Construção de aplicações de captura e acesso baseada em recorrência de funcionalidades

Renan Gonçalves Cattelan

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito: 19/02/2004

Assinatura: _____

Construção de aplicações de captura e acesso baseada em recorrência de funcionalidades¹

Renan Gonçalves Cattelan

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

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

**São Carlos
2004**

¹Trabalho realizado com auxílio financeiro da FAPESP (02/05677-4).

*Aos meus pais,
pelo apoio constante e incondicional.*

Agradecimentos

À minha orientadora, Graça Pimentel, por expandir meus horizontes.

A Laércio Baldochi e Khai Truong, pelas contribuições técnicas na definição da arquitetura.

Aos professores João Batista e Paulo Cesar Masiero, pelas críticas e sugestões apresentadas no exame de qualificação.

Aos colegas do Laboratório Intermídia.

À minha família, pelo carinho e compreensão.

À FAPESP, pelo apoio financeiro (02/05677-4).

Resumo

Aplicações de captura e acesso exploram o paradigma de computação ubíqua — que consiste em popular o ambiente com aplicações e dispositivos computacionais a fim de auxiliar transparentemente as pessoas na realização de suas atividades — para dar apoio à captura automática de informação em experiências “ao vivo” e à correspondente geração de documentos passíveis de armazenamento, recuperação, visualização e extensão ao longo do tempo. Devido à sua natureza distribuída, à heterogeneidade dos dispositivos computacionais envolvidos e à diversidade nas funcionalidades providas, essas aplicações são difíceis de se construir e requerem infra-estruturas e serviços de software que auxiliem o desenvolvedor nessa tarefa. Este trabalho investiga a construção e o uso de aplicações de captura e acesso por meio do desenvolvimento da xINCA, uma infra-estrutura estendida baseada em componentes de software reutilizáveis que englobam as funcionalidades recorrentes nessa classe de aplicações. A xINCA é uma extensão da infra-estrutura INCA — uma infra-estrutura de baixo nível que provê abstrações de comunicação para aplicações de captura e acesso. Complementares, as infra-estruturas INCA e xINCA provêem um modelo simplificado para o desenvolvimento de aplicações de captura e acesso, considerando aspectos de projeto, implementação e reuso. Associada ao modelo de armazenamento do serviço StRES, a xINCA tem ainda papel na estruturação da informação capturada com o uso de XML e tecnologias correlatas.

Palavras-chave: Aplicações de captura e acesso, Computação Ubíqua.

Abstract

Capture and access applications explore the ubiquitous computing paradigm — that basically consists on populating the environment with computational devices and applications to transparently assist people on their activities — to support the automatic capture of information on live experiences and generate corresponding documents that can be further stored, retrieved, visualized and extended. Due to their distributed nature, heterogeneity of involved devices and variety of provided functionalities, capture and access applications are hard to build and require software infrastructures and services that help the developer out on his task. This work investigates the building and use of capture and access applications through the development of xINCA, a component-based software infrastructure comprising the most common functionalities present in such a class of applications. xINCA is an extension of INCA — a low-level infrastructure that provides communication abstractions for capture and access applications. Complementary, INCA and xINCA provide a simplified model for the development of capture and access applications, concerning design, implementation and reuse aspects. Allied with the StRES storage model, xINCA also plays an important role on the structuring of captured information by using XML and corresponding technologies.

Keywords: Capture and Access Applications, Ubiquitous Computing.

Lista de Figuras

2.1	Dispositivos de computação ubíqua	7
3.1	Captura e acesso no domínio de sala de aula	15
3.2	Sala instrumentada usada no LiteMinutes	16
4.1	Possíveis configurações do Registry	22
4.2	Camadas da arquitetura xINCA	24
4.3	Componente xINCA de <i>whiteboard</i>	25
4.4	Componente xINCA de texto	26
4.5	Fluxo de comunicação entre componentes xINCA	29
4.6	Diagrama de classes UML do componente xINCA de <i>web logging</i> e de seu modelo de dados.	31
4.7	O modelo de aplicação xINCA-StRES	34
5.1	Arquitetura do sistema iClass	38
5.2	Formulário de opção por reuso	39
5.3	Formulário de criação e configuração de aula sem reuso	39
5.4	Formulário de criação e configuração de aula com reuso	40
5.5	Listagem das aulas pré-preparadas	40
5.6	Seleção de funcionalidades no iClass	41
5.7	Captura de aula em andamento	43
5.8	Documento XML do iClass	43
5.9	Interface gráfica de acesso do iClass	45
5.10	Interface de apresentação SMIL	45
5.11	Interface de criação e configuração de uma nova aplicação	47

5.12	<i>Front-end</i> para aplicação com combinação texto + <i>whiteboard</i>	48
5.13	Possível interface de acesso para uma sessão capturada	48
6.1	Integração entre mídias no LiteMinutes	52
A.1	Diagrama de classes UML do componente xINCA de áudio.	68
A.2	Diagrama de classes UML do modelo de dados do componente xINCA de áudio.	69
A.3	Diagrama de classes UML do componente xINCA de texto.	70
A.4	Diagrama de classes UML do modelo de dados do componente xINCA de texto.	70
A.5	Diagrama de classes UML do componente xINCA de vídeo.	71
A.6	Diagrama de classes UML do modelo de dados do componente xINCA de vídeo.	72
A.7	Diagrama de classes UML do componente xINCA de <i>web logging</i>	72
A.8	Diagrama de classes UML do modelo de dados do componente xINCA de <i>web logging</i>	73
A.9	Diagrama de classes UML do componente xINCA de <i>whiteboard</i>	74
A.10	Diagrama de classes UML do modelo de dados do componente xINCA de <i>whiteboard</i>	75

Lista de Tabelas

2.1	Eras da computação	6
4.1	Funcionalidades presentes em aplicações de C&A.	23
5.1	Comparação entre os sistemas eClass e iClass.	45

Sumário

1	Introdução	1
1.1	Contextualização	1
1.2	Motivação	2
1.3	Objetivos	2
1.4	Estrutura da Dissertação	3
2	Computação Ubíqua	5
2.1	Considerações Iniciais	5
2.2	Dispositivos de Computação Ubíqua	6
2.3	Aplicações de Computação Ubíqua	6
2.3.1	Interfaces Naturais	8
2.3.2	Captura e Acesso de Atividades Humanas	8
2.3.3	Computação Ciente de Contexto	9
2.4	Considerações Finais	10
3	Aplicações de Captura e Acesso	11
3.1	Considerações Iniciais	11
3.2	Características de Aplicações de Captura e Acesso	12
3.2.1	Domínio de Aplicação	12
3.2.2	Heterogeneidade de Dispositivos	12
3.2.3	Espaço de Projeto	12
3.2.4	Estruturação do Problema de Captura e Acesso	13
3.2.5	Acesso	13
3.2.6	Captura Pública e Privada	14

3.3	Ambientes que Suportam a Captura de Sessões	14
3.3.1	Captura e Acesso em Sala de Aula	14
3.3.2	Captura e Acesso em Reuniões	15
3.3.3	Captura e Acesso em Outras Áreas	16
3.4	Considerações Finais	17
4	Infra-estruturas e Serviços para Aplicações de Captura e Acesso	19
4.1	Considerações Iniciais	19
4.2	O Projeto InCA-SERVE	20
4.3	INCA	21
4.3.1	Abstração de Comunicação	21
4.3.2	Abstração de Informação	22
4.4	Estendendo a INCA: xINCA	22
4.4.1	Componentes xINCA	24
4.4.2	Funcionamento dos Componentes	28
4.4.3	Criação de Novos Componentes	32
4.4.4	Algumas Observações Sobre o Uso da INCA	32
4.5	Automatizando o Armazenamento: a Abordagem xINCA-StRES	33
4.6	Considerações Finais	36
5	Aplicações Desenvolvidas	37
5.1	Considerações Iniciais	37
5.2	iClass	37
5.2.1	Pré-produção	38
5.2.2	Gravação “ao vivo”	41
5.2.3	Pós-produção	42
5.2.4	Acesso	44
5.2.5	Comparação eClass — iClass	44
5.3	CAT	46
5.4	Sobre a Avaliação de Infra-estruturas	47
5.5	Considerações Finais	49
6	Trabalhos Relacionados	51
6.1	Considerações Iniciais	51
6.2	LiteMinutes	52
6.3	JETS	53

6.4	Habanero	53
6.5	EasyLiving	53
6.6	Gaia	54
6.7	Aura	54
6.8	Interactive Workspaces	55
6.9	Roomware	55
6.10	Considerações Finais	55
7	Conclusões	57
7.1	Considerações Iniciais	57
7.2	Resultados e Contribuições	58
7.3	Limitações	59
7.4	Trabalhos Futuros	59
	Referências	60
A	Diagramas de Classes UML da xINCA	67
B	Esquemas XML dos Componentes xINCA	77
B.1	Esquema XML do Componente xINCA de Áudio	77
B.2	Esquema XML do Componente xINCA de Texto	77
B.3	Esquema XML do Componente xINCA de Vídeo	78
B.4	Esquema XML do Componente xINCA de <i>Web Logging</i>	78
B.5	Esquema XML do Componente xINCA de <i>Whiteboard</i>	79

Introdução

1.1 Contextualização

Experiências envolvendo interação entre humanos baseiam-se, em sua maior parte, na troca de informação. Devido a isso, as pessoas freqüentemente empregam tempo e esforço para registrar o conteúdo dessas experiências. Parte desse conteúdo pode ser capturada de modo automático, produzindo artefatos que possibilitem a reconstituição da experiência e permitindo que as pessoas concentrem sua atenção na experiência em si, sem que precisem se preocupar com a tarefa de registrar informação. Ambientes automatizados de captura exploram o paradigma de computação ubíqua — que consiste em popular o ambiente com aplicações e dispositivos computacionais a fim de auxiliar transparentemente as pessoas na realização de suas atividades (Weiser, 1991) — para dar apoio à *captura* automática de informação em experiências “ao vivo” e à correspondente geração de documentos passíveis de *armazenamento*, *recuperação*, *visualização* e *extensão* ao longo do tempo.

Um cenário possível para aplicações de captura e acesso (C&A) é o de sala de aula, em que são *capturados* e *armazenados* o áudio e as anotações realizadas pelo professor sobre uma lousa eletrônica (*eletronic whiteboard* ou, simplesmente, *whiteboard*). *Recuperação* e *visualização* ocorrem à medida que a informação capturada é posteriormente acessada pelas partes interessadas (pelos alunos para estudo, por exemplo). *Extensão* pode envolver desde o reuso da informação capturada para produção de novos artefatos até a criação de ligações (*links*) entre documentos correlatos.

