiperdocumento, composto normalmente de código HTML.

Existe um arquivo gerador da aplicação para cada família de hiperdocumentos³⁵ a ser gerada. Isso significa que um mesmo arquivo gerador da aplicação contém códigos base para a geração de múltiplos hiperdocumentos de uma mesma família. Por exemplo, os hiperdocumentos de apresentação das informações das reuniões são gerados a partir de um único arquivo gerador da aplicação. O arquivo gerador recebe, como parâmetro, o código da reunião. Nesse exemplo, o hiperdocumento gerado varia de acordo com o código da reunião passado ao arquivo gerador, conforme mostra a Figura 34. Logo, os hiperdocumentos gerados 1, 2 e 3 possuem a mesma estrutura, porém dados e informações diferentes, de acordo com o código da reunião associado.

Figura 34: Exemplo de um arquivo gerador e seus hiperdocumentos gerados. O arquivo gerador recebe o código da reunião e gera hiperdocumentos de mesma família.

Apesar de existirem arquivos geradores completamente diferentes, dependendo dos hiperdocumentos que se deseja gerar, cada arquivo gerador da aplicação deve conter uma estrutura básica, indicada na Figura 35.

```

<?php
//No início do arquivo
include_once("emeet/db/header.php");
include_once("emeet/layout/body.php");
?>
...
... // Código a ser gerado
<?php
// No final do arquivo
include_once("emeet/layout/close_body.php");
include_once("emeet/db/footer.php");
?>
  
```

Figura 35: Estrutura básica de um arquivo gerador do *eMeet/SERVE*.

³⁵ Neste documento, o termo família de hiperdocumento é utilizado para representar hiperdocumentos que foram gerados a partir de um mesmo arquivo gerador, ou seja, são hiperdocumentos muito semelhantes que tem apenas seus conteúdos variáveis em relação aos outros.

A estrutura básica, ilustrada na Figura 35, visa garantir que cada módulo da arquitetura esteja sendo incluído de maneira correta na aplicação e que a criação de novos hiperdocumentos, e conseqüentemente de novos arquivos geradores, deve ser realizada de forma correspondente com o restante da aplicação.

Segundo a Figura 35, todos os arquivos geradores do *eMeet/SERVE* devem conter os dois primeiros includes (`header.php` e `body.php`) no início do arquivo e os dois últimos includes (`close_body.php` e `footer.php`) no final.

Esses quatro arquivos, que estão presentes em todos os arquivos geradores de hiperdocumentos da aplicação do *eMeet/SERVE*, garantem que todos os módulos da arquitetura estão sendo incluídos. Cada arquivo é explicado a seguir:

- `header.php`: realiza a inclusão do módulo de acesso no arquivo. É o arquivo responsável por incluir arquivos específicos de acesso à base de dados e de utilização dos serviços do servidor de usuário. Esse arquivo está localizado no diretório `emeet/include/db/`.
- `footer.php`: fecha as conexões abertas pelo arquivo `header.php`. Assim como o `header.php`, esse arquivo está localizado em `emeet/include/db/`.
- `body.php`: realiza a inclusão do módulo de idiomas e do módulo de *layout*. Somente arquivos geradores utilizam os módulos de idiomas e de *layout*, pois somente eles geram hiperdocumentos. Por esse motivo, esse arquivo está localizado no diretório `emeet/include/layout`. Além disso, o `body.php` faz a inclusão da parte comum de *layout* de todos os hiperdocumentos. A Figura 36 mostra um hiperdocumento gerado pelo *eMeet/SERVE*. A barra superior com o logo, o menu lateral esquerdo e a caixa de *login* (quando um usuário realiza o *login*, a caixa de *login* é substituída por uma saudação ao usuário), destacados na Figura 36, são elementos comuns de *layout* a todos os hiperdocumentos. O código responsável pela geração destes elementos também está presente no arquivo `body.php`.
- `close_body.php`: arquivo responsável por complementar o arquivo `body.php`, a fim de fechar as tabelas e outros elementos de hiperdocumentos que foram abertos pelo arquivo `body.php`.

Figura 36: Hiperdocumento gerado pelo *eMeet/SERVE* com parte comum destacada. A parte comum representa a barra superior, o logo do *eMeet*, o menu lateral esquerdo e a caixa/indicação de *login*.

Arquivos geradores normalmente fazem chamada ao módulo de acesso somente para recuperar informações. É recomendável que arquivos gerados não realizem nenhuma ação efetiva no sistema, como, por exemplo, cadastrar uma nova sala. Ações como cadastro de salas devem ser feitas por arquivos de tratamento, como explicado a seguir.

5.4.5.2. Arquivos de Tratamento

Arquivos de tratamento são arquivos internos da aplicação que não geram saída aos usuários, ou seja, eles são transparentes aos usuários.

Tais arquivos são utilizados para realizar ações efetivas no sistema, como, por exemplo, criar uma nova reunião. Esse procedimento está ilustrado na Figura 37. Para isso, o arquivo de tratamento (A) recebe os dados do formulário de criação de uma nova reunião (de um hiperdocumento criado por um arquivo gerador), realiza o tratamento adequado, (B) ativa o módulo de acesso para gravar os dados dessa nova reunião e, dependendo do resultado, (C) redireciona o usuário para o hiperdocumento adequado (fazendo uma chamada a um arquivo gerador).

Arquivos de tratamento podem chamar um ou mais arquivos geradores (que apresentam o hiperdocumento gerado como saída aos usuários). Além disso, arquivos de tratamento normalmente são chamados a partir de arquivos geradores. Como dito anteriormente, arquivos de tratamento são transparentes aos usuários finais do *eMeet/SERVE*, uma vez que eles não geram nenhuma saída. Porém, recomenda-se que, para cada ação efetiva do sistema, como inserções,

edições e remoções das informações contidas no sistema, exista pelo menos um arquivo de tratamento que realize tal ação. Com isso, há a separação lógica das ações do sistema, com as ações de geração de hiperdocumentos, facilitando a programação e, principalmente, a manutenção do sistema.

Figura 37: Exemplo de utilização de um arquivo de tratamento. (A) envio de dados do formulário, (B) ativação do módulo de acesso e (C) redirecionamento para o hiperdocumento adequado através de um arquivo gerador.

O tratamento de erros no *eMeet/SERVE* é feito através de um único arquivo gerador. Quando um arquivo de tratamento ou gerador verifica que há erros no sistema, como, por exemplo, o não preenchimento de um campo obrigatório ou a falta de parâmetros no sistema, o usuário é redirecionado a um arquivo de erros, que é o responsável por exibir o erro adequadamente ao usuário. O arquivo de erros deve receber como parâmetro o código do erro e exibir o texto referente ao erro recebido ao usuário. O arquivo de erros encontra-se em `/emeet/http/error/error.php`. Na maioria dos casos, o arquivo de erros é chamado por arquivos de tratamento, pois a maioria das ações lógicas no sistema, sujeitas a um maior número de erros, são realizadas por este tipo de arquivo.

Assim como arquivos geradores, os arquivos de tratamento também possuem uma estrutura básica, conforme ilustrada na Figura 38. Caso seja necessário criar um novo arquivo de tratamento, essa estrutura básica deve ser seguida, garantindo que tal arquivo esteja inserido dentro do contexto da arquitetura proposta.

A diferença da estrutura básica de um arquivo de tratamento para um arquivo gerador é que, no arquivo de tratamento não estão presentes os arquivos responsáveis pela inclusão dos módulos de *layout* e de idiomas (`body.php` e `close_body.php`). Os demais arquivos da estrutura básica são os mesmos da estrutura dos arquivos geradores (`header.php` e `footer.php`), responsáveis em incluir o módulo de acesso. Por não gerar saída aos usuários,

nem hiperdocumentos, arquivos de tratamento não devem incluir o módulo de *layout* e o módulo de idiomas.

```
<?php
//No início do arquivo
include_once("emeet/db/header.php");
?>

...
... // Código de tratamento
<?php
// No final do arquivo
include_once("emeet/db/footer.php");
?>
```

Figura 38: Estrutura básica de um arquivo de tratamento do *eMeet/SERVE*.

Portanto, a aplicação do *eMeet/SERVE* é composta de um arquivo de configuração, além de uma série de arquivos geradores e de tratamento. Além disso, a aplicação contém arquivos de auxílio, como o `body.php` e o `close_body.php`, que realizam a inclusão de módulos e partes comuns do sistema nos arquivos da aplicação.

5.5. Considerações finais

A modelagem e análise de requisitos do *eMeet/SERVE* foram realizadas através do uso de parte da especificação UML (Rosenberg & Scott, 1999). A partir dos estudos de caso realizados, quatro seções independentes foram definidas para o *eMeet/SERVE*. As quatro seções são: Seção Reuniões, Seção Salas, Seção Usuários e Seção Fale Conosco. As ligações para tais seções e para o hiperdocumento inicial estão sempre visíveis no menu lateral esquerdo dos hiperdocumentos gerados pelo *eMeet/SERVE*.

A arquitetura LAIAc, proposta neste trabalho, pode ser utilizada por qualquer sistema de geração de hiperdocumentos, uma vez que seus módulos podem ser implementados em qualquer linguagem de programação desejada. Portanto, a arquitetura LAIAc é independente de linguagem de *scripts server-side*.

A utilização da arquitetura LAIAc pelo *eMeet/SERVE* mostra uma instância real dos conceitos propostos por esta arquitetura. O *eMeet/SERVE* obteve uma série de vantagens por utilizar a arquitetura LAIAc, como independência de base de dados e dispositivos de armazenamento, suporte a múltiplos idiomas, customização do *layout* dos hiperdocumentos gerados, integração com reutilização de serviços por outras aplicações e preparação para instalação, manutenção e expansão do sistema.

6.1. Considerações iniciais

Hiperdocumentos devem possuir uma série de características para uma melhor interação com os usuários, tais como usabilidade, consistência e facilidade de navegação.

Nielsen (1993) define usabilidade como a qualidade de interação entre usuários e sistemas de software. A consistência é um dos mais importantes princípios de usabilidade que deve ser considerado no projeto de interfaces (Dix et al., 1998). Já a facilidade de navegação foi especificada por Rocha and Baranauskas (2000), que discutem práticas de usabilidade no contexto da World Wide Web.

Netto (2002) realizou um levantamento sobre regras de usabilidade em hiperdocumentos, e propôs um conjunto de regras, formadas por heurísticas e *guidelines*, destinadas à apresentação multimídia em hiperdocumentos. Essas regras correspondem a uma especialização das heurísticas de Nielsen (1993, 1995, 1997).

A interface visual dos hiperdocumentos gerados pelo *eMeet/SERVE* foi desenvolvida para respeitar ao máximo as regras fornecidas por Netto (2002), que são compatíveis com outros fatores de usabilidade propostos por outros autores. O autor deste documento possui experiências no projeto e na implementação de interfaces, sendo que alguns aspectos desenvolvidos nas interfaces dos hiperdocumentos gerados foram decorrentes dessa experiência.

Os hiperdocumentos responsáveis por apresentar as informações capturadas das reuniões ainda contam com um mecanismo de controle de acesso onde é permitido, de acordo com a necessidade, configurar o nível de segurança dessas informações. Dessa forma, usuários distintos podem ter acesso a informações distintas das reuniões capturadas.

Por se tratar de um sistema multimídia, o *eMeet/SERVE* possui aspectos de sincronização entre o áudio/vídeo, *slides* e a linha do tempo. Tais aspectos de sincronização, bem como a

organização visual, o controle de acesso e a forma de interação entre os usuários e os hiperdocumentos gerados são discutidos a seguir.

6.2. Estrutura e Conteúdo dos Hiperdocumentos

Os hiperdocumentos gerados pelo *eMeet/SERVE* apresentam interfaces desenvolvidas para melhorar a qualidade de interação entre os mesmos e os usuários.