As pesquisas do Projeto InCA-SERVE¹ têm investigado os problemas envolvidos no desenvolvimento e no uso de aplicações de C&A à informação hipermídia evolucionária, que explora o relacionamento entre atividades realizadas em sala de aula, na forma de multimídia, com espaços de discussão colaborativos (Pimentel et al., 2001b). Nesse sentido, vem sendo desenvolvido um conjunto de infra-estruturas e serviços computacionais com o objetivo de facilitar a construção de aplicações de C&A.

1.2 Motivação

A literatura tem reportado investigações relativas à construção de aplicações de C&A. Pesquisadores do Georgia Institute of Technology (Georgia Tech) têm dado ênfase aos aspectos arquiteturais e de projeto envolvidos na construção de aplicações de C&A, basicamente provendo abstrações ao nível de rede que tornam transparente a comunicação entre aplicações (Abowd, 1999b; Truong & Abowd, 2004). Trabalhos realizados no FX Palo Alto Laboratory têm investigado o uso de arquiteturas modulares na combinação de diferentes fluxos de mídia (Chiu et al., 2001).

A *motivação* para este trabalho reside no fato de que a construção de aplicações de C&A tem sido feita, geralmente, desconsiderando-se similaridades arquiteturais e aspectos de extensão e reuso. No escopo do Projeto InCA-SERVE, foi desenvolvida a infra-estrutura INCA, que provê um modelo simplificado para o projeto, a construção e a evolução de aplicações de C&A segundo a óptica arquitetural (Truong & Abowd, 2004). Entretanto, há demanda por um mecanismo que trate a recorrência de funcionalidades. Muitas das funcionalidades existentes nessa classe de aplicações são recorrentes — por exemplo: *whiteboard*, captura de texto, captura de áudio, captura de vídeo e captura de histórico de navegação hipertexto (*web logging*) — e o reuso de soluções eficazes para as mesmas deve ser estimulado.

1.3 Objetivos

O *objetivo* deste trabalho é investigar a construção e o uso de aplicações de C&A através da extensão da infra-estrutura INCA para inclusão de um conjunto de componentes de software reutilizáveis que implementem as funcionalidades recorrentes existentes em um grande número de aplicações de C&A, dando origem à infra-estrutura xINCA. Complementares, as infra-estruturas INCA e xINCA provêm um modelo simplificado para o desenvolvimento de

¹O InCA-SERVE é um projeto da área de computação ubíqua, fruto de uma parceria entre o Georgia Tech e o ICMC-USP (<http://coweb.icmc.usp.br/incaserve>).

aplicações de C&A, considerando aspectos de projeto, implementação e reuso. Associada ao modelo de armazenamento do serviço StRES, a xINCA tem ainda papel na estruturação da informação capturada com o uso de XML e tecnologias correlatas.

1.4 Estrutura da Dissertação

O restante do texto encontra-se estruturado da seguinte maneira: no Capítulo 2 são abordados os principais conceitos relacionados à área de computação ubíqua; no Capítulo 3 são caracterizadas as aplicações de C&A e apresentados exemplos reportados na literatura; no Capítulo 4 são detalhadas as infra-estruturas e serviços desenvolvidos no contexto do Projeto InCA-SERVE, sendo a xINCA uma delas; no Capítulo 5 é mostrado como a abordagem desenvolvida foi aplicada no desenvolvimento de duas aplicações de C&A — iClass e CAT; no Capítulo 6 é feito um levantamento dos principais trabalhos relacionados; e, finalmente, no Capítulo 7 são apresentadas as conclusões, os resultados e as contribuições obtidas.

Computação Ubíqua

2.1 Considerações Iniciais

A evolução da computação e o surgimento de novas tecnologias — *timesharing*, microprocessadores, redes, sistemas distribuídos, interfaces gráficas, Internet, etc. — têm alterado a dinâmica da vida cotidiana das pessoas e apresentado-lhes novos valores. Não é a tecnologia em si que realmente importa, mas sim a maneira com que as pessoas se relacionam com ela. (Weiser, 1991) vislumbrou uma forma pouco intrusiva de fazer uso do computador no cotidiano das pessoas. Segundo ele, facilidades computacionais devem ser incorporadas ao ambiente a fim de auxiliar atividades humanas mudando minimamente a forma como tais atividades são realizadas. Essa forma transparente de integrar tecnologia às atividades diárias das pessoas foi denominada computação ubíqua.

(Weiser & Brown, 1996) dividem os últimos cinquenta anos da computação nas eras do *mainframe* e do computador pessoal. A era do *mainframe* é caracterizada por muitas pessoas compartilhando um único e raro recurso computacional. Na era da computação pessoal, essa relação muda para um computador para cada pessoa. A Internet e a difusão da computação distribuída representam uma fase de transição, agregando componentes das eras do *mainframe* e da computação pessoal através do uso massivo do modelo cliente-servidor, em que os computadores pessoais são os clientes *Web* e os *mainframes*, os servidores *Web*. Essa transição conduziria a uma terceira era: a da computação ubíqua, com vários computadores disponíveis para cada pessoa. Esses computadores estariam embutidos em praticamente tudo (nas paredes,

na mobília, nas roupas das pessoas, etc.) e, principalmente, estariam interligados a uma rede tal como a Internet, possibilitando, assim, as mais diversas formas de utilização. A Tabela 2.1 ilustra a relação entre o número de pessoas e o número de computadores para cada uma das eras da computação.

Tabela 2.1: Relação entre o número de pessoas e de computadores para cada uma das eras da computação (Weiser & Brown, 1996).

Era	Pessoa(s)	Computador(es)
<i>Mainframe</i>	n	1
Computação pessoal	1	1
Internet / Comp. distribuída	... transição ...	
Computação ubíqua	1	n

Neste capítulo são apresentados os principais conceitos relacionados à área de computação ubíqua, são explorados aspectos gerais da área, como dispositivos e aplicações computacionais, e é dada ênfase aos três temas clássicos que concentram seu foco de pesquisa. Isso é feito nas Seções 2.2 e 2.3. Na Seção 2.4 são apresentadas as considerações finais sobre a área e sua importância no contexto deste trabalho.

2.2 Dispositivos de Computação Ubíqua

As visões de (Weiser, 1991) previam, para um futuro próximo, a proliferação de dispositivos computacionais ubíquos de diversos tamanhos: pequenos e pessoais (*inch-scale*), de médio porte (*foot-scale*) e grandes e de uso coletivo (*yard-scale*). De fato, isso acabou acontecendo: dispositivos pessoais de pequeno e médio porte tais como *pads* (Figura 2.1(a)), *personal digital assistants* (PDA's), *tablets* digitais (Figura 2.1(b)) e *laptops* tornaram-se comuns no final da década de 90. Da mesma forma, dispositivos maiores, como lousas eletrônicas (Figura 2.1(c)), passaram a fazer parte de ambientes de uso comum, tais como salas de reuniões, salas de aula e laboratórios. Além desses, hoje em dia, faz-se uso de dispositivos que extrapolam as escalas inicialmente previstas por (Weiser, 1991). São os chamados *wall-sized*, incluindo, por exemplo, paredes inteiras cujas superfícies são monitoradas por *mimios* (Figura 2.1(d)).

2.3 Aplicações de Computação Ubíqua

Para explorar o uso desses novos dispositivos, surgiram novas aplicações computacionais, cujo desenvolvimento está diretamente associado a três temas, que concentram o foco de

2.3 Aplicações de Computação Ubíqua

(a) *Pad* (<http://nano.xerox.com/parctab>).

(b) *Tablet* (<http://www.fxpal.com/xlibris>).

(c) *Lousa eletrônica* (<http://www.mimio.com/meetings>).

(d) *Mimio* (<http://www.mimio.com/meet/xi/>).

Figura 2.1: Alguns dispositivos de computação ubíqua.

pesquisa da área de computação ubíqua: interfaces naturais, captura e acesso de atividades humanas e computação ciente de contexto (Abowd & Mynatt, 2000).

2.3.1 Interfaces Naturais

A computação ubíqua inspira um certo deslocamento da metáfora tradicional de *desktop* (teclado + *mouse* + *display*) para uma forma de interação mais parecida com o modo com o qual os humanos interagem com o mundo físico. Humanos falam, gesticulam e escrevem para se comunicarem uns com os outros e tudo isso poderia ser usado, implícita ou explicitamente, como entrada para sistemas de computação ubíqua.

Aqui, obviamente, existe uma grande intersecção com a área de interação humano-computador, cuja comunidade científica vem há anos trabalhando no desenvolvimento de interfaces cada vez mais amigáveis. Resultados desses esforços incluem paradigmas de interação baseados em voz (Hauptmann, 1995), em escrita manual (Schilit et al., 1998), na manipulação de artefatos eletrônicos através do uso de objetos do mundo físico, criando o conceito de interfaces tangíveis (Ishii & Ullmer, 1997), e no uso de sensores em dispositivos computacionais, de forma a permitir que a manipulação física desses dispositivos seja corretamente interpretada pelas aplicações que os controlam (Harrison et al., 1998). Trabalhos mais recentes exploram o uso de *ambient displays*, dispositivos abstratos retratando informação periférica ao foco de atenção do usuário (Heiner et al., 1999; Mankoff et al., 2003).

Esses novos tipos de interação trazem novas necessidades e desafios. Novos tipos primitivos de dados (áudio, vídeo, *ink*¹ e informação sensorial) passam a ser necessários para o desenvolvimento eficaz de interfaces naturais. Além disso, são necessários mecanismos para o tratamento e reconhecimento de erros associados às novas interfaces (Abowd & Mynatt, 2000).

2.3.2 Captura e Acesso de Atividades Humanas

As pessoas gastam muito tempo registrando, com maior ou menor precisão, os eventos que as cercam. Tudo para que, no caso de uma necessidade, possam recuperar determinados detalhes de acontecimentos passados. Uma das utilidades potenciais de um ambiente de computação ubíqua é que ele pode ser usado para capturar as experiências cotidianas das pessoas e tornar esses registros disponíveis para uso futuro (Abowd, 1999b). A proposta é utilizar ferramentas especializadas na tarefa de registrar informação — os computadores — e permitir que as pessoas se concentrem na síntese e na compreensão da experiência propriamente dita, com total

¹*Ink* é a tinta virtual presente nos traços esboçados em uma lousa eletrônica, ou em um *tablet*. Assim sendo, trata-se de uma forma de entrada de dados nesses tipos de interface.

confiança de que os detalhes estão sendo registrados e serão disponibilizados para consultas futuras.