As regras, formadas de heurísticas e *guidelines*, propostas por Netto (2002) serviram como base para o desenvolvimento das interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE*.

De acordo com as Figuras 39, 40 e 41, que ilustram interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE*, pode-se observar que existem padrões de apresentação relativas à estrutura e ao conteúdo dos hiperdocumentos. Esses padrões ilustram a utilização de regras de usabilidade, descritos a seguir:

- a. O logo do *eMeet* está sempre visível na parte superior esquerda de todos os hiperdocumentos gerados. O logo serve como uma âncora que leva os usuários do *eMeet/SERVE* para o hiperdocumento inicial. Isso caracteriza uma padronização visual e facilita a navegação e identificação dos hiperdocumentos;

Figura 39: Hiperdocumento gerado: (a) logo do *eMeet*, (b) menu lateral esquerdo, (c) seção correspondente, (d) botão voltar, (e) instruções aos usuários, (f) opções de ajuda, (g) botões de ação e (h) padrão de cores, fontes e estilos.

- b. O menu das seções do *site* (Início, Resumos, Salas, Usuários e Fale Conosco) está sempre visível na parte esquerda de todos os hiperdocumentos gerados. Cada seção tem seu nome e serve como âncora para levar o usuário à seção desejada. Além de facilitar a navegação através de ligações rápidas para qualquer uma das seções, a presença do menu em todos os hiperdocumentos na mesma posição melhora a familiaridade, padronização e consistência dos hiperdocumentos;
- c. Em todos os hiperdocumentos há a indicação da seção corrente ao usuário. Essa indicação faz com que os usuários localizem rapidamente em qual seção estejam navegando. A posição dessa informação é a mesma para todos os hiperdocumentos gerados, tendo por objetivo atender aos princípios de familiaridade, padronização e consistência (Dix et al., 1998) dos hiperdocumentos gerados;
- d. Em praticamente todos os hiperdocumentos gerados existe a opção de voltar, representada por um botão vermelho, seguido da indicação “voltar”. O botão de voltar normalmente está localizado abaixo da tabela de ação central. Âncoras que permitem o retorno da navegação devem ser fornecidas (Netto, 2002), pois muitos desconhecem ou não utilizam a função de voltar do próprio navegador;
- e. Instruções sobre o que o usuário pode realizar no hiperdocumento devem sempre estar presentes. As instruções devem ser simples, claras e objetivas, com o intuito de auxiliar a utilização dos hiperdocumentos por parte dos usuários;
- f. Botões de ajuda devem estar presentes para o caso do usuário necessitar de informações adicionais que não estão visíveis nos hiperdocumentos. Tais ajudas auxiliam principalmente os usuários menos experientes dos hiperdocumentos. As ajudas implementadas nos hiperdocumentos gerados pelo *eMeet/SERVE* permitem que o usuário posicione o mouse sobre as indicação de ajuda e um texto com explicações é exibido. Maiores informações encontram-se no Apêndice A – Manual do Usuário do *eMeet/SERVE*;
- g. Botões de ação indicam lugares onde usuários podem realizar alguma ação. Tais botões possuem uma imagem e a indicação da ação. Esses botões são padronizados e atendem aos princípios de familiaridade e de consistência em relação aos hiperdocumentos gerados;
- h. As cores, fontes e estilos padronizados utilizados na interface dos hiperdocumentos gerados atendem aos princípios de familiaridade, consistência e usabilidade;

- i. Enquanto os usuários estão autenticados no *eMeet/SERVE*, há uma saudação e uma indicação de qual usuário realizou o *login*. Esta informação auxilia na segurança dos hiperdocumentos, uma vez que tal informação apresenta um tratamento customizado e familiar para os usuários;

Figura 40: Hiperdocumento gerado: (f) opções de ajuda, (i) saudação, (j) campos obrigatórios, (k) funcionalidade principal do hiperdocumento, (l) campos de formulários e (m) ordem dos botões de ação.

- j. Indicação de campos obrigatórios estão presentes nos locais onde os usuários devem preencher ou selecionar determinados campos de formulários. O significado da indicação (*) deve estar presente em um local visível do hiperdocumento;
- k. Logo abaixo da indicação da seção corrente encontra-se a indicação da funcionalidade do hiperdocumento. No exemplo da Figura 40, o hiperdocumento tem a finalidade de criar uma nova sala no *eMeet*. No caso da Figura 41, a finalidade do hiperdocumento é exibir as informações de uma reunião aos usuários. Neste caso, deve existir a indicação de qual reunião será exibida. É recomendável que hiperdocumentos tenham somente uma finalidade específica, a fim de não confundir sua utilização, e que tal funcionalidade esteja clara para os usuários. Hiperdocumentos que realizam muitas funcionalidades geralmente possuem muitos botões e âncoras. Estes podem deixar os usuários indecisos sobre o que fazer no hiperdocumento;
- l. Os campos dos formulários devem estar alinhados e conter uma breve descrição sobre o que deverá ser preenchido naquele campo. Isso melhora a organização e usabilidade dos hiperdocumentos que apresentam formulários;

- m. A ordem dos botões de ação é um outro fator importante quando o hiperdocumento possui um formulário. Recomenda-se que a ação principal (por exemplo, envio de um formulário) esteja sempre na posição mais à direita. Isso é importante, pois muitas vezes usuários agem por instinto sem ler a indicação do botão (Morkes & Nielsen, 1997; Nielsen, 1995). Além disso, é sempre recomendável que existam botões que limpem formulários a serem preenchidos;

Figura 41: Hiperdocumento gerado: (d) botão voltar, (k) funcionalidade principal do hiperdocumento, (n) controle de vídeo, (o) controle de slides, (p) tempo de apresentação, (q) slide atual, (r) linha do tempo, (s) troca de slide e (t) não utilização de frames e barras de rolagem horizontais.

- n. Deve haver opções para iniciar ou parar a reprodução de mídias contínuas, como vídeo e áudio. Usuários devem ter opções de realizar as operações básicas sobre qualquer tipo de mídia (Netto, 2002);
- o. Assim como no item “n”, deve haver opção para troca manual de slides e elementos desse tipo. As âncoras devem ser bem representadas a fim de não causar dúvidas aos usuários. Isto facilita a navegação e usabilidade dos hiperdocumentos;

- p. No caso de indicação de tempo de uma apresentação, deve haver indicações que auxiliem o usuário a saber o tempo total da mesma, assim como o tempo atual da apresentação. Além disso, tais indicações podem servir como âncoras que auxiliem a navegação e utilização dos hiperdocumentos pelos usuários;
- q. A indicação do *slide* atual, no caso da Figura 41, ou de qualquer elemento similar é uma informação importante para o usuário;
- r. A linha de tempo indica as trocas de *slides* e o tempo atual através de barras dispostas ao longo da linha do tempo (mais informações encontram-se na Seção 6.4.). Hiperdocumentos, como o da Figura 41, devem informar aos usuários o máximo possível de detalhes, como o número de *slides*, o tempo de cada *slide*, dentre outros;
- s. Além de informar quando ocorrerá a troca de *slides* de acordo com a posição na linha do tempo, a barra serve como âncora para facilitar a navegação entre *slides*, caso o usuário queira saltar para um *slide* específico;
- t. Barras de rolagem horizontais e *frames* não foram ser utilizados. Barras de rolagem verticais foram evitadas. Esses fatores respeitam os princípios de usabilidade de apresentação (Nielsen, 1997).

O conteúdo dos hiperdocumentos gerados pode ser observado nas Figuras 39, 40 e 41. Existe uma parte comum a todos hiperdocumentos gerados, que inclui barra superior, logotipo, menu lateral esquerdo e informações de *login*. Além disso, existe uma parte reservada para a apresentação de conteúdo, localizado no centro dos hiperdocumentos. É nessa parte central dos hiperdocumentos que encontram-se as informações mais importantes.

Todos os hiperdocumentos gerados pelo *eMeet/SERVE* respeitam os padrões listados anteriormente. O modelo de interface final nasceu após uma série de outras tentativas que desrespeitavam muitas regras de usabilidade. A interface utilizada no *eMeet/SERVE* é um primeiro passo para a geração de hiperdocumentos de boa usabilidade e navegabilidade.

Apesar de se basear nas regras de Netto (2002), além de outros autores de usabilidade, como Nielsen (1993, 1995, 1997), Dix et al. (1998) e Rocha e Baranauskas (2000), nenhuma avaliação científica foi feita nas interfaces dos hiperdocumentos gerados. Um dos trabalhos futuros é fazer uma avaliação dessas interfaces. Com isso, pode-se estudar melhores formas de representar a estrutura e o conteúdo das interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE*.

6.3. Interação

A interação entre os usuários do *eMeet/SERVE* e os hiperdocumentos gerados deve ocorrer da forma mais natural e familiar possível. Os hiperdocumentos gerados pelo *eMeet/SERVE* possuem uma série de características implementadas para melhorar a interação usuário-computador.

Alguns pontos que já foram apresentados na seção anterior, mais outras características importantes do *eMeet/SERVE* referentes à interação, são destacados a seguir. Tais características são referentes para qualquer hiperdocumento Web.

6.3.1. Botões de ação

Ao realizar ações no *eMeet/SERVE*, usuários interagem com hiperdocumentos. Muitas vezes, as representações visuais da ação são feitas de maneiras diferentes em um mesmo hiperdocumento. Esse fato pode confundir o usuário e acabar diminuindo a usabilidade e a utilidade dos hiperdocumentos.

Para amenizar esse problema, um padrão para representar visualmente as possíveis ações do usuário foi criado. A indicação de ações que o usuário pode disparar nos hiperdocumentos é composta de uma imagem de um botão, seguido de uma indicação textual, como apresentada na Figura 42.

Figura 42: Exemplo de botões visuais, seguidos de indicações textuais, utilizados nos hiperdocumentos gerados pelo *eMeet/SERVE* para representar ações.

Além dos usuários saberem que as ações são representadas por botões visuais seguidos de indicações textuais, outros fatores podem facilitar a interação entre hiperdocumentos e usuários.

Um dos fatores é a ordem de exibição dos botões. No *eMeet/SERVE*, os botões que realizam a ação principal do hiperdocumento, como “Criar”, “Atualizar” e “Remover” estão sempre localizados em uma posição mais à direita do que os demais. Muitas vezes, os usuários agem por instinto e não lêem a indicação textual dos botões (Morkes & Nielsen, 1997; Nielsen, 1995).

Outro fator é a consistência da cor dos botões. No *eMeet/SERVE*, os botões de voltar são vermelhos, enquanto os botões de outras ações são azuis. Isso auxilia os usuários a encontrarem um determinado botão rapidamente³⁶.

³⁶ A experiência profissional do autor deste documento comprova que botões de diferentes cores e formatos auxiliam os usuários na operação de hiperdocumentos Web.

Um fator que não foi utilizado no *eMeet/SERVE*, mas que também serviria para auxiliar os usuários, seria utilizar botões de formatos diferentes. No *eMeet/SERVE*, todos os botões são ovais. Porém, outros sistemas podem variar o formato dos botões, como triângulos, quadrados ou outras representações gráficas, a fim de relacionar a determinadas ações³⁶.

Em hiperdocumentos que possuam formulários, o número de botões existentes é um fator que pode confundir os usuários. Nesses hiperdocumentos, o autor deste documento recomenda que existam, no máximo, quatro botões³⁷, sendo que um deles é o botão voltar, que deve estar sempre presente.

6.3.2. Indicador de ações

Ao realizar uma ação no sistema, utilizando os botões de ações, os usuários devem ser informados se as operações solicitadas foram de fato realizadas ou não. Para isso, foram implementados no *eMeet/SERVE* os indicadores de ações. Tais indicadores, como o da Figura 43, têm a finalidade de manter os usuários informados sobre o *status* das operações solicitadas no sistema.

Figura 43: Exemplo de indicador de ações nos hiperdocumentos gerados.

No *eMeet/SERVE*, qualquer ação realizada pelos usuários, como criar, editar, selecionar ou remover um item, são seguidas de uma indicação de sucesso ou não, sobre o *status* da ação

³⁷ O autor deste documento possui experiência profissional com usuários de hiperdocumentos. Um número maior de quatro botões em hiperdocumentos que tenham formulários geralmente causa dúvidas nos usuários desses hiperdocumentos. Portanto, recomenda-se que existam, no máximo, quatro botões.

desejada. Tais indicações garantem que as operações solicitadas pelos usuários foram realmente executadas pelo sistema.

Essas indicações aparecem em cor vermelha nos hiperdocumentos gerados pelo *eMeet/SERVE*, para serem destacadas aos usuários.

6.3.3. Instruções e Informações de Ajuda

Instruções e informações de ajuda presentes nos hiperdocumentos são outros fatores importantes na interação usuário-computador. (Netto, 2002; Nielsen, 1993)

Instruções são textos simples e objetivos que são exibidos aos usuários pelos hiperdocumentos. Já informações de ajuda são explicações mais detalhadas que podem ser exibidas aos usuários pelos hiperdocumentos, caso o usuário as solicite.

Nos hiperdocumentos gerados pelo *eMeet/SERVE*, as instruções, localizadas na parte superior, são textos que explicitam qual a ação principal que o usuário poderá realizar no hiperdocumento. No caso da Figura 40, a instrução é “Preencha corretamente os campos abaixo” que, juntamente com a indicação da finalidade do hiperdocumento “Criar uma nova reunião”, resume aos usuários o que eles poderão realizar no hiperdocumento de forma objetiva.

Além das instruções que estão visíveis nos hiperdocumentos, usuários poderão obter ajuda adicional se desejarem, que é um texto explicativo mais detalhado, deixando o cursor do mouse sobre o ícone de ajuda, representada pela Figura 44.

Figura 44: Ícone de ajuda e um exemplo de ajuda adicional.

Os textos com informações de ajuda já estão presentes nos hiperdocumentos, porém só são exibidos quando o usuário deixar o cursor do mouse sobre a indicação . Isso é implementado através da tag `<DIV>` do HTML, que simula camadas visuais nos hiperdocumentos (mais detalhes sobre as camadas do HTML são apresentados na Seção 6.4).

6.3.4. Tratamento de Erros

Erros de execução do sistema não devem ser apresentados aos usuários. Portanto, todas as operações críticas, como, por exemplo, conexão com o banco de dados, até operações de caráter navegacional, como preenchimento de campos obrigatórios em um formulário, devem ser tratadas a fim de evitar que erros do sistema sejam exibidos diretamente aos usuários. Se, neste

tratamento, o sistema verificar um erro, ele deve formatá-lo adequadamente e, dessa maneira, exibir aos usuários.

Além disso, as mensagens de erro são muito importantes para os usuários. A mensagem deve ser clara e sucinta, indicando qual erro ocorreu e o qual ação o usuário poderá realizar para contornar ou solucionar o problema.

Quando um erro ocorre na utilização dos hiperdocumentos, usuários são redirecionados para um hiperdocumento responsável por exibir a mensagem de todos os erros possíveis, conforme mostra a Figura 37.

O arquivo gerador (mais detalhes na Seção 5.4.5.1.) responsável pelo tratamento de erros recebe como parâmetro o código do erro. De acordo com esse código, o arquivo gerador recupera a mensagem apropriada do módulo de idiomas e gera o hiperdocumento de erro, que é apresentado aos usuários. A Figura 45 mostra um exemplo de um hiperdocumento gerado com uma mensagem de erro exibida aos usuários. O arquivo gerador responsável pelo tratamento de erros está localizado no diretório:

`/emeet/http/erro/erro.php`

Figura 45: Exemplo de hiperdocumento com mensagem de erro aos usuários.

Existem seis tipos de erros que são tratados pelo *eMeet/SERVE*, descritos a seguir:

1. Falha na conexão do *eMeet/SERVE* com o gerenciador de base de dados;
2. Campos obrigatórios de um formulário não preenchidos pelos usuários;

3. Falta de parâmetros para visualização de hiperdocumentos. Isso pode ocorrer se usuários tentarem simular uma passagem de parâmetros pelo próprio endereço³⁸;
4. Remoção de itens na base de dados não permitidos devido à violação de chaves estrangeiras;
5. Erro no cadastro de senhas dos usuários, devido a não igualdade das senhas digitadas;
6. Falta de permissão dos usuários para acessar determinados hiperdocumentos.

Outros erros podem ser adicionados ao sistema. Para isso, é necessário realizar alterações simples no arquivo de erro (`erro.php`) e adicionar os textos das mensagens de erro no arquivo de idioma desejado, no módulo de idiomas.

6.4. Tratamento de sincronização

Alguns hiperdocumentos gerados pelo *eMeet/SERVE* são multimídia, pois eles contêm informações em áudio/vídeo e uma sequência de *slides*. Essa combinação de elementos de mídia discreta (imagem dos *slides*) e de mídia contínua permite classificá-los como hiperdocumentos multimídia (Steinmetz & Nahrstedt, 1995).

Os hiperdocumentos multimídia gerados são os de apresentação das informações das reuniões capturadas, ilustrado na Figura 6. Observando a Figura 6, pode-se notar que existem três elementos principais: a área que contém *slides*, a área da linha do tempo e a área de exibição e controle de vídeo.

Na parte central do hiperdocumento encontra-se a área de *slides*, onde a sequência de *slides*, que faz parte da reunião escolhida, será visualizada pelos usuários. Quando o hiperdocumento é visualizado, o primeiro *slide* da sequência é exibido automaticamente.

Abaixo da área de *slides* está a área da linha do tempo. A linha do tempo tem a função de indicar o tempo total da reunião e o tempo atual da apresentação de vídeo (representado pela barra vermelha), além de indicar em que momento as trocas de *slides* ocorreram durante a reunião (representadas pelas barras verdes). A linha do tempo está em destaque na Figura 42.

Abaixo do menu lateral esquerdo encontra-se a área de exibição e controle de vídeo. Essa área, além de apresentar o áudio/vídeo, possui três comandos básicos para manipulação do vídeo: *back*, *play* e *pause*.

³⁸ É possível passar parâmetros para um hiperdocumento através de seu próprio endereço. Por exemplo, o endereço <http://www.emeet.com/http/index.php?ordem=1> passa o parâmetro “ordem” com o valor 1 para o hiperdocumento `index.php`.

Maiores informações sobre a utilização do sistema e a descrição detalhada de cada um desses elementos encontram-se no Apêndice A – Manual do Usuário do *eMeet/SERVE*.

Esses três elementos – vídeo, *slides* e linha do tempo – necessitam estar sincronizados, pois quando os usuários iniciam o vídeo, através do botão *play*, a troca automática de *slides* e a representação do tempo atual da apresentação na linha do tempo devem estar sincronizadas com a reprodução do vídeo.

Alguns requisitos eram desejados no processo de sincronização, tais como:

- A apresentação do áudio/vídeo com os *slides* e a linha do tempo deveriam estar sincronizadas quando o vídeo for iniciado. Quando o vídeo for pausado, a sincronização deveria ser desativada;
- Os hiperdocumentos multimídia não poderiam ser recarregados no cliente;
- O processo de sincronização deveria funcionar em navegadores dos mais utilizados do mercado, tais como Internet Explorer 4 ou superior e Netscape Communicator 4.7 ou superior. O Netscape Communicator ainda deveria funcionar nas plataformas Windows e Linux;
- A barra de tempo atual deveria se deslocar automaticamente à medida que a reprodução de vídeo avança;
- A troca de *slides* deveria ocorrer automaticamente à medida que a reprodução de vídeo avança. Tal troca deveria simular o momento em que o *slide* foi apresentado durante a reunião;
- Os hiperdocumentos multimídia não poderiam conter *frames*;
- Os usuários poderiam avançar ou retroceder os *slides* manualmente. Nesse caso, a reprodução de vídeo e a barra de tempo atual deveriam se deslocar automaticamente para o momento da exibição do *slide* desejado, caso a sincronização estiver ativada (o que é feito quando o vídeo é reproduzido);
- Como podem ocorrer atrasos durante a reprodução do vídeo, a barra de tempo atual da linha do tempo deveria representar o tempo de execução do vídeo e não o tempo real de apresentação;
- O vídeo a ser exibido seria gerado a partir do *eMeet/InCA*, através do *Java Media Framework*, no formato *.avi* ou *.mov*. Logo, o *eMeet/SERVE* deveria reproduzir esses formatos de vídeo.

Houve uma investigação de quais tecnologias seriam as escolhidas para realizar tal implementação. As principais estão descritas a seguir.

A possibilidade mais simples de sincronização seria mapear um *frame* de controle que recarregaria o hiperdocumento multimídia em intervalos de tempos, fazendo com que a barra de vídeo se deslocasse e os *slides* trocassem. Porém, não era desejado que os hiperdocumentos fossem recarregados no cliente e nem mesmo utilizassem *frames*.

Outra possibilidade seria construir um *applet* Java que seria responsável por apresentar o vídeo, os *slides* e a linha do tempo, controlando o sincronismo entre esses elementos. A utilização de um *applet* Java seria compatível com a reprodução do formato de vídeo. Porém, todas as informações relevantes no hiperdocumento estariam dentro do *applet*, além de todo o controle de sincronia. Esses fatos descaracterizariam os hiperdocumentos e esta hipótese, apesar de boa, foi descartada.

Uma possibilidade interessante seria utilizar o *Smil* para gerar os hiperdocumentos multimídia e controlar o sincronismo entre os elementos. O problema é que o *Smil 1.0* normalmente é reproduzido através da aplicação Real Player. Já o *Smil 2.0* é compatível somente com o Internet Explorer 5.5 ou superior. Devido a essa não compatibilidade com os outros navegadores, segundo os requisitos apresentados, essa opção também foi descartada.

Finalmente, após uma série de testes e estudos, optou-se em utilizar uma programação avançada em HTML para implementar os *slides* e a linha do tempo e um pequeno *applet* Java para reproduzir o vídeo. Toda a sincronia e comunicação entre os *slides*, o vídeo e a linha do tempo é feita com JavaScript. A vantagem da utilização do JavaScript é que ele é executado no cliente, não sobrecarregando o servidor e funcionando de acordo com a necessidade do cliente. Essa configuração é totalmente compatível com os requisitos apresentados e apresenta desempenho satisfatório³⁹. A seguir, segue um detalhamento de cada um desses elementos e a forma de implementação do processo de sincronização.

6.4.1. Área de *Slides*

A área de *slides* apresenta todos os *slides*, um de cada vez, de acordo com a sequência capturada no momento da reunião pelo *eMeet/InCA*. A área de *slides* está representada na Figura 46.

O ponto chave da implementação da área de *slides* está no fato de que, quando há troca de *slides*, o hiperdocumento não pode ser recarregado, dando a impressão de que o *slide* muda

³⁹ Como o controle de sincronização é feito no navegador do usuário, ou seja, no próprio cliente, não há aumento de processamento do servidor. Para um desempenho satisfatório da sincronização, espera-se que o usuário do eMeet/SERVE tenha um computador com capacidade de processamento médio, como, por exemplo, um *Pentium II*, da *Intel*, de velocidade superior a 350 MHz.

dinamicamente dentro dos hiperdocumentos. Porém, hiperdocumentos HTML são estáticos e não podem ser alterados sem que sejam recarregados.

Figura 46: Área de *slides* nos hiperdocumentos multimídia. A indicação do *slide* atual é apresentada na parte inferior central. É possível avançar ou retroceder os *slides*, através dos botões “Anterior” e “Próximo”, respectivamente.