Essas experiências podem ser vistas como geradoras de rico conteúdo multimídia. Prover mecanismos automatizados para a captura, integração e acesso desses registros multimídia é um dos desafios da computação ubíqua.

Por serem um tema central neste trabalho, aplicações de captura e acesso (C&A) são discutidas com mais detalhes no Capítulo 3.

2.3.3 Computação Ciente de Contexto

De acordo com (Dey, 2001), informação de contexto² é aquela que pode ser utilizada para caracterizar a situação de uma entidade. Uma entidade pode ser uma pessoa, um lugar, ou um objeto considerados relevantes para a interação entre o usuário e a aplicação. Um sistema é considerado ciente de contexto se usar informação de contexto para prover informação ou serviços relevantes para o usuário de acordo com a tarefa que o mesmo deseja realizar.

Computação ciente de contexto envolve o desenvolvimento de aplicações que permitam a coleta de contexto e a mudança de comportamento com base nas informações de contexto coletadas. Além disso, uma aplicação ciente de contexto deve ser capaz de associar significado aos eventos do mundo exterior e usar essa informação de maneira efetiva (Abowd, 1999b).

As aplicações de computação ubíqua precisam ter ciência de contexto, de modo a serem capazes de adaptar seu comportamento com base nas informações extraídas dos ambientes físico e computacional. Muitas aplicações utilizam informação de contexto de uma maneira simplificada (basicamente identificação de usuário e local), mas existem também desafios referentes à criação de mecanismos mais complexos para sua representação, como as dimensões de projeto tratadas na Seção 3.2.3.

(Roman et al., 2002a) estendem o conceito para o cenário de redes *ad hoc* formadas por numerosos dispositivos móveis que interagem uns com os outros oportunisticamente por meio de conexões sem fio intermitentes. Um método de especificação permite à aplicação definir um contexto individualizado que se estende por muitos dispositivos na rede *ad hoc*. Um protocolo de manutenção de contexto delega a contínua reavaliação do contexto e das operações sobre o mesmo a uma camada de *middleware* operando abaixo do nível de aplicação.

(Griswold et al., 2003) descrevem uma arquitetura de software que provê capacidade de extensão e integração de serviços em uma infra-estrutura para aplicações cientes de contexto. Como resultado, obteve-se uma ramificação intuitiva de organizações de classe, inter e intra-

²Optou-se por fazer a tradução do termo *context*, usado pelo autor original, como informação de contexto.

camadas, com o objetivo de especializar, organizar e compor mecanismos que tratam da construção de aplicações cientes de contexto.

2.4 *Considerações Finais*

Neste capítulo foi realizada uma contextualização da área de computação ubíqua, relatando seu surgimento e apresentando os principais conceitos envolvidos. Foram classificados e exemplificados os dispositivos computacionais mais comumente utilizados em pesquisas da área. Com relação às aplicações de computação ubíqua, foram abordados os três temas principais: interfaces naturais, captura e acesso de atividades humanas e computação ciente de contexto. Apesar de possuírem características distintas, esses temas não são tratados de maneira isolada, estando diretamente relacionados e havendo significativa sinergia entre eles.

O segundo tema — captura e acesso de atividades humanas — concentra o foco de interesse deste trabalho por englobar o desenvolvimento de aplicações de C&A, tratadas com mais detalhes no Capítulo 3.

Aplicações de Captura e Acesso

3.1 Considerações Iniciais

(Truong et al., 2001) definem *captura e acesso* como sendo as tarefas de preservar um registro de alguma experiência “ao vivo” (sessão) e de permitir que o mesmo possa ser revisto em algum ponto no futuro. *Captura* ocorre quando uma entidade (dispositivo, programa, etc.) gera um artefato que documenta o histórico do que aconteceu. Esse artefato, ou dado capturado, é gravado como um fluxo (*stream*) de informação indexado ao longo do tempo. *Acesso* é definido como o ato de revisitar um artefato capturado decorrido um certo intervalo de tempo após a etapa de captura. Esse intervalo de tempo pode variar de milissegundos (acesso síncrono, ou instantâneo) a anos (acesso a longo termo).

Aplicações de captura e acesso (C&A) automatizam o processo de captura das atividades humanas do cotidiano, permitindo a geração automática de documentos hipermídia passíveis de armazenamento, recuperação e apresentação ao longo do tempo.

Este capítulo tem o propósito de apresentar uma visão geral das aplicações de C&A, caracterizando-as e definindo os conceitos-chave envolvidos. Na Seção 3.2 são apresentadas as características principais a serem consideradas na construção de aplicações de C&A; na Seção 3.3 são descritos alguns exemplos de aplicações de C&A; e, por fim, na Seção 3.4 são apresentadas as considerações finais sobre o tema e seu relacionamento com o trabalho realizado.

3.2 Características de Aplicações de Captura e Acesso

3.2.1 Domínio de Aplicação

O domínio de aplicação de ambientes automatizados de captura e acesso pode variar desde reuniões informais, sem planejamento prévio, como, por exemplo, encontros casuais, até experiências mais formais, com programação e periodicidade pré-determinadas, como, por exemplo, cursos tradicionais ou reuniões de negócios.

3.2.2 Heterogeneidade de Dispositivos

Em um ambiente instrumentado, qualquer dispositivo, aplicação ou processo que colete dados do ambiente pode ser visto como uma entidade de captura. Desse modo, a etapa de captura em si ocorre em uma espécie de eixo vertical, envolvendo desde dispositivos de hardware, como microfones e câmeras, até sofisticados componentes de software. Essa característica é particularmente relevante para o desenvolvimento de infra-estruturas de *middleware* e serviços para aplicações de C&A, em que uma tentativa de se cobrir todos os níveis resultará, muito provavelmente, em uma solução complexa e difícil de se usar. Nesses casos, pode-se optar por uma abordagem em camadas, com vários níveis provendo abstrações uns para os outros.

3.2.3 Espaço de Projeto

Outro aspecto importante na construção de aplicações de C&A refere-se ao espaço de projeto. (Truong et al., 2001) definem cinco dimensões para o projeto de aplicações de C&A:

- *Quem* são os usuários durante as fases de captura e acesso? É importante identificar quantos usuários estão envolvidos, quais os seus papéis no domínio da aplicação e se os registros capturados são públicos, privados ou uma mistura de ambos;
- *O que* é capturado e acessado? Define-se a experiência em termos dos artefatos manipulados e dos fluxos de informação gerados, quais são importantes para consulta futura e em que nível de granulosidade;
- *Quando* a captura e o acesso ocorrem? Deve-se identificar com que frequência a captura e o acesso ocorrem, se há algum padrão passível de previsão e o tempo que se passará entre a captura da experiência e seus acessos;
- *Onde* a captura e o acesso ocorrem? São considerados aspectos de localização física e de mobilidade durante a captura e o acesso;

3.2 Características de Aplicações de Captura e Acesso

- *Como* são feitos a captura e o acesso? Identificam-se quais dispositivos e ferramentas são necessários para a fase de captura e para a fase de acesso.

Essas dimensões apresentam relacionamento direto com o tema ciência de contexto, provendo uma base para o armazenamento de informação de contexto e permitindo, através de combinações dessa informação, eventuais inferências sobre o estado da aplicação e adaptações em seu comportamento.

3.2.4 Estruturação do Problema de Captura e Acesso

(Abowd et al., 1996) sugerem estruturar o problema de captura e acesso em quatro fases:

- *Pré-produção*: consistindo da preparação de materiais para a sessão de captura;
- *Gravação “ao vivo”*: sincronização e captura dos fluxos de informação relevantes;
- *Pós-produção*: integração dos fluxos de informação capturados;
- *Acesso*: visualização, pelos usuários-finais, da informação capturada.

Outros autores utilizam essa estruturação com algumas adaptações. (Richter et al., 2001) propõem o tratamento da segunda e da terceira fases como se fossem uma única, sem nítida distinção dos processos de captura, sincronização e integração dos fluxos de mídia. (Pimentel et al., 2001b) agregam ainda o que pode ser considerada uma quinta fase, a de *extensão*, explorando o conceito de informação hipermídia evolucionária, permitindo aos usuários dar continuidade ao processo de captura por meio de interação, complemento e inter-relacionamento das informações capturadas e visualizadas.

3.2.5 Acesso

Acesso pode ser pensado como recuperação e visualização de informação capturada, exercendo diferentes expectativas sobre os usuários. Por exemplo, supondo-se uma aplicação para a captura de aulas, um professor pode estar interessado em visualizar um resumo do conteúdo da última aula que ministrou, com o intuito de preparar a seguinte, enquanto que um aluno, por outro lado, pode estar interessado nos detalhes da aula para estudar para uma prova que se aproxima. A interface de acesso apresenta outra importante característica, geralmente servindo de ponto de partida para reuso e extensão de informação capturada.

3.2.6 Captura Pública e Privada

A captura de registros para acesso futuro pode ocorrer em dois níveis: acesso público, cujo conteúdo capturado consiste de artefatos públicos — criados por membros de um grupo, sendo de propriedade de e podendo ser acessado por todos os membros do grupo — e acesso privado, cujo conteúdo capturado consiste de artefatos privados — criados, manipulados e pertencentes a somente uma pessoa (Greenberg et al., 1999). No segundo caso, a informação pode interessar somente ao usuário, envolvendo aspectos de privacidade e segurança da informação.

3.3 Ambientes que Suportam a Captura de Sessões

Sistemas baseados na captura de experiências têm sido investigados em diversos contextos, sendo que dois deles concentram a grande maioria das aplicações desenvolvidas: o domínio de sala de aula e o de reuniões. Nesta seção é apresentado um levantamento dos principais exemplos de aplicações de C&A reportados na literatura. A classificação dos sistemas é baseada em critérios elaborados por (Truong & Abowd, 2004), segundo o domínio de uso.

3.3.1 Captura e Acesso em Sala de Aula

Aplicações de C&A para o domínio de sala de aula permitem a captura automática das atividades conduzidas por professores e alunos durante uma aula, geralmente realizada em um ambiente instrumentado com dispositivos de captura tais como microfones, lousas eletrônicas, *tablets*, *mímios*, câmeras e projetores (vide Figura 3.1(a)). O conteúdo capturado geralmente inclui os *slides* apresentados pelo professor e as anotações sobre os mesmos, áudio, vídeo, URL dos sites visitados durante a aula, e execuções arbitrárias de programas. O acesso a tal conteúdo é provido tipicamente por meio de uma interface de usuário que integra os diferentes fluxos de mídia capturados (vide Figura 3.1(b)).