O que acontece no *eMeet/SERVE* é que todos os *slides* já estão contidos nos hiperdocumentos multimídia, porém estão apenas escondidos. Isso significa que, por exemplo, se um hiperdocumento possui cinco *slides*, somente um *slide* está aparecendo de cada vez e os demais estão escondidos. Porém, os cinco *slides* já estão presentes no código-fonte do hiperdocumento.

Apesar do código-fonte do hiperdocumento já conter o código dos cinco *slides*, os arquivos das imagens dos *slides* são requisitados ao servidor sob-demanda. Isso significa que somente o primeiro arquivo de imagem do *slide* será requisitado ao servidor quando o hiperdocumento é carregado. Os demais arquivos de imagens dos *slides* serão requisitados à medida que houver necessidade de troca de *slides*. A vantagem é que não há espera para que o hiperdocumento apareça no navegador do cliente. A desvantagem é que pode haver atrasos durante a troca de *slides*, devido ao próprio tempo de requisição e recebimento do arquivo de imagem.

Isso é implementado utilizando o conceito de camadas, através da tag `<DIV>` do HTML. A tag `<DIV>` permite que hiperdocumentos tenham camadas de código HTML, assim como existem camadas (*layers*) em um editor de imagens, como Adobe Photoshop ou Corel Draw.

Portanto, cada *slide* está dentro de uma camada própria. Um atributo importante da tag <DIV> é o *visibility*, que permite que a camada seja exibida ou escondida. Além disso, o atributo *id* define o nome da camada. Um exemplo da utilização da tag <DIV> está ilustrado na Figura 47.

```
<div id="slide1" style="Z-INDEX: 1; LEFT: 200px; VISIBILITY:
visible; POSITION: absolute; TOP: 85px">
...
... //Código HTML
...
</div>
```

Figura 47: Exemplo de utilização da tag <DIV> do HTML.

Para fazer com que camadas de HTML sejam exibidas ou escondidas dinamicamente, é necessário criar funções em JavaScript. É importante ressaltar que os navegadores Internet Explorer e Netscape Communicator possuem diferenças na representação de camadas em JavaScript. O Internet Explorer representa uma camada HTML através do código `document.all['ID'].style` e o Netscape Communicator utiliza o código `document.layers['ID']`. O ID representa o nome da camada. Logo, as funções criadas, fazem um tratamento especial para essa diferença entre os navegadores. Essa foi uma das dificuldades encontradas para realizar a troca “dinâmica” de *slides* em hiperdocumentos HTML.

Portanto, quando há necessidade de troca de *slide*, seja manualmente, pelos usuários ou automaticamente, pelo processo de sincronização, funções em JavaScript, que realizam tal operação dinamicamente, são chamadas.

6.4.2. Área de Linha do Tempo

A área de linha do tempo indica quando ocorre a troca de *slides*, o tempo atual e o tempo total da reunião. A área de linha do tempo está representada na Figura 48.

Figura 48: Área de linha do tempo nos hiperdocumentos multimídia. Existe uma barra vermelha que indica o tempo atual. As demais são verdes e indicam as trocas de *slides*. Os números azuis indicam tempos da reunião.

A barra vermelha indica o tempo atual da reprodução do vídeo, portanto ela se desloca sincronamente com a apresentação de vídeo. As barras verdes indicam o momento em que trocas de *slides* ocorreram e servem de âncoras para avançar ou retroceder a reunião. Os números em

azul representam o tempo inicial, o tempo do primeiro terço da reunião, o tempo do segundo terço da reunião e o tempo total da reunião, respectivamente. Os números azuis também são âncoras para avançar ou retroceder a apresentação do vídeo e dos *slides* para o tempo indicado.

Quando o vídeo é iniciado, o processo de sincronização também é iniciado. Na linha do tempo, somente a barra de tempo atual (vermelha) sofre alterações com o processo de sincronização. À medida que a reprodução do vídeo avança, a linha vermelha deve acompanhá-la, deslocando-se ao longo da linha do tempo. Para isso, o sistema calcula, para cada reunião, quanto equivale uma medida de tempo para um *pixel* de deslocamento horizontal. Essa conta varia de acordo com o tempo de duração total da reunião.

O ponto chave do processo de sincronização na linha do tempo é que, quando a barra vermelha alcança uma barra verde, deve ocorrer a troca de *slides* na área de *slides*. A barra vermelha desloca-se à medida que a apresentação do vídeo avança. Ela também deve ser deslocada quando usuários trocam manualmente os *slides*, caso o vídeo esteja sendo reproduzido.

Para deslocar a barra vermelha, utiliza-se um processo similar aos *slides*. A barra vermelha é uma camada, através da tag <DIV> do HTML. Porém, ao invés de funções em JavaScript exibir e esconder a barra vermelha, o que ocorre é que elas alteram a posição horizontal da camada quando necessário, dando a impressão aos usuários que a barra de tempo atual está “andando”. Isto é feito sem que haja uma nova requisição ao servidor.

Os atributos da tag <DIV> responsáveis pela posição da camada são o TOP e o LEFT, que alteram a posição vertical e horizontal, respectivamente.

Portanto, quando o vídeo é iniciado, a barra de tempo atual sofre deslocamentos horizontais dinâmicas (através do atributo LEFT da tag <DIV>), de acordo com a relação tempo/*pixel*, através de chamadas a funções JavaScript.

6.4.3. Área de Vídeo

A área de vídeo exibe os controles de vídeo, juntamente com a apresentação do mesmo. A área de vídeo está representada na Figura 49.

O ponto chave no processo de sincronização do vídeo é que funções em JavaScript dos hiperdocumentos multimídia precisam se comunicar e trocar informações com o vídeo. Para isso, foi construído um *applet* Java para reproduzir o vídeo. A grande vantagem disso é que, além de reproduzir os formatos desejados (.avi e .mov), a comunicação entre o *applet* Java e o JavaScript é natural dentro de hiperdocumentos. A desvantagem da utilização de um *applet* Java

é que, por utilizar o *Java Media Framework* (JMF), é necessário que o pacote JMF esteja instalado na máquina do cliente.

Figura 49: Área de vídeo durante a apresentação da reunião e os controles *back*, *play* e *pause*, respectivamente.

Os controles do vídeo – *back*, *play* e *pause* – são âncoras e imagens do próprio hiperdocumento HTML. Portanto, somente a reprodução do vídeo é feita pelo *applet* Java. Um dos motivos para a escolha da implementação de um *applet* Java para reprodução do vídeo é que o *eMeet/InCA*, que faz a captura e geração do vídeo, também utiliza a linguagem Java. Logo, a integração e compatibilidade são feitas naturalmente.

O *applet* Java possui algumas funções para manipulação do vídeo que serão chamadas pelo JavaScript. As funções mais importantes são:

- `play()`: inicia a reprodução do vídeo.
- `pause()`: pausa a reprodução do vídeo.
- `getTime()`: retorna o tempo atual da reprodução de vídeo.
- `setTime()`: move a reprodução de vídeo para um momento específico.

As funções do *applet* Java que são acessadas pelos hiperdocumentos devem estar com o modificador `public` na declaração das funções. Tal modificador permite que programas externos, como o JavaScript de um hiperdocumento, acessem o *applet* e utilizem tais funções.

O JavaScript consegue acessar funções públicas do *applet* Java diretamente. Um exemplo de como acessar a função `getTime()` do *applet* Java, através de JavaScript, está ilustrado na Figura 50.

```
<script type="javascript">
actual = document.video.getTime();
</script>
```

Figura 50: Exemplo de acesso a função `getTime()` do *applet* Java através do JavaScript.

Algumas dificuldades para o controle da reprodução do vídeo com a utilização do *applet* Java foram encontradas. Ao iniciar a reprodução do vídeo, o *applet* demora alguns segundos até iniciá-lo. O problema é que durante esse tempo, o *applet* Java encontra-se em um *status*

intermediário entre parado e reproduzindo. Se nesse meio tempo, o *applet* for interrompido ou chamado por outros programas, o vídeo não é reproduzido e uma mensagem de erro é exibida. Para evitar o erro, foi implementado um controle de *status* no processo de sincronização que evita chamadas ao *applet* enquanto o mesmo estiver neste *status* intermediário. É importante observar que somente os botões de *back*, *play* e *pause* foram implementados. Outras funções como o *fast forward* e *fast rewind* não foram implementadas por dificuldades da manipulação de vídeo no Java.

Portanto, o processo de sincronização do vídeo ocorre com a comunicação constante entre o *applet* Java e o JavaScript dos hiperdocumentos. Tal comunicação garante que a linha do tempo e os *slides* sejam atualizados de acordo com o decorrer da reprodução do vídeo. A sincronização do vídeo é ativada somente quando o vídeo é iniciado.

6.4.4. Controlador Central do Processo de Sincronização

Apesar da área de *slides*, da área de linha do tempo e da área de vídeo estarem preparadas para o processo de sincronização, é necessário que exista um controlador central que seja o responsável por trocar os *slides*, atualizar a barra de tempo atual e sincronizar o vídeo, à medida que a reprodução de vídeo avança.

O controlador central fica no código-fonte do próprio hiperdocumento multimídia, ilustrado na Figura 6. Tal controlador é uma função em JavaScript, chamada de `checksync()`, que é automaticamente executada assim que o hiperdocumento é carregado no navegador do cliente. Isso é feito através da chamada ilustrada na Figura 51.

```
<script language="javascript">
window.onload = checksync;
</script>
```

Figura 51: Função `checksync()` é chamada assim que o hiperdocumento é carregado no navegador do cliente.

Uma vez executada, a função `checksync()` inicia um contador, no cliente, responsável por chamar novamente a própria função em intervalos de tempo. Esse intervalo de tempo é configurável no arquivo de configuração da aplicação do *eMeet/SERVE*. A função de sincronização deve ser chamada em intervalos de tempo, pois ela é a responsável por atualizar os *slides*, a barra de tempo atual e a reprodução de vídeo.

A função `checksync()` realiza seis passos para controlar a sincronização entre os *slides*, a linha de tempo e o vídeo. A Figura 52 ilustra, em linhas gerais, os seis passos de sincronização, que são discutidos a seguir:

1. Verificar se o vídeo foi iniciado, pois somente com o vídeo em reprodução é que o processo de sincronização automática é ativado. Além disso, a função verifica se o *status* do *applet* é válido para manipular informações. Como observado na seção anterior, o *applet* pode estar em um *status* em que ele não pode aceitar instruções, causando problemas na reprodução do vídeo e exibindo erros no hiperdocumento. Se o vídeo foi iniciado e está com *status* válido, a função continua a execução dos próximos passos. Caso contrário, ela termina e espera ser chamada novamente;
2. Obter o tempo atual da reprodução do vídeo;
3. Calcular qual *slide* deve ser mostrado de acordo com o tempo de vídeo obtido no Passo 2. Esta verificação é feita em memória no cliente, sem qualquer consulta no servidor;
4. Calcular a nova posição da barra de tempo atual a partir do tempo atual de vídeo capturado no Passo 2. Este cálculo também é feito em memória no cliente;
5. Trocar o *slide* que deve ser mostrado, calculado no Passo 3;
6. Mover a barra de tempo atual para a posição calculada no Passo 4.

Figura 52: Passos da função responsável pela sincronização.

O processo de sincronização é realizado com esses seis passos sendo executados em intervalos de tempos. A maior vantagem desse processo de sincronização é que ele necessita de baixo processamento, tendo um atraso praticamente imperceptível aos usuários e não atrapalha a apresentação das informações da reunião. A desvantagem é que em reuniões muito curtas com duração de segundos, a barra de tempo atual parece dar saltos muito grandes. O tempo de

atualização é configurável, porém, estima-se que o intervalo de tempo ideal para a função de sincronia ser chamada é de três segundos. Isso significa que de três em três segundos, os *slides* e a barra de tempo atual serão sincronizados com a reprodução do vídeo.

6.5. Controle de acesso

Cada reunião do *eMeet* conta com um mecanismo de controle de acesso, responsável por permitir ou não que usuários acessem os hiperdocumentos com informações de uma determinada reunião, de acordo com seus perfis.

Controle de Acesso:*

- ☐ Livre - público a qualquer pessoa que acessar o sistema. ?
- ☐ Cadastro - apenas para pessoas cadastradas no sistema. ?
- ☒ Participantes - restrito aos participantes dessa reunião. ?
- ☐ Administrador - somente o administrador do sistema tem acesso. ?

Figura 53: Formulário para escolha do nível de controle de acesso de uma reunião.

Existem quatro níveis de controle de acesso nas reuniões do *eMeet/SERVE*, que variam de acordo com a necessidade de segurança das informações. A Figura 53 mostra o formulário responsável pela escolha do nível de controle de acesso. São eles:

1. Livre: Qualquer usuário, incluindo visitantes, poderá acessar as informações da reunião. Não há necessidade dos usuários realizarem autenticação através do *login*.
2. Cadastro: Para acessar os hiperdocumentos com as informações da reunião os usuários devem ser cadastrados no *eMeet/SERVE* e efetuar autenticação através do *login*.
3. Participantes: Somente os participantes da reunião poderão acessar os hiperdocumentos com informações da reunião. Tais participantes devem realizar a autenticação no sistema, através do *login*. Os participantes de uma reunião são definidos a qualquer momento no *eMeet/SERVE*, juntamente com as suas salas, conforme ilustra a Figura 54. Isso pode ser feito desde que a reunião já tenha sido criada no *eMeet*.
4. Administrador: Apenas o administrador do *eMeet/SERVE* terá acesso aos hiperdocumentos com informações da reunião. A realização da autenticação é obrigatória através do *login*.

Conforme a Figura 54, os participantes de uma reunião podem estar distribuídos fisicamente em salas virtuais distintas. Durante o cadastro de um participante, é necessário informar qual sala virtual ele pertence. No *eMeet/SERVE*, existe uma seção própria para o gerenciamento (cadastro, edição e remoção) de salas virtuais.

O controle de acesso do *eMeet/SERVE* foi implementado através de variáveis de sessão da linguagem PHP. Quando um usuário realiza a autenticação no sistema, através do *login*, variáveis de sessão são criadas dinamicamente no servidor Web. Um *cookie*⁴⁰ de memória é enviado para o navegador do usuário, a fim de identificar qual o navegador responsável pela criação das variáveis de sessão. Tal *cookie* é mantido em memória e portanto não é armazenado na máquina do cliente. Quando o usuário realiza uma nova requisição no servidor Web do *eMeet*, o *cookie* de memória é lido pelo servidor e o usuário é identificado.

Caso o navegador do usuário esteja configurado para não aceitar *cookies*, a linguagem PHP passa a utilizar automaticamente outras formas de controle de sessão.

Olá, **Administrador**.

[Logout](#)

Reuniões
Inserir/Remover salas e participantes na reunião: **(Reunião Teste)**

Lista de salas e usuários atualmente cadastrados nessa reunião: ?

Nome da Sala:	Nome do Usuário:	
sala USP	hey you	Remover
Sala de Conferências	popo	Remover

Selecione salas e usuários para cadastrá-los nessa reunião: ?

Nome da Sala:* ?

Nome do Usuário:* ?

* Campos de preenchimento obrigatório.

[Terminar](#)

Figura 54: Hiperdocumento com formulário para cadastro/remoção de participantes em uma reunião.

Cada reunião pode ter o seu próprio nível de controle de acesso aos hiperdocumentos gerados. Com isso, é possível segmentar o conteúdo dos hiperdocumentos gerados entre os diversos usuários do sistema, através das permissões de cada usuário e de cada reunião.

⁴⁰ *Cookies* são informações que são enviadas para o navegador do cliente e são lidas quando tal navegador realiza uma nova requisição no mesmo servidor Web.

6.6. Considerações finais

Conceitos como usabilidade (Nielsen, 1993), consistência (Dix et al., 1998) e facilidade de navegação (Rocha and Baranauskas, 2000) foram utilizados durante o desenvolvimento das interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE*. Além disso, regras fornecidas por Netto (2002) também auxiliaram no desenvolvimento dessas interfaces.

Apesar de utilizar diversas regras e conceitos de usabilidade e interação em suas interfaces, os hiperdocumentos gerados pelo *eMeet/SERVE* não foram formalmente avaliados para determinar realmente quão usáveis são. Porém, muitos cuidados foram tomados e padrões foram seguidos para o desenvolvimento dessas interfaces.

Alguns aspectos específicos de interação usuário-computador foram discutidos neste capítulo, apresentando características importantes voltadas à interface de hiperdocumentos. Tais características podem ser utilizadas por qualquer outro sistema de geração de hiperdocumentos.

O tratamento de sincronização realizado entre os elementos (*slides*, vídeo e linha do tempo) dos hiperdocumentos multimídia do *eMeet/SERVE* foi uma das maiores contribuições deste trabalho. A sincronização respeitou uma lista de requisitos desejados e foi realizada utilizando um *applet* para apresentação do vídeo e funções em JavaScript para realizar a comunicação e controle entre os elementos.

Os hiperdocumentos gerados pelo *eMeet/SERVE*, responsáveis em apresentar as informações das reuniões capturadas, contam com quatro níveis de controle de acesso de usuários. Com isso, é possível segmentar o conteúdo das reuniões entre usuários distintos.

Capítulo 7. Conclusões

7.1. Considerações iniciais

Hiperdocumentos estão sendo cada vez mais utilizados como forma de apresentação de informações capturadas em experiências ao vivo. Algumas características importantes relativas à evolução desses hiperdocumentos não estão presentes nos sistemas responsáveis em realizar a geração automática dos mesmos, conforme reportados na literatura.

Este trabalho teve como objetivo implementar um sistema de geração automática de hiperdocumentos multimídia a partir de informações capturadas em reuniões distribuídas na Web, denominado *eMeet/SERVE*. O *eMeet/SERVE* foi implementado utilizando a arquitetura LAIAc, proposta neste trabalho, a fim de suportar características desejáveis a um sistema gerador de hiperdocumentos, como a possibilidade de atualizações constantes de conteúdo e de *layout*.

7.2. Discussão dos resultados

Este trabalho apresentou a arquitetura LAIAc de suporte para desenvolvimento de sistemas de geração de hiperdocumentos. Além disso, foi implementado o *eMeet/SERVE*, sistema de geração automática de hiperdocumentos multimídia, utilizando-se a arquitetura LAIAc. As principais vantagens que o *eMeet/SERVE* adquiriu por utilizar tal arquitetura foram:

- Independência de base de dados e dispositivos de armazenamento;
- Suporte a múltiplos idiomas;
- *Layout* configurável em alguns aspectos comuns;
- Integração com reutilização dos serviços desenvolvidos por outras aplicações;
- Preparação para instalação, manutenção e expansão.

O *eMeet/SERVE* utilizou parte da especificação UML durante a fase de modelagem e análise de requisitos. A partir dessa modelagem, foi definido que o *eMeet/SERVE* teria quatro seções independentes: Seção Reuniões, Seção Salas, Seção Usuários e Seção Fale Conosco.

As interfaces dos hiperdocumentos gerados pelo *eMeet/SERVE* foram desenvolvidas utilizando-se padrões de usabilidade, que foram apresentados neste trabalho. Tais padrões

ilustram de uma forma prática algumas regras já conhecidas na literatura ou que foram propostas neste trabalho, que podem ser utilizados para melhorar a usabilidade da interface de hiperdocumentos.

Além disso, este trabalho apresentou uma forma de sincronização entre elementos multimídia em hiperdocumentos, como vídeo, *slides* e linha do tempo. O vídeo é reproduzido através de um *applet* Java. Toda a sincronização é baseada na linguagem JavaScript, e é compatível com os dois principais navegadores existentes, como Internet Explorer e Netscape Communicator. Além disso, a sincronização respeita uma série de requisitos que são desejáveis a muitos sistemas Web.

Desenvolvedores de sistemas Web podem utilizar o *eMeet/SERVE* como modelo a ser seguido, tanto na parte de arquitetura, quanto na parte de sincronização, apresentação e estruturação dos hiperdocumentos.

Leigos em interfaces podem ter uma idéia prática de conceitos de usabilidade, através dos padrões utilizados por este trabalho. Especialistas em interfaces poderão realizar avaliações nas interfaces desenvolvidas do *eMeet/SERVE*, a fim de medir a sua real usabilidade.

7.3. Contribuições

As contribuições deste trabalho são:

1. Identificação, estudos e classificação de sistemas de captura e apresentação de informações multimídia na Web, através de hiperdocumentos;
2. Definição da arquitetura LAIAc de suporte a sistemas de geração de hiperdocumentos;
3. Documentação da utilização de padrões de usabilidade, que ilustram, de forma prática, alguns princípios já consagrados de usabilidade em hiperdocumentos, além de algumas outras regras propostas pelo autor deste documento;
4. Implementação de um mecanismo de sincronização entre elementos multimídia em hiperdocumentos;
5. Implementação de um mecanismo de controle de acesso de usuários em hiperdocumentos;
6. Implementação do sistema *eMeet/SERVE* para acessar, recuperar e apresentar as informações (vídeo, áudio e *slides* com anotações) capturadas em reuniões distribuídas,

através da geração automática de hiperdocumentos. Esse sistema foi implementado com o uso da arquitetura LAIAc proposta e as interfaces dos hiperdocumentos gerados foram desenvolvidas a partir de padrões de usabilidade.

7.4. Trabalhos Futuros

Como trabalhos futuros destacam-se:

1. Inclusão de fóruns de discussão, formando comunidades virtuais e permitindo que discussões sejam realizadas no ambiente Web do *eMeet/SERVE*. A dificuldade de realização desse trabalho é pequena, pois é possível integrar a *CoTeia* (Arruda Jr. & Pimentel, 2001) com o *eMeet/SERVE*;
2. Inclusão de novas maneiras de navegação e visualização de informações dos hiperdocumentos gerados, tais como vídeo, áudio e *slides*. A realização desse trabalho facilitará a visualização de informações pelos usuários no *eMeet/SERVE*, melhorando a usabilidade do sistema. A dificuldade de realização desse trabalho varia de acordo com as novas maneiras de navegação e visualização desejadas;
3. Inclusão de mecanismos de busca por informações, permitindo que usuários procurem por determinados trechos das informações capturadas. A dificuldade de realização desse trabalho é variável de acordo com os mecanismos de busca a serem implementados;
4. Inclusão de salas de conversação, permitindo que usuários debatam sincronamente assuntos de seus interesses mesmo após a sessão da reunião. A dificuldade de realização desse trabalho é pequena, pois é possível integrar o *eMeet/SERVE* com outras aplicações já desenvolvidas, incluindo salas de conversação;
5. Inclusão de novos tipos e dispositivos de armazenamento de dados e metadados. Com a realização desse trabalho, o *eMeet* poderá ser utilizado em ambientes que não tenham o banco de dados MySQL disponível para utilização, aumentando a compatibilidade do sistema com outros meios de armazenamento. A dificuldade de realização desse trabalho é pequena, pois o *eMeet* utiliza a arquitetura LAIAc que mantém o acesso aos dados e metadados em camadas independentes e preparadas para receber novas expansões;

6. Expansão da arquitetura LAIAc para geração de outros tipos de conteúdos, como XML⁴¹, além de hiperdocumentos HTML. A dificuldade de realização desse trabalho é pequena, uma vez que somente a parte referente à aplicação da arquitetura precisará ser reformulada. Porém, a dificuldade para converter o *eMeet/SERVE* para gerar hiperdocumentos através de XML é média, pois os arquivos gerados da aplicação precisarão ser reescritos;
7. Avaliação das interfaces visuais dos hiperdocumentos gerados a fim de verificar seus níveis de usabilidade. Após essa avaliação, alterações na interface visual podem ser realizadas a fim de melhorar a usabilidade dos hiperdocumentos gerados. A dificuldade de realização deste trabalho pode variar segundo nível de avaliações e alterações desejadas;
8. Implementação de geradores de resumos automáticos de reuniões, permitindo que usuários que não participaram das reuniões possam visualizar, de forma rápida, as partes mais importantes da reunião. A dificuldade de realização desse trabalho é grande, uma vez que estudos devem ser realizados e mecanismos de resumos devem ser implementados;
9. Inclusão de mecanismos que possibilitem a adição de novas informações a uma reunião previamente capturada. Isso permitirá que o conteúdo capturado esteja sempre atualizado. A dificuldade de realização deste trabalho varia de acordo com os mecanismos de adição desejados;
10. Inclusão de mecanismos que possibilitem o reaproveitamento das informações capturadas para serem utilizadas em novas reuniões, facilitando a preparação de novas reuniões. A dificuldade de realização deste trabalho varia entre média e grande, pois estudos e implementações deverão ser realizados a fim de recuperar e utilizar os dados já capturados em novas reuniões;
11. Integração do *eMeet/SERVE* com outros serviços já implementados pelo grupo de Hipermídia do ICMC/USP. A dificuldade desse trabalho é pequena, pois o *eMeet/SERVE* utiliza a arquitetura LAIAc e está preparado para ser integrado com outras aplicações.