Na literatura, é possível encontrar vários exemplos de sistemas bem sucedidos. Por exemplo, o Cornell Lecture Browser (Mukhopadhyay & Smith, 1999) e o eClass (Abowd, 1999a) permitem a captura de uma aula tradicional para uma audiência em um auditório tradicional. Já o AutoAuditorium (Bianchi, 1998) e o Smart Classroom (Shi et al., 2003) têm recursos para a transmissão ao vivo de uma aula convencional a uma audiência remota. Há também exemplos de aplicações comerciais como o Silicon Chalk¹ e os softwares distribuídos com *mímios* e lousas eletrônicas SmartBoardTM.

¹<http://www.silicon-chalk.com>.

3.3 Ambientes que Suportam a Captura de Sessões

(a) Sala de aula instrumentada utilizada no Sistema eClass.

(b) Hiperdocumento gerado automaticamente pelo Sistema eClass.

Figura 3.1: Captura e acesso no domínio de sala de aula (Pimentel et al., 2000).

Outros sistemas, como o DEBBIE (Berque, 1999) e o StuPad (Truong et al., 1999), foram criados para explorar o uso de dispositivos pessoais tais como *tablets* e PDAs, permitindo aos alunos fazerem suas próprias anotações com base no conteúdo apresentado pelo professor.

Alguns sistemas se baseiam na captura de informações em contextos diferentes de apoio ao ensino. Um exemplo é o Smart Kindergarten (Chen et al., 2002), no qual um ambiente de pré-escola é instrumentado para coletar informações sobre as crianças, os professores e os objetos com os quais eles interagem. Projetado para coletar, gerenciar e combinar dados obtidos pelos sensores, o objetivo do projeto é auxiliar educadores a investigar com profundidade o aprendizado dos estudantes. Outro exemplo é o The Walden System (White et al., 2003), construído para facilitar a coleta de dados, por avaliadores, sobre o comportamento de crianças autistas, com o objetivo de observar seu desenvolvimento em termos da interação com outras crianças e com professores.

3.3.2 Captura e Acesso em Reuniões

O domínio de reuniões também tem sido alvo de pesquisas na área de captura e acesso. Além de disporem de ambientes instrumentados (vide Figura 3.2), geralmente permitem formas mais colaborativas de interação entre seus participantes, com o uso de dispositivos pessoais, como *laptops*, e de uso compartilhado, como lousas eletrônicas. Trabalho pioneiro no contexto de captura de reuniões tradicionais foi desenvolvido pelo Xerox PARC (Minneman et al., 1995). Igualmente pioneiros, o DOLPHIN (Streitz et al., 1994) e o Tivoli (Pedersen et al., 1993) foram desenvolvidos para dar suporte a reuniões distribuídas em que usuários podem interagir com uma lousa eletrônica compartilhada. O Dynamite (Wilcox et al., 1997) permite a integração de

anotações com áudio e vídeo capturado durante a reunião. O sistema DUMMBO (Brotherton et al., 1999) tem como foco a captura de sessões casuais, cujas ocorrências são normalmente aperiódicas e dependem de situações oportunas. O LiteMinutes (Chiu et al., 2001) é um sistema baseado em Internet para resumo de reuniões que permite a captura de anotações, *whiteboard* e vídeo. O TeamRoom (Roseman & Greenberg, 1996) é um sistema de *groupware* que suporta trabalho colaborativo presencial e a distância, servindo também como repositório para os documentos e artefatos manipulados. O TeamSpace (Richter et al., 2001) integra captura de reuniões a um ambiente de trabalho colaborativo para gerenciamento de processos de trabalhos compartilhados entre equipes de diferentes companhias.

Figura 3.2: Sala instrumentada usada no LiteMinutes (Chiu et al., 2001).

3.3.3 Captura e Acesso em Outras Áreas

Aplicações de C&A não se restringem aos domínios de sala de aula e de reuniões e, embora menos numerosos, existem exemplos de uso em outras áreas, como conferências e museus. O Conference Assistant (Dey et al., 1999) permite que um usuário, fazendo uso de um dispositivo móvel pessoal, tenha suas anotações capturadas durante uma conferência. As anotações pessoais do usuário são posteriormente integradas, com base em informação de localização física do usuário, provida por sensores, com informação publicamente capturada sobre a conferência. O HP Rememberer (Fleck et al., 2002) é um sistema de autoria no qual o visitante de um museu pode gerar um página *Web* documentando sua experiência, com parte do conteúdo sendo provida manualmente pelo usuário e parte obtida a partir de sensores e câmeras espalhados pelo salão de exibição. O Flatland (Mynatt et al., 1999) investiga o uso a longo termo de interfaces para lousas eletrônicas em escritórios, envolvendo técnicas para o gerenciamento do espaço de edição da lousa, semântica da aplicação e histórico das ações executadas na lousa. O

3.4 Considerações Finais

Ambulant Annotator (Bulterman, 2003) é um ambiente interativo para o domínio de sistemas de informação médica que permite a criação e visualização de anotações em fichas de pacientes. Aspectos considerados nesse sistema tratam do processo de engenharia de documentos e da representação das anotações sobre SMIL.

3.4 Considerações Finais

O objetivo central deste trabalho é facilitar a construção de aplicações de C&A através do desenvolvimento da xINCA, detalhada na Seção 4.4. Este capítulo apresentou uma visão geral das aplicações de C&A. Foram abordadas suas principais características, bem como uma série de exemplos reportados na literatura, dos clássicos ao mais atuais.

Infra-estruturas e Serviços para Aplicações de Captura e Acesso

4.1 Considerações Iniciais

Embora existam muitas demonstrações de aplicações de captura e acesso (C&A), construí-las e mantê-las não é uma tarefa trivial. Constituídas de uma confederação de componentes que precisam trabalhar juntos de maneira harmoniosa, as aplicações de C&A geralmente consomem muito tempo de implementação e requerem familiaridade em uma grande variedade de domínios de problema. Sistemas distribuídos, sistemas hipermídia, programação concorrente e computação móvel, para citar uns poucos, são exemplos.

A despeito de suas complexidades e desconsiderando similaridades de projeto, aplicações de C&A têm sido construídas segundo uma abordagem *ad hoc*, visando objetivos específicos e sem propostas de extensão e reuso. Isso tem levado novas aplicações a simplesmente revisitarem idéias já exploradas e privado pesquisadores de avaliarem seus protótipos sob longos períodos de tempo em cenários de uso reais (Truong & Abowd, 2004).

Há, portanto, demanda por um modelo de desenvolvimento que permita a prototipação rápida de uma aplicação de C&A, bem como seu desenvolvimento, manutenção e evolução frente a uma realimentação constante provida pela comunidade de usuários (Abowd, 1999b).

Uma abordagem comum para se atender a esse tipo de requisito consiste (i) na criação de abstrações para a classe de aplicações especificada, (ii) no desenvolvimento de uma solução

arquitetural para dar apoio ao projeto e à construção dessas aplicações, (iii) na implementação de infra-estruturas e serviços que englobem essas abstrações, e (iv) na experimentação com as abstrações, arquitetura e infra-estruturas por meio do desenvolvimento de aplicações dentro do espaço de projeto delimitado (Truong & Abowd, 2004).

Neste capítulo são descritas algumas das pesquisas realizadas no contexto do Projeto InCA-SERVE, descrito na Seção 4.2, que ataca o problema de captura e acesso segundo essa abordagem. Em um primeiro nível de abstração, o de rede, é descrita a infra-estrutura INCA (Seção 4.3), responsável pela comunicação entre módulos de captura e acesso acoplados às aplicações. Em seguida, respectivamente nas Seções 4.4 e 4.5, são apresentadas a infra-estrutura xINCA, uma extensão da INCA constituída de componentes de software reutilizáveis que implementam as funcionalidades mais comumente encontradas em aplicações de C&A, e o serviço StRES, responsável pelo armazenamento dos documentos produzidos na etapa de captura. A modelagem e implementação da infra-estrutura xINCA, juntamente com a integração de seus componentes ao modelo de armazenamento do serviço StRES, constitui o foco deste trabalho de mestrado, bem como uma de suas principais contribuições. Por fim, na Seção 4.6 são apresentadas as considerações finais do capítulo.

4.2 O Projeto InCA-SERVE

O trabalho aqui descrito foi desenvolvido no contexto do Projeto InCA-SERVE, cujo objetivo é implementar infra-estruturas de captura e acesso genéricas e fornecer facilidades para o armazenamento, a extensão, a recuperação e a visualização de informações provenientes de atividades de captura.

O Projeto InCA-SERVE advém de uma parceria entre o Georgia Tech, sob supervisão do Dr. Gregory Abowd, e o ICMC-USP, onde é supervisionado pela orientadora deste projeto, recebendo apoio financeiro, da National Science Foundation (NSF), nos EUA, e de CNPq e FAPESP, no Brasil (Pimentel et al., 2001a).

A InCA (*Infrastructure for Capture and Access*) é a parte do Projeto InCA-SERVE responsável pela implementação da infra-estrutura de captura e acesso de informações associadas a experiências “ao vivo” em vários domínios. A SERVE (*Infrastructure for Store, Extend, Retrieve and Visualize Evolutionary Information*) responde pela parte de armazenamento, extensão, recuperação e visualização de informação evolucionária. De uma forma simplificada, pode-se dizer que a InCA gera informações como resultado de atividades de captura e a SERVE armazena e estende tais informações a fim de otimizar a recuperação da informação armazenada e permitir sua visualização.

4.3 INCA

As infra-estruturas INCA e xINCA são produtos da porção InCA do Projeto InCA-SERVE enquanto que o serviço StRES, da porção SERVE.

4.3 INCA

Durante a construção de aplicações de C&A, muitas similaridades podem ser notadas sob a perspectiva do desenvolvedor. (Truong & Abowd, 2004) sugerem expressar o projeto de uma aplicação de C&A de acordo com as principais tarefas que a mesma desempenha: a *captura* de informação como fluxos de dados associados a atributos representando meta-informação contextual, a *conversão* dessa informação entre diferentes formatos e tipos de dados, o *armazenamento* da informação e atributos correspondentes, e o *acesso* a fluxos de dados inter-relacionados (de acordo com seus atributos) como resposta a consultas baseadas em contexto.