⁴¹ XML – *eXtensible Markup Language*. Mais informações encontram-se em <http://www.w3.org/xml/>.

7.5. Considerações finais

O sistema *eMeet* foi implementado para dar suporte a reuniões virtuais distribuídas. A parte *SERVE*, responsável por acessar, recuperar e apresentar as informações capturadas durante as reuniões distribuídas, através da geração sob-demanda de hiperdocumentos multimídia, foi desenvolvida com a utilização da arquitetura LAIAc. Esta primeira versão do software está preparada para integração com os demais serviços implementados no contexto do projeto InCA-SERVE, visando a evolução contínua dos hiperdocumentos gerados automaticamente.

Referências Bibliográficas

- (Abowd & Mynatt, 2000) Abowd, G.; Mynatt, E. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction*, vol. 7, nº1, p. 29-58, mar 2000.
- (Abowd, 1999) Abowd, G. (1999). Classroom 2000: an experiment with the instrumentation of a living educational environment.. *IBM Systems Journal*, vol. 38, n.4, p. 508-530, out. 1999.
- (Abowd et al., 1998a) Abowd, G.; Atkenson, C.; Brotherton, J.; Enqvist, T.; Gully, P.; Lemon, J. (1998a). Investigating the capture, integration and access problem of ubiquitous computing in an educational setting.. *Proc. CHI '98*, ACM, New York, p.440-447, 1998.
- (Abowd et al. 1998) Abowd, G.; Brotherton, J.; Bhalodia, J. (1998b). Classroom 2000: a system for capturing and accessing multimedia classroom experiences. *Proc. CHI '98*, mai. 1998.
- (Abowd et al. 1999) Abowd, G.; Pimentel, M.G.C.; Kerimbaev, B.; Ishiguro, Y. E Guzdial M. (1999). Anchoring discussions in lecture: an approach to collaboratively extending classroom digital media. *Proceedings of Computer Supported Collaborative Learning CSCL '99*. Stanford University, 1999.
- (Akscyn et al. 1988) Akscyn, R.M.; Mccracken, D.L.; Yoder, E.A. (1988). KMS: a distributed hypermedia system for managing knowledge in organizations. *Communications of the ACM* 31(7), p.820-835.
- (Arruda Jr. & Pimentel, 2001) Arruda Jr., C.R.E.; Pimentel, M.G.C. (2001). Projeto e implementação de um sistema colaborativo de edição. *Revista Eletrônica de Iniciação Científica da Sociedade Brasileira de Computação (REIC-SBC)*. Ano 1, nov. 2001.
- (Bila, 2002) Bila, W. S. (2002). *eMeet/InCA: captura e acesso de informação multimídia em reuniões distribuídas*. Dissertação apresentada para obtenção do título de mestre, ICMC/USP, 2002.
- (Brotherton et al., 1998) Brotherton, J.A. E Abowd, G.D. (1998). Rooms take note: room takes notes! *Working Papers Of Aaai '98 Spring Symposium*, mar 1998.
- (Bush, 1945) Bush, V. (1945). As we may think. *Atlantic Monthly*, jul 1945 (também em

<http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm>)

- (Chiu et al., 1999) Chiu P.; Kapuskar A.; Reitmeier, S.; Wilcox, L. (1999). Notelook: taking notes in meetings with digital video and ink. *ACM Multimedia '99*, Orlando, Fl. USA. out 1999.
- (Christel et al., 1995) Christel, M.; Kanade, T.; Mauldin, R.; Sirbu, M.; Stevens, S.; Wactlar, H. (1995). Informedia digital video library. *Communications of the ACM*, 38(4):p.57-58, 1995.
- (Conklin, 1987) Conklin, J. (1987). Hypertext: an introduction and survey. *IEEE Computer*, p.17-41, set 1987.
- (Dix et al., 1998) Dix, A.; Finlay, J.; Abowd, G., Beale, R. (1998). *Human-Computer Interaction*. Prentice Hall, Europe, second edition, 1998.
- (Eleutério & Pimentel, 2001) Eleutério, P.M.S.; Pimentel, M.G.C. (2001). Geração sob Demanda de Hiperdocumentos Multimídia Abertos a Partir de Informação Capturada em Reuniões Distribuídas. *I Workshop de Teses e Dissertações do VII Simpósio Brasileiro de Sistemas Multimídia e Hiperemídia*, pages 313-320, Florianópolis, Brazil, 2001.
- (Ellis et al., 1991) Ellis, C.A.; Gibbs, S.J.; Rein, G.L. (1991). Groupware: some issues and experiences. *Communication of the ACM*, 34(1), 38-58, 1991.
- (Engelbart, 1984a) Engelbart, D. (1984a). Authorship provisions in augmente. *Proceedings IEEE Comcon Conference* (p. 465-472). San Francisco: IEEE.
- (Engelbart, 1984b) Engelbart, D. (1984b) Collaboration support provisions in augmente. *In AFIPS Office Automation Conference, Oac'84 Digest* (p. 51-58). Los Angeles. San Francisco: AFIPS.
- (Grønbaek & Trigg, 1999) Grønbaek K.; Trigg R. H. (1999). *From web to workplace: designing open hypermedia systems*. The Mit Press, Cambridge, Massachusetts, 1999.
- (Guzdial, 1999) Guzdial, M. (1999). Supporting learners as users. *The Journal Of Computer Documentation*, 23(2), p. 3-13, 1999.
- (Halasz, 1987) Halasz, F.G. (1987). Notecards in a nutshell. *Proceedings Of The 1987 ACM Conference of Human Factors in Computer Systems*, p. 45-52, Abril 1987
- (Halasz, 1988) Halasz, F.G. (1988). Reflections on notecards: seven issues for the next generation of hypermedia systems. *Communications Of The ACM*, 31(7): p.836-852, 1988.

- (He et al., 2000) He, L.; Grudin, J.; Gupta, A. (2000). Designing presentations for on-demand viewing. *Proceedings of the ACM Conference on Computer-Supported Cooperative Work '00*. p. 127-134, 2000.
- (He et al., 1999) He, L.; Sanocki, E.; Gupta, A.; Grudin, J. (1999). Auto-summarization of Audio-video presentations. *ACM Multimedia '99*, Orlando, Fl. USA, p.489-498, out 1999.
- (Hindus & Schmandt, 1992) Hindus, D.; Schmandt, C. (1992). Ubiquitous audio: capturing spontaneous collaboration. *Proceedings of the ACM Conference On Computer-Supported Cooperative Work (Cscw'92, Toronto, Canada, Oct. 31-Nov.4)*, M. Mantel; R. Baecker, Eds. ACM Press, New York, Ny, p.210-217.
- (Jancke et al., 2000) Jancke, G.; Grudin, J.; Gupta, A. (2000). Presenting to local and remote audiences: design and use of the TELEP System. *Proc. Chi '2000*, p.384-391. 2000.
- (Li et al., 2000) Li, F. C.; Gupta, A.; Sanocki, E.; He, L.; Rui, Y. (2000) Browsing digital video. *Proc Chi' 2000*. p.169-176. 2000.
- (Maurer et al. 1999) Maurer H.; Rozsenich, N.; Sapper, M. (1999). How to make discussion forums work on the WWW. *Proc. Webnet 99*, Aace, Charlottesville, USA (1999), p.717-722
- (Meyrowitz et al. 1986) Meyrowitz, N.; Garrett, L.N.; Smith, K.E. (1986). Intermedia: issues, strategies, and tactics in the design of a hypermedia document system. *Proceedings from The Conference on Computer Supported Work* (Pp. 163-174). Austin, Tex.: Mcc Software Technology Program.
- (Minneman et al., 1995) Minneman, S.; Harrison, S.; Janseen, B., Kurtenback, G.; Moran, T.; Smith, I. E Van Melle, B. (1995). A confederation of tools for capturing and accessing collaborative activity. *ACM Conference On Multimedia (Multimedia'95)*. San Francisco, CA pp. 523-533, nov 1995.
- (Morkes & Nielsen, 1997) Morkes, J.; Nielsen, J. (1997) Concise, Scannable, And Objective: How to write for the Web. <http://www.useit.com/papers/webwriting/writing.html>
- (Mukhopadhyay & Smith, 1999) Mukhopadhyay S.; Smith, B. (1999) Passive capture and structuring of lectures. *ACM Multimedia '99*, Orlando, Fl. USA, out 1999.
- (Netto, 2002) Heurísticas e *Guidelines* para Apresentação de Hiperdocumentos Multimídia na Web. Dissertação apresentada para obtenção do título de mestre, ICMC/USP, 2002.
- (Nielsen, 1993) Nielsen, J. (1993). *Usability Engineering*. Academic Press, London, United

Kingdom, 1993.

- (Nielsen, 1995) Nielsen, J. (1995) Guidelines for multimedia on the Web. <http://www.useit.com/alertbox/9512.html>.
- (Nielsen, 1997) Nielsen, J. (1997). Concise, Scannable, and Objective: How to Write for the Web. <http://www.useit.com/papers/webwriting/writing.html>. On-line in World Wide Web.
- (Pedersen et al., 1993) Pedersen, E.; McCall, K.; Moran, T.P.; Halasz, F. (1993). Tivoli: an electronic whiteboard for informal workgroup meetings. *Proceedings Interchi 1993*. Amsterdam, The Netherlands, p. 391-398, apr 1993.
- (Pimentel et al., 2001a) Pimentel, M.G.C.; Macedo, A.A.; Abowd, G.D. (2001) Linking homogeneous web-based repositories. *International Workshop On Information Integration On The Web WIW'01*. Rio de Janeiro, abril 2001.
- (Pimentel et al., 2001b) Pimentel, M.G.C.; Abowd, G.; Kerimbaev, B.; Ishiguro, Y.; Guzdial, M. (2001b). Supporting long-term educational activities through dynamic web interfaces. *Journal Interacting With Computers*, 13(3):353-374, fev 2001.
- (Pimentel et al., 2000) Pimentel, M.G.C.; Abowd, G. Ishiguro, I. (2000). Linking by interacting: a paradigm for authoring hypertext. *Proceedings of the ACM Hypertext 2000*. San Antonio, TX, p. 39-48, mai 2000.
- (Pimentel & Abowd, 1999) Pimentel, M.G.C.; Abowd, G. (1999). Development and understanding of automated capture environments to support long-term use. Projeto De Cooperação Internacional Do PROTEM-CC-CNPq/Brasil E NSF/EUA, Out. 1999.
- (Pressman, 2000) Pressman, R.S. (2000). *Software Engineering – A Practitioner's Approach*. Fifth edition. McGraw-Hill, 2000.
- (Richter et al., 2001) Richter, H.; Abowd, G.; Geyer, W.; Fuchs, L.; Daijavad, S.; Poltrock, S. (2001) Integrating Meeting Capture within a Collaborative Team Environment. *Proceedings of Ubicomp 2001*, Atlanta, Georgia, Set/2001, p. 123-138.
- (Rocha & Baranauskas 2000) Rocha, H.; Baranauskas, M. (2000). *Design e Avaliação de Interfaces Humano-Computador*. Escola de Computação 2000, Departamento de Ciência da Computação – USP, São Paulo, SP, 2000.
- (Rosenberg & Scott, 1999) Roserberg, D.; Scott, K.; (1999). *Use Case Driven Object Modeling*

with UML – A Practical Approach. Addison-Wesley Pub. Co. March, 1999.