Para validar sua proposta, (Truong & Abowd, 2004) desenvolveram a INCA (*IN*frastructure for *C*apture & *A*ccess Applications), que provê suporte arquitetural para a construção de aplicações de C&A por meio da instanciação e implementação, respectivamente, de módulos e interfaces de software correspondentes aos conceitos-chave de projeto — captura, armazenamento, conversão e acesso à informação.

Mais especificamente, a parte do sistema responsável pela captura de informação implementa a interface de captura (Capturer) e instancia um módulo de captura (CaptureModule). Da mesma maneira, existem módulos de armazenamento (StorageModule), conversão (TransductionModule) e acesso (AccessModule), e suas respectivas interfaces: Storer, Transducer e Accessor.

De modo a dar ao desenvolvedor da aplicação maior liberdade para se concentrar em aspectos específicos dos processos de captura e acesso, o suporte provido pela INCA consiste de dois níveis de abstração: abstração de comunicação e abstração de informação, detalhadas a seguir.

4.3.1 Abstração de Comunicação

Quando desenvolvedores implementam aplicações de C&A usando a INCA, eles não precisam se preocupar com detalhes de comunicação entre os diferentes módulos (por exemplo, conexões via *sockets* ou protocolos de rede suportados). A própria INCA provê mecanismos que abstraem esse processo.

A INCA possui uma entidade de registro (Registry) que controla automaticamente a presença de todos os módulos no sistema, que, uma vez registrados sob os mesmos atributos, tem o fluxo de comunicação entre si imediatamente habilitado. Na Figura 4.1(a) é ilustrada uma configuração básica para uma aplicação de C&A operando em modo isolado (*stand-alone*) —

todos os componentes residindo na mesma máquina. Entretanto, configurações mais complexas também são possíveis, como pode ser visto na Figura 4.1(b), devido à capacidade dos Registrys de compartilhar informação uns com os outros.

(a) Todos os componentes residindo em uma mesma máquina.

(b) Componentes distribuídos em várias máquinas.

Figura 4.1: Possíveis configurações do Registry (Truong & Abowd, 2002).

4.3.2 Abstração de Informação

Quando aplicações são construídas para suportar um tipo específico de informação, essa limitação pode vir a dificultar a evolução do sistema como um todo. Se novos tipos de informação precisam ser suportados, grandes porções do sistema precisam ser re-implementadas.

A INCA trata a informação capturada e disponibilizada para acesso por meio de objetos binários. A informação é encapsulada como objetos de dados (DataObjects) com uma lista de pares de atributos nome-valor que descrevem meta-informação sobre os dados. Ao impor à informação capturada essa representação uniforme, a especificidade da informação é abstraída da infra-estrutura. Fica, portanto, a cargo da aplicação a manipulação e o uso da informação capturada.

4.4 Estendendo a INCA: xINCA

A INCA é um esforço pioneiro na tentativa de facilitar a construção de aplicações de C&A segundo uma abordagem mais organizada e estruturada. Isso foi conseguido através da iden-

4.4 Estendendo a INCA: xINCA

tificação e classificação de aspectos funcionais comuns presentes em um amplo conjunto de aplicações estudadas e desenvolvidas pelo grupo de pesquisadores do Georgia Tech (Abowd, 1999a; Brotherton & Abowd, 1998; Brotherton et al., 1998; Brotherton et al., 1999; Richter et al., 2001). As camadas de abstração providas pela INCA — comunicação e informação — têm por objetivo dar ao desenvolvedor da aplicação maior liberdade para se concentrar em questões mais relevantes ao processo de captura e acesso, tais como as dimensões de projeto discutidas na Seção 3.2.3.

Entretanto, algumas questões ainda foram deixadas em aberto, principalmente por fugirem do foco desejado pelos pesquisadores do Georgia Tech. A principal delas talvez seja a de que não foi definido um mecanismo que trate a recorrência de funcionalidades em aplicações de C&A. Na Tabela 4.1 são apresentadas funcionalidades presentes em exemplos clássicos de aplicações de C&A reportados na literatura.

Tabela 4.1: Funcionalidades presentes em aplicações de C&A.

Sistema	Áudio	Texto	Vídeo	Web logging	Whiteboard
AutoAuditorium (Bianchi, 1998)	X	–	X	–	X
DOLPHIN (Streitz et al., 1994)	X	X	X	X	X
DUMMBO (Brotherton et al., 1999)	X	–	–	–	X
Dynomite (Wilcox et al., 1997)	X	X	–	–	–
eClass (Abowd, 1999a)	X	–	X	X	X
Flatland (Mynatt et al., 1999)	–	–	–	–	X
LiteMinutes (Chiu et al., 2001)	–	X	X	–	X
Smart Classroom (Shi et al., 2003)	X	X	X	–	X
TeamRoom (Roseman & Greenberg, 1996)	X	X	X	X	X
TeamSpace (Richter et al., 2001)	X	X	X	–	X
Tivoli (Pedersen et al., 1993)	–	–	–	–	X

Funcionalidades como lousa eletrônica (*whiteboard*), troca de mensagens textuais (*chat*), captura de áudio, captura de vídeo e captura de histórico de navegação hipertexto (*web logging*) estão presentes em boa parte das aplicações de C&A e aparecem nas mais variadas combinações.

Da mesma forma, há uma grande expectativa de que avanços na produtividade de software dependerão da capacidade de se combinar peças de software existentes para produzir novas aplicações (Boehm & Scherlis, 1992). A construção de software a partir de blocos reutilizáveis deverá tornar possível a criação de aplicações de software grandes e de alta qualidade de uma maneira muito mais rápida que as técnicas “construir-do-zero” que dominam a maioria da produção de software (Garlan et al., 1995).

A xINCA¹ (*Extended INfrastructure for Capture & Access Applications*) foi projetada como uma camada acima da INCA, utilizando-se dos módulos e paradigmas de abstração da mesma, conforme ilustrado na Figura 4.2. Constituída por componentes de software que implementam as funcionalidades mais comumente encontradas em aplicações de C&A (*whiteboard*, texto, áudio, vídeo e *web logging*), a xINCA permite que uma nova aplicação de C&A seja construída por meio do agrupamento e arranjo dos componentes desejados.

Figura 4.2: Camadas da arquitetura xINCA.

4.4.1 Componentes xINCA

A xINCA é uma extensão da infra-estrutura INCA, constituindo-se de um conjunto de componentes de software reutilizáveis que implementam as funcionalidades recorrentes existentes em um grande número de aplicações de C&A. Usa-se o termo *componente* para denotar um pacote coerente de artefatos de software que possa ser desenvolvido e utilizado independentemente como uma unidade e que possa ser composto, inalterado, com outros componentes para se construir algo maior (D'Souza & Wills, 2001).

A seguir são descritos os componentes xINCA desenvolvidos no contexto deste trabalho, alguns de seus detalhes técnicos e seus potenciais cenários de uso:

Whiteboard

Uma *whiteboard*, ou lousa eletrônica, é uma ferramenta que consiste, basicamente, de um espaço de edição em que o usuário pode desenhar fazendo uso de um dispositivo de entrada como o *mouse* ou uma caneta eletrônica. Cenários de uso para tal ferramenta incluem reuniões de projeto em que os membros de uma equipe discutem e esboçam suas idéias na lousa eletrônica, e ambientes de ensino em que um professor substitui a lousa convencional pela lousa eletrônica. A *whiteboard* fornece ainda funcionalidades extras como exibição de imagens e gráficos como *background*, navegação entre quadros e a possibilidade de captura das ações do usuário.

¹A xINCA pode ser obtida, sob licença GPL, na URL <http://xınca.sf.net>.

4.4 Estendendo a INCA: xINCA

O espaço de edição do componente xINCA de *whiteboard*, definido na classe XBoard, consiste de um objeto JPanel cujas dimensões podem ser ajustadas segundo as preferências do usuário. Novos quadros, chamados de *slides*, podem ser criados a partir de comandos do usuário, bem como navegados, deletados e duplicados. Imagens no formato JPG e PNG podem ser carregadas como *backgrounds* para os *slides* e também geradas como representação dos mesmos. Há dois modos de interação: o caneta, para criação de traços (*strokes*) sobre os *slides*, e o borracha, para remoção de traços. Há ainda propriedades personalizáveis, como espessura e cor da caneta.

Essas funcionalidades podem ser usadas em sua totalidade ou apenas em parte. Para o componente xINCA de *whiteboard* foi criada uma barra de ferramentas padrão, XBoardToolBar, cujas funções, implementadas através de botões, incluem: navegação entre *slides*, criação de novos *slides*, deleção de *slides*, duplicação de *slides*, mudança de cor e espessura de caneta e borracha.

Na Figura 4.3 é exibido o componente xINCA de *whiteboard*. Na parte esquerda encontra-se a barra de ferramentas padrão com a descrição das funções de cada botão. Na parte direita, o espaço de edição com um *slide* em branco.

Figura 4.3: Componente xINCA de *whiteboard*.

Os eventos resultantes da interação do usuário, como desenho de novos traços e navegação entre *slides*, são transmitidos pela INCA e repassados a aplicação por meio de chamadas

callback, como `handleStroke()` e `handleJumpToSlide()`, definidas na interface do componente, `XBoardListener`.

Texto

O componente xINCA de texto, `XChat`, permite a troca síncrona de mensagens textuais entre aplicações. Esse tipo de funcionalidade é muito comum em ambientes que suportam conferência eletrônica e, no cenário de ambientes de ensino, pode permitir, por exemplo, que alunos encaminhem dúvidas e comentários ao professor.

Na Figura 4.4 é apresentada a interface gráfica do componente xINCA de texto. Na parte inferior encontra-se a área de edição de mensagens e, na parte superior, a área de visualização de mensagens.

Figura 4.4: Componente xINCA de texto.

A interface do componente, `XChatListener`, repassa as mensagens trocadas entre usuários através do método `handleChatMessage()`.

Captura de Áudio

Como a fala é um dos principais meios de expressão do ser humano, é imprescindível a existência de um componente que registre esse tipo de interação no contexto de uma infra-estrutura como a xINCA. O componente xINCA para captura de áudio, `XSound`, faz uso da API JMF² (*Java Media Framework*), permitindo a gravação de áudio a partir de qualquer dispositivo e em qualquer formato suportados por essa API.

Trata-se de um componente *multithread* que faz uso da abstração `DataSource` da API JMF para representação do dispositivo de captura, que no caso pode ser um microfone especificado pelos *drivers* `DirectSoundCapture` ou `JavaSound`. O fluxo de mídia capturado é armazenado

²<http://java.sun.com/jmf>.