- (Schmandt & Arons, 1985) Schmandt, C.; Arons, B. (1985). Phone Slave: a graphical telecommunications interface. *Proceedings Of The Society For Information Display*, 26(1): p.79-82, 1985.
- (Steinmetz & Nahrstedt, 1995) SteinMetz, R.; Nahrstedt, K. (1995) *Multimedia: Computing, Communications and Applications*. Prentice Hall, Hardcover. May, 1995.
- (Trigg & Suchman, 1989) Trigg, R. H.; Suchman, L.A. (1989). Collaborative writing in notecards in R. Mcaleese (Ed.), *Hypertext: Theory Into Practice* (p. 45-61). Norwood, N.J.: Ablex.
- (Weiser, 1993) Weiser, M. (1993). Some computer science issues in ubiquitous computing. *Communications of the ACM*. v.6 n°7. p. 75-84, jul 1993.

Apêndice A. Manual do Usuário do

A.1. Introdução e requisitos

Este documento tem como finalidade instruir os usuários sobre como proceder na utilização do *eMeet/SERVE*. A linguagem utilizada para a escrita deste manual é voltada para os usuários do *eMeet/SERVE*.

Para utilizar o *eMeet/SERVE*, os usuários devem respeitar os seguintes requisitos:

- Estar conectado na Internet ou ter acesso ao servidor do *eMeet/SERVE*;
- Ter instalado na sua máquina o JMF1.2;
- Utilizar os navegadores Internet Explorer 4 ou Netscape Communicator 4.7 ou superiores.

Durante todas as páginas do sistema, o usuário visualizará um texto curto indicando a sua localização dentro do sistema e algumas instruções sobre o que ele pode realizar na página. Normalmente, tais textos estarão na parte superior da página. Além disso, o usuário poderá obter ajuda on-line, se desejar, passando o cursor do mouse sobre o ícone de ajuda, representada pela Figura 44. Ao deixar o cursor do mouse sobre tal indicação, um texto de ajuda será exibido, auxiliando a operação do sistema.

Algumas seções são de acesso exclusivo do administrador do sistema. Portanto, para acessar todas as seções descritas a seguir, o usuário deverá estar autenticado como administrador.

O *eMeet/SERVE* é composto basicamente de cinco seções, que são discutidas a seguir.

A.2. Página Inicial

Ao acessar o endereço do *eMeet/SERVE*, o usuário visualizará a tela indicada na Figura 55. Esta é a tela de introdução do *eMeet/SERVE* e contém algumas dicas sobre a utilização do sistema, bem como os requisitos para utilização. Além disso, o menu lateral esquerdo contém as seções existentes no sistema. Para acessá-las, basta clicar na seção desejada.

Abaixo do menu lateral esquerdo encontra-se um espaço para o usuário realizar o *login* no sistema. Caso ele seja cadastrado no sistema, é recomendável que ele digite seu usuário e senha e realize o *login* no sistema. Isso é necessário, pois o acesso a algumas reuniões é exclusivo para um determinado perfil de usuário.

O usuário poderá realizar o seu *login* a qualquer momento e em qualquer página, pois os campos de usuário e senha estarão sempre presentes abaixo do menu lateral esquerdo até que o *login* seja feito. Quando isto acontecer, os campos de *login* serão substituídos pela indicação de que o usuário está autenticado, além do botão de *logout*.

Figura 55: Página inicial do *eMeet/SERVE*.

Assim como o *login*, o *logout* pode ser feito a qualquer momento, clicando-se no botão *logout*. Quando o usuário realiza o *logout*, ele volta a visualizar a página inicial do *eMeet/SERVE*.

O usuário pode voltar a página inicial do sistema a qualquer momento, clicando na ligação “Início” do menu lateral esquerdo ou no logo “*eMeet*” no canto superior esquerdo da página, acima do menu.

A.3. Seção Reuniões

Ao acessar a Seção Reuniões, o usuário visualizará a página inicial dessa seção, que conterà a lista de serviços oferecidos. A Seção Reuniões é a responsável por oferecer serviços referentes às reuniões do *eMeet/SERVE*, como, por exemplo, a visualização das reuniões cadastradas. Tal página está representada na Figura 56.

Figura 56: Página inicial da Seção Reuniões.

Assim como em qualquer outra página do sistema, o menu lateral esquerdo, o espaço de indicação *login/logout* e os ícones de ajuda estão presentes nas páginas dessa seção.

Os serviços oferecidos nesta seção são o cadastro, a visualização, a edição e a remoção de reuniões. Tais serviços são discutidos a seguir.

A.3.1. Cadastro de novas reuniões

Para cadastrar uma reunião no sistema, o usuário deverá preencher um formulário contendo os seguintes campos, representados pela Figura 57:

- Nome da reunião: este campo deve conter o nome da reunião. É por esse nome que a reunião será identificada no sistema. Este é um campo obrigatório e deverá ser preenchido necessariamente.
- Campo de comentário: este campo deve conter outras informações sobre a reunião, como, por exemplo, o motivo da realização da mesma ou apenas um lembrete que melhor identifique a reunião no sistema.
- Controle de acesso: as informações das reuniões poderão ser visualizadas por diversos grupos de acesso, que variam desde somente o administrador até todos os usuários do sistema. O usuário deverá escolher um dos quatro níveis de controle de acesso existentes no *eMeet/SERVE* de forma a melhor proteger as informações contidas na reunião.

Além dos campos descritos acima, o usuário terá a opção de inserir salas e participantes da reunião. Para isso, ele deve marcar a caixa com a indicação “Inserir Salas e Participantes da reunião (na tela seguinte)”.

Preencha corretamente os campos abaixo: ?

Nome da Reunião:* ?

Campo de comentário: ?

Controle de Acesso:*

- ☐ Livre - público a qualquer pessoa que acessar o sistema. ?
- ☐ Cadastro - apenas para pessoas cadastradas no sistema. ?
- ☒ Participantes - restrito aos participantes dessa reunião. ?
- ☐ Administrador - somente o administrador do sistema tem acesso. ?

☒ Inserir salas e participantes da reunião (na tela seguinte). ?

Figura 57: Campos para cadastro de novas reuniões.

Após preencher os campos acima, o usuário deverá clicar no botão “Criar”. Dependendo da opção de cadastrar salas e participantes, o usuário poderá finalizar o cadastro da reunião ou não.

Se o usuário decidiu realizar o cadastro de salas e participantes da reunião, a página seguinte, representada pela Figura 54, será visualizada. Nesta página, o usuário visualizará as salas e participantes já cadastrados na reunião (se houver algum) e terá a opção de removê-los ou de cadastrar novas salas e participantes para a reunião.

Na parte superior da página será exibida a lista de todas as salas e participantes já cadastrados na reunião. Para removê-los, o usuário deverá clicar no botão “Remover” que está localizado à direita de cada participante.

A parte inferior da página é responsável pelo cadastro de novas salas e participantes. Para isso, o usuário deverá selecionar uma sala e um participante e clicar no botão “Cadastrar”. Instantaneamente, o usuário passará a visualizar a sala e o participante cadastrado na lista de cadastros atuais.

O usuário pode terminar a operação clicando no botão “Terminar”. É importante ressaltar que todos esses dados poderão ser editados a qualquer momento.

A.3.2. Visualização de reuniões

Ao clicar nessa opção, o usuário verá uma lista de reuniões, de acordo com a Figura 58. É importante salientar que somente as reuniões cujo usuário tem permissão de visualização serão listadas, portanto, tal lista varia de acordo com o usuário que está utilizando o sistema.

Para visualizar as informações de uma determinada reunião, clique no botão “Assistir” da reunião correspondente. A partir daí, o usuário visualizará a página mais importante do *eMeet/SERVE*, representada pela Figura 6, que é a página de apresentação dos dados de uma reunião.

Figura 58: Lista de reuniões que o usuário poderá acessar.

Observando a página representada pela Figura 6, pode-se notar as três partes principais: a área de *slides*, a linha do tempo e o vídeo. Além dessas três partes, que são detalhadas a seguir, existe a opção “Dados da reunião”, que exibe uma janela com algumas informações da reunião, como duração, data e participantes, conforme ilustra a Figura 59.

O usuário ainda pode optar em clicar no botão Ajuda, e uma nova janela irá se abrir com informações mais detalhadas sobre como operar a parte principal do sistema, que é justamente a apresentação dos dados da reunião. O usuário continua podendo consultar as ajudas on-line, que são representadas pelos ícones de ajuda mostrados na Figura 44.

Figura 59: Janela com informações sobre a reunião.

A.3.2.1. Área de slides

A área de *slides*, localizada na parte central da página é responsável pela exibição dos *slides* que fazem parte da reunião escolhida. A área de *slides* está representada pela Figura 46.

O primeiro *slide* é exibido automaticamente quando a página é carregada. A indicação de qual *slide* está sendo exibido está localizada na parte central da tabela, logo abaixo do *slide*, entre os botões “Próximo” e “Anterior”. Para avançar ou retroceder *slides* a qualquer momento, clique nos botões “Próximo” e “Anterior”, respectivamente.

Quando o vídeo é iniciado, os *slides* são trocados automaticamente, de acordo com o tempo do vídeo. A troca de *slides* automática simula o momento exato em que o *slide* foi apresentado na reunião durante a fase de captura. Se os botões “Próximo” e “Anterior” forem acionados nesse momento, o *slide* é trocado e o vídeo avança ou retrocede para o momento exato em que tal *slide* foi exibido. Portanto, quando o vídeo é iniciado qualquer ação referente aos *slides* é refletida no vídeo.

A.3.2.2. Linha do tempo

A linha do tempo, localizada na parte inferior da página, tem a função de indicar o tempo total da reunião, além de indicar o tempo atual da apresentação de vídeo e quando ocorrerão as trocas de *slides*. A linha do tempo está representada pela Figura 48.

O tempo total da reunião pode ser visto abaixo da linha do tempo, onde existem indicações do tempo inicial, de 1/3 do tempo total da reunião, de 2/3 do tempo total da reunião e, finalmente, do tempo total da reunião. Tais indicações estão dispostas da esquerda para a direita ao longo da linha do tempo e servem para mover a apresentação para o tempo indicado.

O traço vermelho, que inicialmente está no início da linha do tempo, indica o tempo atual da apresentação. Quando o vídeo é iniciado, a linha vermelha começa a se mover, à medida que o vídeo avança, em direção ao final da linha do tempo.

Os traços verdes, que cortam a linha do tempo, indicam que houve uma troca de *slides* naquele momento. Portanto, quando o vídeo é iniciado e a linha vermelha (que indica o tempo atual da apresentação) cruza com uma linha verde, o *slide* é trocado automaticamente. Cada linha verde quando clicada, também serve como atalho para mover a apresentação para aquele determinado momento. Portanto, as linhas verdes podem servir para trocar os *slides*, movendo a apresentação caso o vídeo for iniciado, assim como os botões “Próximo” e “Anterior” da área de *slides*.