4.4 Estendendo a INCA: xINCA

em um arquivo local por meio de um objeto `DataSink` e, ao final da sessão, transmitido, via INCA, às partes interessadas. Devido a uma limitação da INCA, que só permite transmissão de pacotes XML (`DataObjects`), não é suportado *streaming* de áudio.

Na interface do componente xINCA de áudio, `XSoundListener`, especial atenção deve ser dada ao método `getOutputFormat()`, que retorna um `FileTypeDescriptor` com o formato de saída de áudio desejado, entre os quais: AIFF, AU, AVI, GSM, MIDI, MPEG, QuickTime, RMF e WAV.

Captura de Vídeo

Para o caso do ambiente instrumentado de captura dispor de câmera de vídeo, é interessante a existência de um componente que capture informação de vídeo. A exemplo do componente de áudio, o componente xINCA para captura de vídeo, `XVideo`, faz uso da API JMF para a gravação de vídeo a partir dos dispositivos e nos formatos suportados pela mesma.

Também a exemplo do componente de áudio, a gravação do fluxo de vídeo ocorre em um thread separado, fazendo uso das abstrações `DataSource`, para o dispositivo, e `DataSink`, para gravação de um arquivo local a ser posteriormente transmitido pela INCA. Não há suporte a *streaming* de vídeo.

A interface do componente de vídeo, `XVideoListener`, é uma especialização da interface do componente de áudio, possuindo métodos extras, como `getVideoDevice()` e `getVideoFormat()`, que especificam, respectivamente, o dispositivo e o formato de vídeo a serem utilizados. Dentre os formatos de vídeo suportados estão: AVI, QuickTime, MPEG e RTP.

Captura de Histórico de Navegação Hipertexto

Muitas vezes o ambiente físico em que ocorre a sessão de captura dispõe de infra-estrutura para acesso à Internet. Nesses casos, é interessante registrar um histórico dos sites visitados durante uma sessão. Por exemplo, em um ambiente de ensino, o professor pode apresentar aos seus alunos sites relacionados ao conteúdo da aula e ter essa rica informação capturada para ser posteriormente acessada pelos alunos.

O componente xINCA de *web logging* permite a captura das URLs dos sites visitados durante a sessão. Usando um objeto `WebProxyServer`, disponibilizado pela API da INCA, é instanciado um *proxy* na máquina local que monitora as requisições HTTP feitas pelo *browser*. O *proxy* não realiza nenhum tipo de controle ou processamento sobre o conteúdo acessado, limitando-se a armazenar as URLs dos sites visitados. Quando uma página *Web* é acessada, geralmente ela referencia outros documentos, principalmente imagens, o que implica em novas requisições do *browser*, as quais, na verdade, não foram feitas pelo usuário, mas fazem parte do

conteúdo apresentado. Para evitar que o componente xINCA de *web logging* armazene URLs não provenientes de ações do usuário (navegação explícita por endereço e/ou por *links*), é usado o campo Referer do protocolo HTTP 1.0, que fornece a URL a partir da qual originou-se a requisição. Esse campo permite diferenciar requisições decorrentes de ações de usuário (valor nulo) de requisições decorrentes de documentos associados ao documento principal acessado (valor não-nulo).

4.4.2 Funcionamento dos Componentes

Componentes xINCA de uma mesma classe (funcionalidade) comunicam-se através de um identificador de sessão. Uma *sessão* caracteriza um período de interação entre componentes xINCA e possui um identificador único durante todo seu tempo de duração. Novas instâncias de componentes xINCA, constituintes das aplicações, podem ser adicionadas e removidas a qualquer instante durante uma sessão.

Os componentes xINCA possuem instâncias dos módulos INCA de captura, de acesso ou de ambos. Esses módulos são registrados em tempo de execução, segundo o identificador de sessão, em um Registry da infra-estrutura INCA que pode ou não ser remoto. Componentes xINCA, dotados de módulos INCA, registrados em um mesmo Registry (ou em uma arquitetura de Registries distribuída e compartilhada — vide Figura 4.1(b) na página 22), sob um mesmo identificador de sessão, estão habilitados a trocar informação entre si.

Na Figura 4.5 é ilustrado, em linhas gerais, o fluxo de comunicação entre quatro aplicações de C&A dotadas de componentes xINCA registrados em duas sessões simultâneas. A Aplicação #1 possui um componente de *whiteboard* registrado na Sessão #1 e um componente de áudio registrado somente para captura na Sessão #2; a Aplicação #2 possui um componente de áudio registrado para acesso na Sessão #2; a Aplicação #3 possui um componente de *whiteboard* registrado na Sessão #1; por fim, a Aplicação #4 possui um componente de *whiteboard* registrado para acesso na Sessão #1. As informações de *whiteboard* produzidas pela Aplicação #1 e pela Aplicação #3 são acessadas por elas próprias e pela Aplicação #4, enquanto que a informação de áudio produzida pela Aplicação #1 é acessada pela Aplicação #2. A troca de informação é coordenada por dois Registries interligados.

Estruturação da Informação Capturada

De modo a organizar a informação produzida pelos componentes xINCA, foram definidos esquemas XML³ para cada um deles. Os esquemas XML permitem estruturar a informação

³<http://www.w3.org/XML/Schema>.

4.4 Estendendo a INCA: xINCA

Figura 4.5: Fluxo de comunicação entre componentes xINCA para duas sessões simultâneas (retângulos em cinza representam diferentes computadores).

capturada pelos componentes xINCA em elementos e atributos que representam partes específicas da mesma segundo uma hierarquia própria. Esses elementos e atributos possuem tipos de dados padrões e restrições associados, correspondentes ao tipo de conteúdo que armazenam.

A seguir, é apresentado o esquema XML para o componente xINCA de *web logging*. Como pode ser notado, há um elemento raiz *weblogml* (linha 03) contendo zero ou mais logs (linha 06). O elemento *log* (linha 09) compreende um conteúdo tipo *string* (linha 11) que armazena a URL do site visitado e atributos *id*, *time-offset* e *person-id* (linhas 12 a 14) que provêm, respectivamente, o identificador do log, um *timestamp* em milissegundos de quando o site foi visitado (relativo ao início da sessão de captura) e o nome de usuário da pessoa que visitou o site.

```
01 <?xml version="1.0"?>
02 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
03   <xsd:element name="weblogml" type="WeblogmlType"/>
04   <xsd:complexType name="WeblogmlType">
05     <xsd:sequence>
06       <xsd:element name="log" minOccurs="0" maxOccurs="unbounded" type="Log"/>
07     </xsd:sequence>
08   </xsd:complexType>
09   <xsd:complexType name="Log">
10     <xsd:simpleContent>
11       <xsd:extension base="xsd:string">
12         <xsd:attribute name="id" type="xsd:int" use="required"/>
13         <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
14         <xsd:attribute name="person-id" type="xsd:string"/>
15       </xsd:extension>
16     </xsd:simpleContent>
17   </xsd:complexType>
18 </xsd:schema>
```

Fazendo uso de JAXB (*Java Architecture for XML Binding*)⁴, uma API para mapeamento entre documentos XML e objetos Java provida pela Sun Microsystems, foram derivadas classes Java para os elementos e atributos definidos no esquema XML de cada componente xINCA. Instâncias dessas classes são os tipos de dados primitivos manipulados pelos componentes xINCA. Elas são disponibilizadas para a aplicação de C&A por meio de métodos *callback* definidos nas interfaces de programação dos componentes. Cada componente xINCA possui sua própria interface de programação, que deve ser implementada pela aplicação de C&A para ativação do componente. Para permitir extensão, foram criadas as super-classes *XINCAComponent* e *XINCAListener* a partir das quais novos componentes e interfaces, respectivamente, podem ser derivados. Retomando o exemplo da captura de navegação hipertexto, o componente xINCA de *web logging* (*XWebLog*) provê objetos *Log*, criados a partir do método *createLog()* da classe *ObjectFactory*, à aplicação de C&A através da chamada *callback* *handleVisitedURL()* definida em sua interface (*XWebLogListener*). Na Figura 4.6 é apresentada parte do diagrama de classes UML da xINCA correspondente ao componente xINCA de *web logging* e de seu modelo de dados (classes geradas pelo compilador JAXB).

Os objetos *Log*, recebidos pelas chamadas *callback* *handleVisitedURL()*, compõem um objeto *Weblogml*, criado a partir do método *createWeblogml()* da classe *ObjectFactory*, todos conformantes com as definições de classe geradas pelo compilador JAXB. Uma vez que a API JAXB provê, em um processo conhecido por *marshaling*, métodos para o mapeamento de uma árvore de conteúdo formada por objetos Java em sua representação XML, o objeto *Weblogml* pode ser convertido em um documento XML do tipo:

```
01 <weblogml>
02   <log id="1" time-offset="356570" person-id="renan">http://www.acm.org</log>
03   <log id="2" time-offset="426320" person-id="renan">http://coweb.icmc.usp.br</log>
04   <log id="3" time-offset="786490" person-id="renan">http://iclass.icmc.usp.br</log>
05   <log id="4" time-offset="959650" person-id="renan">http://w3.org</log>
06 </weblogml>
```

Os mecanismos de funcionamento para os demais componentes xINCA são semelhantes ao descrito para o componente de *web logging*, de modo que, para evitar que o texto se torne demasiadamente repetitivo, os mesmos foram omitidos desta seção. O diagrama de classes UML completo da infra-estrutura xINCA e os esquemas XML para os demais componentes podem ser encontrados, respectivamente, nos Apêndices A e B.

⁴<http://java.sun.com/xml/jaxb>.

Figura 4.6: Diagrama de classes UML do componente xINCA de *web logging* e de seu modelo de dados.

4.4.3 Criação de Novos Componentes

Ao usar o modelo da xINCA, o desenvolvedor de aplicações de C&A continua tendo acesso direto à camada INCA, sendo capaz de utilizar-se diretamente dos recursos da mesma ou de estender a xINCA por meio da adição de novos componentes.

Um novo componente xINCA pode ser criado pela implementação e encapsulamento da nova funcionalidade em um componente e pela definição da interface e do esquema XML correspondentes. As classes XINCAComponent e XINCAListener servem, respectivamente, como superclasses para novos componentes e interfaces.

4.4.4 Algumas Observações Sobre o Uso da INCA

O uso da INCA como camada de abstração de comunicação evidenciou tanto vantagens quanto desvantagens. Dentre as vantagens, pode-se citar:

- O fato da INCA ser especializada para o domínio de captura e acesso agiliza o projeto de novas aplicações e assegura que as mesmas sigam o mesmo modelo genérico;
- Ao prover diversos recursos e serviços auxiliares, como monitoração (definida na classe Inspector) e gerenciamento (definido na classe Manager), a INCA facilita o desenvolvimento e a manutenção das aplicações.

Algumas das desvantagens observadas foram:

- Apesar da abordagem *publish/subscribe* usada pela INCA ser de fácil compreensão e de uso direto, ela é menos flexível em termos de inter-conexões entre nós do que soluções *peer-to-peer* existentes, como JXTA⁵;
- O fato de o Registry ser um ponto centralizado na arquitetura e poder ser comum a um grande número de módulos pode levar a problemas de escalabilidade;
- Ao contrário dos pacotes de informação trocados entre componentes xINCA (ex. Logs), os DataObjects são apenas contêineres para o transporte de dados e a definição de seu conteúdo e atributos associados fica a cargo do desenvolvedor da aplicação.

⁵<http://www.jxta.org>.

4.5 Automatizando o Armazenamento: a Abordagem xINCA-StRES

Uma vez que o modelo original de armazenamento da INCA suporta apenas um mecanismo simplificado de armazenamento (conteúdos de *DataObjects* indexados por seus atributos organizados em pares nome-valor) e apresenta funcionalidades limitadas para a recuperação de informação, foi necessário estender o modelo de aplicação da xINCA para deixar as operações de armazenamento e recuperação da informação capturada com o mesmo nível de abstração dos componentes xINCA. Essa extensão foi implementada por meio de um serviço chamado StRES (*Storing, Retrieving and Extending Service*) (Baldochi et al., 2002b; Baldochi et al., 2003)⁶, que provê abstrações para a geração e o gerenciamento de informação produzida em sessões de captura “ao vivo”. Essas abstrações são obtidas por uma infra-estrutura de *middleware* que oculta detalhes de implementação, como os relacionados à base de dados usada para armazenar informação.

O StRES compreende dois elementos principais: um *StorageComponent*, que executa no lado do cliente, e um *Transducer*, que executa no lado do servidor. Quando uma aplicação de C&A dotada de componentes xINCA de captura deseja ter a informação produzida pelos mesmos capturada, ela instancia um objeto *StorageComponent* para a respectiva sessão e informa quais componentes xINCA de captura devem ser monitorados. O *StorageComponent* contata então um *daemon* (*Sessiond*) no servidor, que instancia um *Transducer* com os componentes xINCA de acesso correspondentes. O *Transducer*, monitorando as chamadas *callback* produzidas pelas interfaces dos componentes xINCA de acesso, começa a receber a informação produzida pelos componentes xINCA de captura e agrupa essa informação em instâncias das classes geradas via JAXB. No fim da sessão, quando não houver mais nenhum componente xINCA de captura ativo, a informação capturada, mantida em instâncias de objetos JAXB, é mapeada para os documentos XML correspondentes que passarão a representar o conteúdo da sessão recém-finalizada. Na Figura 4.7 é ilustrado tal processo para uma aplicação que faz uso de componentes xINCA de captura de áudio e de *whiteboard*.

Para gerar um documento XML para a sessão capturada, o StRES utiliza-se de um esquema XML em um processo análogo ao usado pela xINCA. Esse esquema de sessão reúne todas as informações produzidas pelos componentes xINCA mais um elemento *context-info*, que provê informação de contexto⁷ sobre a sessão. A seguir, é apresentado o esquema XML de sessão

⁶Doutorando cujo trabalho também se desenvolve no contexto do Projeto InCA-SERVE.

⁷Representada pelas dimensões de contexto — *o que, quando, quem e onde* — e por informação específica da aplicação.

Figura 4.7: O modelo de aplicação xINCA-StRES (Cattelan et al., 2003b).

usado pelo StRES. Nesse esquema XML, as linhas 23 a 25, 27 e 28 contêm ponteiros⁸ para os documentos XML gerados por cada componente xINCA. O elemento context-info (linha 06) contém informações sobre as dimensões de contexto *o que* (elemento what na linha 14), *quem* (elemento who na linha 15), *quando* (elemento when na linha 16) e *onde* (elemento where na linha 17) definidas na sessão de captura, e um elemento app-context (linha 18), o qual por sua vez é um ponteiro para informação de contexto específica da aplicação de captura e acesso.

```

01 <?xml version="1.0"?>
02 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
03   <xsd:element name="session" type="Session"/>
04   <xsd:complexType name="Session">
05     <xsd:sequence>
06       <xsd:element name="context-info" type="ContextInfo"/>
07       <xsd:element name="captured-media" type="Captured-media"/>
08     </xsd:sequence>
09     <xsd:attribute name="id" type="xsd:int"/>
10     <xsd:attribute name="type" type="xsd:int"/>
11   </xsd:complexType>
12   <xsd:complexType name="ContextInfo">
13     <xsd:sequence>
14       <xsd:element name="what" type="What"/>
15       <xsd:element name="who" type="Who"/>
16       <xsd:element name="when" type="When"/>
17       <xsd:element name="where" type="Where"/>
18       <xsd:element name="app-context" type="xsd:string"/>
19     </xsd:sequence>
20   </xsd:complexType>
21   <xsd:complexType name="Captured-media">
22     <xsd:sequence>
23       <xsd:element name="audio" type="xsd:string" minOccurs="0"/>

```

⁸A versão original do serviço StRES combinava os artefatos produzidos pelos diferentes componentes xINCA no mesmo documento XML, ao invés de referenciá-los em documentos separados.

4.5 Automatizando o Armazenamento: a Abordagem xINCA-StRES

```
24     <xsd:element name="chat" type="xsd:string" minOccurs="0" />
25     <xsd:element name="video" type="xsd:string" minOccurs="0" />
26     <xsd:element name="voice" type="xsd:string" minOccurs="0" />
27     <xsd:element name="weblog" type="xsd:string" minOccurs="0" />
28     <xsd:element name="whiteboard" type="xsd:string" minOccurs="0" />
29   </xsd:sequence>
30 </xsd:complexType>
31 <xsd:complexType name="What">
32   <xsd:sequence>
33     <xsd:element name="title" type="xsd:string" />
34     <xsd:element name="description" type="xsd:string" />
35   </xsd:sequence>
36 </xsd:complexType>
37 <xsd:complexType name="Who">
38   <xsd:sequence>
39     <xsd:element name="person" type="xsd:string" maxOccurs="unbounded" />
40   </xsd:sequence>
41 </xsd:complexType>
42 <xsd:complexType name="When">
43   <xsd:sequence>
44     <xsd:element name="date" type="xsd:date" />
45     <xsd:element name="time" type="xsd:time" />
46     <xsd:element name="duration" type="xsd:long" />
47   </xsd:sequence>
48 </xsd:complexType>
49 <xsd:complexType name="Where">
50   <xsd:sequence>
51     <xsd:element name="location" type="xsd:string" />
52     <xsd:element name="type" type="xsd:string" />
53   </xsd:sequence>
54 </xsd:complexType>
55 </xsd:schema>
```

O StRES possui ainda um repositório de documentos distribuído, construído sobre uma base de dados XML⁹ denominada eXist (Meier, 2003). A funcionalidade da base de dados eXist é baseada em uma hierarquia de coleções, similar ao processo de armazenamento de arquivos em um sistema de arquivos convencional. Além das facilidades para criação e remoção de coleções, e de armazenamento e recuperação de documentos, a base de dados eXist provê um mecanismo de consulta baseado em XPath¹⁰, que permite a recuperação de nós (por exemplo, elementos específicos) do documento, e uma API XML-RPC¹¹, modificada para implementar o serviço de armazenamento distribuído do StRES. Uma entidade chamada DocumentManager é responsável pela implementação da política de armazenamento e gerencia um grupo de repositórios distribuído que aparecem para as aplicações como um repositório único, consistente

⁹Em sua primeira versão, o StRES utilizava como base de dados XML o Xindice (<http://xml.apache.org/xindice>).

¹⁰<http://www.w3.org/TR/xpath>.

¹¹<http://xml.apache.org/xmlrpc>.

e centralizado. Essa política de armazenamento é baseada na replicação total dos documentos XML (Baldochi et al., 2003).

4.6 *Considerações Finais*

Neste capítulo foi abordada a demanda existente por mecanismos que auxiliem o desenvolvimento de aplicações de C&A. Foi apresentada uma visão geral do Projeto InCA-SERVE, que procura colaborar com a área de computação ubíqua no sentido de prover abstrações, em termos de infra-estruturas e serviços, que facilitem a construção de aplicações de C&A.

Nesse sentido, este trabalho constitui-se em um desses mecanismos — a xINCA, envolvendo aspectos relacionados à recorrência de funcionalidades em aplicações de C&A por meio de componentes de software reutilizáveis e a estruturação da informação capturada.

Também foram descritos a INCA, que serve como camada de comunicação para o componentes desenvolvidos, e o StRES que agrupa os diferentes blocos de informação produzidos pelos componentes xINCA em um único documento representativo da sessão capturada.

Aplicações Desenvolvidas

5.1 *Considerações Iniciais*

A xINCA, juntamente com as demais infra-estruturas e serviços desenvolvidos no contexto do Projeto InCA-SERVE, serviu de base para a construção de aplicações de captura e acesso (C&A) para diferentes domínios de uso. Neste capítulo, são descritas duas delas: o iClass, na Seção 5.2, para captura de aulas em ambiente educacional instrumentado; e o CAT, na Seção 5.3, um *toolkit* para geração de sessões de captura distribuídas. Na Seção 5.4 são feitas algumas considerações sobre a avaliação de infra-estruturas de software. Por fim, na Seção 5.5 são expostos os comentários finais sobre o capítulo.

5.2 *iClass*

Similar em em funcionalidade ao eClass (Abowd, 1999a), desenvolvido no Georgia Tech, o Sistema iClass¹ (Cattelan et al., 2003a) permite a captura de diferentes fluxos de informação produzidos durante uma aula convencional realizada em uma sala de aula instrumentada. A informação capturada inclui os *slides* apresentados pelo professor em uma lousa eletrônica e as anotações (*strokes*) sobre os mesmos, o áudio gravado a partir de um microfone sem fio preso à roupa do professor, o vídeo proveniente de uma webcam localizada em uma posição fixa na sala, e as URLs dos sites visitados durante a aula. Como resultado, ao final da aula, um documento

¹<http://iclass.icmc.usp.br>.