A.3.2.3. Vídeo

O vídeo, localizado abaixo do menu lateral esquerdo, possui três comandos básicos, que podem ser acionados a qualquer momento, representados pela Figura 49.

- *Back*: volta o vídeo para o início em qualquer momento da apresentação.
- *Play*: inicia o vídeo. Quando o vídeo é iniciado, o traço vermelho da linha do tempo começa a andar, indicando o tempo atual da apresentação. A partir daí, ocorre a troca automática de *slides*, a medida que o traço vermelho alcança os traços verdes dispostos na linha do tempo.

- *Pause*: pausa o vídeo, congelando o traço vermelho e a troca automática de *slides*. Deve-se utilizar o botão *play* para reiniciar o vídeo.

Após ser iniciado, o vídeo segue sincronamente com a linha do tempo e a troca de *slides*. Como explicado, caso haja uma intervenção manual do usuário, trocando o *slide*, o vídeo irá para o momento exato da troca do *slide* automaticamente.

A.3.3. Edição e remoção de reuniões já existentes

Uma caixa de seleção contendo todas as reuniões cadastradas no sistema será exibida, juntamente com os botões “Editar” e “Remover”. Para editar ou remover uma reunião é necessário que o usuário escolha uma das reuniões cadastradas e clique na opção desejada.

Caso o usuário escolha editar uma reunião, ele visualizará uma página semelhante à da Figura 57, porém os dados atuais da reunião escolhida já estarão preenchidos. Assim como no cadastro de reuniões, o usuário poderá editar as salas e participantes da reunião. Isso é feito na página seguinte, da mesma forma indicada na Figura 54. Todas as modificações realizadas serão gravadas no sistema.

Caso o usuário opte em remover uma reunião, ele visualizará uma página de confirmação de remoção. Esta página é exibida para evitar que dados sejam removidos por engano. Se o usuário realmente desejar remover a reunião do sistema, ele deverá clicar no botão “Remover”. A partir daí a reunião é excluída do sistema e todas as suas informações são apagadas da base de dados.

A.4. Seção Salas

A Seção Salas é a responsável por oferecer serviços referentes às salas do *eMeet/SERVE*, como, por exemplo, o cadastro e remoção de salas. As salas são utilizadas para indicar que participantes da reunião estiveram em um determinado local. Ao acessar a Seção Salas, o usuário visualizará a página inicial dessa seção, que conterá a lista de serviços oferecidos. Tal página está representada na Figura 60.

Conforme mostra a Figura 60, observa-se que o menu lateral esquerdo, o espaço de indicação *login/logout* e os ícones de ajuda continuam presentes.

Na Seção Salas os serviços oferecidos são o cadastro, a listagem, a edição e a remoção de salas, que são discutidos a seguir.

Figura 60: Página inicial da Seção Salas.

A.4.1. Cadastro de novas salas

Para cadastrar uma sala no sistema, o usuário deverá preencher um formulário contendo os seguintes campos, representados pela Figura 61:

Figura 61: Campos para cadastro de uma nova sala.

- Nome da sala: este campo deve conter o nome da sala. É por esse nome que a sala será identificada no sistema. Este é um campo obrigatório e deverá ser preenchido necessariamente.
- Endereço Completo: este campo deve conter o endereço da sala. Ele é utilizado somente para armazenar tal informação e seu preenchimento é opcional.

Após preencher tais campos, o usuário deverá clicar no botão “Criar” e a sala será cadastrada no sistema.

A.4.2. Listagem de salas cadastradas

O usuário poderá clicar nesta opção para ver uma listagem de todas as salas cadastradas. A página visualizada conterá uma lista completa com todas as salas cadastradas no sistema.

A.4.3. Edição e remoção de salas já existentes

Nesta página, uma caixa de seleção contendo todas as salas cadastradas no sistema será exibida, juntamente com os botões “Editar” e “Remover”. Para editar ou remover uma sala é necessário que o usuário escolha uma das salas cadastradas e clique na opção desejada.

Caso o usuário escolha editar uma sala, ele visualizará uma página semelhante à da Figura 61, porém os dados atuais da sala escolhida já estarão preenchidos. Ele poderá alterá-los como desejar e clicar no botão “Atualizar”. Todas as modificações realizadas serão gravadas no sistema.

Figura 62: Página de confirmação de remoção de uma reunião.

Caso o usuário escolha remover uma sala, ele visualizará uma página de confirmação de remoção, conforme mostra a Figura 62. Esta página é exibida para evitar que dados sejam removidos por engano. Se o usuário realmente desejar remover a sala do sistema, ele deverá clicar no botão “Remover”. A partir daí a sala é excluída do sistema e todas as suas informações são apagadas da base de dados.

A.5. Seção Usuários

A Seção Usuários possui uma integração direta com o servidor de usuários, que é responsável pelo gerenciamento de usuários nos sistemas como o *eMeet/SERVE*. Sendo assim, as páginas dessa seção do *eMeet/SERVE* são somente uma interface de acesso a alguns serviços oferecidos pelo servidor de usuários e, portanto, qualquer operação dentro dessa seção é feita pelo servidor de usuários.

A Seção Usuários é responsável por oferecer serviços referentes aos usuários do *eMeet/SERVE*, bem como de outros sistemas integrados, como, por exemplo, o cadastro e remoção de usuários.

Figura 63: Página inicial da Seção Usuários.

Ao acessar a Seção Usuários, o usuário visualizará a página inicial dessa seção, que conterá a lista de serviços oferecidos. Tal página está representada na Figura 63.

Na Seção Usuários os serviços oferecidos são o cadastro, a edição e a remoção de usuários, que são discutidos a seguir.

A.5.1. Cadastro de novos usuários

Para cadastrar um usuário no sistema, o usuário deverá preencher um formulário contendo os seguintes campos, representados pela Figura 64:

Figura 64: Campos para cadastro de um novo usuário.

- Nome do usuário: este campo deve conter o nome do usuário. É por esse nome que o usuário será identificado no sistema. Tal nome será listado como possíveis participantes das reuniões. Este é um campo obrigatório e deverá ser preenchido.
- *E-mail* do usuário: este campo deve conter o *e-mail* do novo usuário. Campo obrigatório.

- *Login* do usuário: este campo deve conter o *login* do usuário. O *login* será utilizado pelo usuário, juntamente com a senha, para que ele acesse partes restritas do sistema. Campo obrigatório.
- Senha do usuário: este campo deve conter a senha do usuário. A senha deverá ser uma cadeia secreta de caracteres alfanuméricos, de forma a autenticar o usuário durante o *login*. Campo obrigatório.
- Confirmação de senha: este campo deve conter a mesma cadeia de caracteres alfanuméricos do campo Senha do usuário. Ele é utilizado como validação do cadastro da senha. Campo obrigatório.

Após preencher tais campos, o usuário deverá clicar no botão “Criar” e o usuário será cadastrado no sistema, através da integração com o servidor de usuários.

A.5.2. Edição e remoção de usuários já existentes

Nesta página, uma caixa de seleção contendo todas os usuários cadastrados no sistema será exibida, juntamente com os botões “Editar” e “Remover”. Para editar ou remover um usuário, é necessário que o usuário escolha um dos usuários cadastrados e clique na opção desejada.

Caso a edição de usuário seja escolhida, uma página semelhante à da Figura 64, será exibida. Porém, todos os dados, com exceção da senha e sua confirmação, já estarão preenchidos. O usuário poderá alterar tais dados como desejar e clicar no botão “Atualizar”. Todas as modificações realizadas serão gravadas no sistema.

Caso a remoção de usuário seja escolhida, uma página de confirmação de remoção será exibida. Esta página é exibida para evitar que dados sejam removidos por engano. Se o usuário realmente desejar remover o usuário escolhido do sistema, ele deverá clicar no botão “Remover”. A partir daí o usuário é excluído do sistema e todas as suas informações são apagadas da base de dados.

A.6. Seção Fale Conosco

A Seção Fale Conosco é um canal de comunicação direto com os autores e mantenedores do *eMeet/SERVE*. Se o usuário deseja mandar alguma mensagem para os autores e mantenedores, bem como pedir o seu cadastramento, é nessa seção que o usuário deverá acessar.

Para enviar uma mensagem, preencha os campos indicados pela Figura 65, descritos a seguir:

- Nome: este campo deve conter o nome do usuário. Este campo é de preenchimento obrigatório.
- E-mail: este campo deve conter o e-mail do usuário. É por esse e-mail que os autores poderão entrar em contato com o usuário. Campo obrigatório.
- Comentário: este campo deve conter a mensagem que o usuário quer enviar aos autores do sistema.
- Assunto: o usuário deverá escolher um assunto que melhor se adeque a mensagem que será enviada.

O formulário possui um cabeçalho azul com o texto "Preencha corretamente os campos abaixo: ?". Abaixo dele, há quatro campos de entrada: "Nome:*" e "Email:*" são campos de texto curtos; "Comentário:" é um campo de texto grande com uma barra de rolagem; "Assunto:" é um menu suspenso com "Elogio" selecionado. Cada campo tem um ícone de ajuda (?) ao lado. Na base do formulário, há dois botões: "Limpar" e "Enviar".

Figura 65: Campos para envio de mensagens aos autores e mantenedores do *eMeet/SERVE*.

Após preencher os campos do formulário, o usuário deverá clicar no botão “Enviar” para mandar a sua mensagem aos autores do *eMeet/SERVE*.

Apêndice B. Código de geração das tabelas do MySQL

Para realizar a criação das tabelas do *eMeet/SERVE* no banco de dados MySQL, deve-se executar o código mostrado na Figura 66, após realizar a conexão e a criação da base de dados. Tais operações são realizadas com a execução dos comandos “mysql -u root -p”, “create database emeet” e “use emeet”, respectivamente.

```
create table meeting
(
  mtgcod      integer(8) not null  auto_increment,
  mtgnam      varchar(100) not null,
  mtgvid      varchar(40),
  mtgstadat   datetime,
  mtgenddat   datetime,
  grpcod      integer(8),
  mtgdes      varchar(255),
  primary key (mtgcod)
);

create table slide
(
  mtgcod      integer(8) not null,
  slicod      integer(8) not null,
  sligif      varchar(40),
  slitst      integer(12),
  primary key (mtgcod, slicod)
);

alter table slide add constraint fk_meeting_slide FOREIGN KEY (mtgcod) references meeting
(mtgcod);

create table penstroke
(
  mtgcod      integer(8) not null,
  slicod      integer(8) not null,
  pencod      integer(8) not null,
  pengif      varchar(100),
  pentst      integer(12),
  pentopx     integer(5),
  pentopy     integer(5),
  penbotx     integer(5),
  penboty     integer(5),
  primary key (mtgcod, slicod, pencod)
);

alter table penstroke add constraint fk_slide_penstroke FOREIGN KEY (mtgcod,slicod) references
slide (mtgcod,slicod);
```

```

create table room
(
    roocod integer(8) not null auto_increment,
    roonam varchar(100) not null,
    rooadd varchar(100),
    primary key (roocod)
);

create table meeting_room
(
    mtgcod integer(8) not null,
    roocod integer(8) not null,
    usrcod integer(8) not null,
    primary key (mtgcod, roocod, usrcod)
);

alter table meeting_room add constraint fk_room_mtgroom FOREIGN KEY (roocod) references
room (roocod);
alter table meeting_room add constraint fk_meeting_mtgroom FOREIGN KEY (mtgcod) references
meeting (mtgcod);

```

Figura 66: Código de criação das tabelas do *eMeet/SERVE* para MySQL.

O código de criação das tabelas do *eMeet/SERVE* para MySQL também se encontra em um arquivo físico, no diretório:

`emeet/sql/tables_emeet.sql`