XML que integra as diferentes mídias capturadas é gerado e processado para criar documentos derivados em uma variedade de formatos de apresentação.

O iClass segue as fases sugeridas por (Abowd et al., 1996) para estruturação do problema de captura e acesso: pré-produção, gravação “ao vivo”, pós-produção e acesso, detalhadas na Seção 3.2.4. Na Figura 5.1 é ilustrada a arquitetura do iClass de acordo com cada uma das fases, que são detalhadas a seguir.

Figura 5.1: Arquitetura do sistema iClass.

5.2.1 Pré-produção

A fase de pré-produção está relacionada à configuração da aula a ser capturada, adicionando informações contextuais, como a que curso essa aula deve ser associada. Uma série de páginas JSP (Figuras 5.2 a 5.5) são usadas para a criação de aulas (Sante, 2003)², possibilitando que o professor carregue *slides* pré-preparados (imagens JPEG ou PNG) para cada aula. Esses *slides* servem de *background* para o componente xINCA de *whiteboard* usado na sessão de captura e o

²Trabalho de mestrado também desenvolvido no contexto do Projeto InCA-SERVE.

5.2 iClass

professor pode desenhar sobre os mesmos, através de uma lousa eletrônica dotada de superfície sensível ao toque que simula o uso do *mouse*.

Figura 5.2: Formulário inicial de criação de aula em que é optado ou não pelo reuso de material.

Figura 5.3: Formulário de criação de aula sem reuso de material: o professor seleciona um curso e fornece informações básicas sobre a aula (título, assunto, resumo e arquivo de *slides*).

The screenshot shows the 'New Lecture' form in the 'Preparation Module' of the 'AutorE class' application. The form is titled 'New Lecture' and contains the following fields and options:

- Course:** A dropdown menu with 'Multimídia (SCE219)' selected.
- Term:** A dropdown menu with '1o Semestre' selected.
- Year:** A dropdown menu with '2004' selected.
- Title:** A text input field with 'MPEG' entered.
- Date:** A text input field with '16/02/2004' entered.
- Subject:** A dropdown menu.
- Abstract:** A text input field.
- Options:**
 - ☒ Add new slides. Slides file: C:\Documents and Settings\... Browse...
 - ☐ Reuse slides used during the preparation of previous lectures.
 - ☒ Reuse slides captured in previous lectures.
 - ☐ Reuse slides searched by the following keywords: [text input field]
- Navigation:** 'BACK' and 'NEXT >>' buttons.

Figura 5.4: Formulário de criação de aula com reuso de material: o professor seleciona um curso, fornece informações básicas sobre a aula (título, assunto, resumo e novos *slides*) e pode escolher o tipo de reuso desejado (reuso de *slides* originais, anotados ou uma mistura de ambos).

The screenshot shows the 'Select Slides' screen in the 'Preparation Module' of the 'AutorE class' application. The screen is titled 'New Lecture - Select Slides' and displays the following information:

- Course:** SCE219
- Term:** 1o Term - 2004
- Slides available to be used to prepare lectures:**
 - MPEG (16/02/2004): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22
 - Padroes de video (16/02/2004): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22
- Captured Slides available:**
 - 1o the Term - 2004
 - Padroes de video (16/02/2004): Session 1:
- Navigation:** 'BACK' and 'NEXT >>' buttons.

Figura 5.5: Listagem das aulas pré-preparadas: após a criação da aula, o professor tem acesso à lista de aulas preparadas e que estão prontas para serem gravadas.

5.2.2 Gravação “ao vivo”

Nessa fase um *applet* é responsável por instanciar os componentes xINCA de captura a serem utilizados durante a aula. Estão atualmente disponíveis componentes de *whiteboard*, áudio, vídeo e *web logging*, selecionados por opção do professor e conforme a disponibilidade de dispositivos de captura do ambiente. Na Figura 5.6 é apresentada a interface de seleção de aula e dos componentes a serem utilizados.

Figura 5.6: Formulário de início de captura de aula, com opção de visualização dos *slides* e seleção dos componentes desejados: áudio, áudio/vídeo e *web logging* (assume-se sempre a presença do componente de *whiteboard*).

O componente xINCA de *whiteboard* permite que *slides* preparados com antecedência sejam apresentados na lousa e anotados pelo professor durante a aula — em qualquer caso, novos *slides* “em branco” podem ser criados conforme a necessidade. Quando *slides* do tipo PowerPointTM são utilizados, o texto original dos *slides* é automaticamente extraído, por meio de uma função macro do MS-OfficeTM, para ser utilizado em operações de indexação, busca e ligação automática (Pimentel et al., 2001b). No caso de não serem utilizados *slides* preparados com antecedência, o professor simplesmente escreve na lousa em branco, gerando quantos *slides* forem necessários. Para cada novo *slide* criado, uma imagem correspondente é gerada automaticamente para ser disponibilizada após a aula.

Quando presente, o componente xINCA para captura de áudio grava um arquivo local em

formato GSM³ (*Global System for Mobile telecommunication*), que será posteriormente transmitido ao Transducer. Esse formato de áudio foi escolhido por sua alta taxa de compressão (que é de aproximadamente 6MB/hora em qualidade mono e taxa de amostragem de 8KHz a 16 bits). O mesmo vale para o componente de vídeo, exceto pelo formato utilizado — o QuickTime, que apresenta uma taxa de compressão de aproximadamente 43MB/hora com resolução de 176 por 144 pixels a 15 quadros por segundo.

O *applet* de captura do cliente iClass também faz uso da barra de ferramentas padrão do componente de *whiteboard* com funções de navegação entre *slides*, criação de novos *slides*, deleção de *slides*, duplicação de *slides*, mudança de cor e espessura de caneta e borracha.

O componente xINCA de *web logging* permite que cada página *Web* apresentada pelo professor durante a aula tenha seu endereço capturado, compondo uma lista de páginas que fará parte do documento gerado automaticamente para a aula. Assim, os alunos não precisam se preocupar em registrar quais sites são visitados durante a aula.

A informação é armazenada pelo StRES com um Transducer para cada sessão de captura em progresso. Componentes xINCA de acesso, executados no servidor, recebem a informação produzida pelos componentes xINCA de captura correspondentes, executados no *applet* do cliente. Usando o mecanismo de chamadas *callback*, os componentes de acesso repassam a informação recebida ao Transducer para que a mesma seja armazenada. Componentes de uma sessão (aula) específica são registrados sob o mesmo e inequívoco identificador de sessão, permitindo que várias aulas ocorram simultaneamente. Na Figura 5.7 é apresentado o andamento da captura de uma aula.

5.2.3 Pós-produção

O início da fase de pós-produção é caracterizado pelo encerramento da sessão de captura (aula), quando é gerado um documento XML representativo da mesma. Esse processo recebe o nome de *marshaling*. Em seguida, o documento é armazenado em um banco de dados XML — o Xindice⁴. Tanto a geração quanto o armazenamento do documento XML são feitos pelo serviço StRES através do Transducer correspondente. Um exemplo de documento XML⁵ representativo de uma aula capturada é mostrado na Figura 5.8.

Ainda nessa fase, o documento XML da sessão é processado em conjunto com três folhas de

³<http://kbs.cs.tu-berlin.de/jutta/gsm/js-intro.html>.

⁴A versão do serviço StRES utilizada na construção do Sistema iClass ainda fazia uso do banco de dados Xindice.

⁵A versão do serviço StRES utilizada na construção do Sistema iClass combinava os artefatos produzidos pelos diferentes componentes xINCA no mesmo documento XML.

Figura 5.7: Captura de uma aula em andamento. Na parte superior à esquerda, é mostrada em detalhe a lousa eletrônica SmartBoardTM utilizada para digitalização dos traços. Na parte inferior à direita, detalhe da professora ministrando uma aula.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <session id="46" type="2" date="2002.10.24 AD at 12:11:00 BRST">
+ <context_info>
+ <chatml />
- <whiteboardml board_width="640" board_height="480">
+ <slide slide_id="0">
+ <slide slide_id="1">
+ <slide slide_id="6">
+ <slide slide_id="2">
+ <slide slide_id="3">
+ <slide slide_id="4">
+ <slide slide_id="5">
  <visit visit_id="0" duration="5038" slide_id="0" />
  <visit visit_id="1" duration="1372" slide_id="1" />
  <visit visit_id="2" duration="891" slide_id="2" />
  <visit visit_id="3" duration="2070" slide_id="3" />
  <visit visit_id="4" duration="2423" slide_id="4" />
  <visit visit_id="5" duration="8466" slide_id="5" />
  <visit visit_id="6" duration="77758" slide_id="3" />
+ <visit visit_id="7" duration="201694" slide_id="1">
+ <visit visit_id="8" duration="400999" slide_id="1">
+ <visit visit_id="9" duration="208863" slide_id="6">
+ <visit visit_id="10" duration="36408" slide_id="1">
+ <visit visit_id="11" duration="62987" slide_id="6">
+ <visit visit_id="12" duration="64527" slide_id="1">
</whiteboardml>
<weblogml />
</session>
```

Figura 5.8: Documento XML (expandido até o 3º nível) representativo de uma aula capturada.

estilo XSLT, gerando documentos SMIL⁶, XHTML+SMIL⁷ e HTML⁸. Os documentos SMIL e XHTML+SMIL possibilitam a apresentação multimídia da sessão, enquanto que o documento HTML é destinado à revisão rápida do conteúdo dos *slides*, permitindo também a impressão de todos eles.

5.2.4 Acesso

A interface de acesso (Andrade, 2003)⁹, mostrada na Figura 5.9, é gerada automaticamente com *scripts* PHP que fazem consultas ao Xindice através de XML-RPC. Para tanto, utilizou-se uma classe PHP, a *class_xindice*, que submete consultas XPath (como *listCollections()* e *listDocuments()*) ao Xindice. Com a lista de semestres, obtida através dessas consultas, é montada uma interface hierárquica de anos e semestres, selecionáveis quando possuírem cursos. Ao se selecionar um semestre, são listados seus cursos, que podem ter ligações caso possuam aulas capturadas. Quando um curso é selecionado, uma nova tela com suas aulas capturadas é gerada. As listas de cursos e aulas também são obtidas por meio de consultas ao Xindice. Para cada aula do curso selecionado, são apresentadas as opções de visualização SMIL, XHTML+SMIL e HTML, cujos documentos foram gerados na fase anterior, de pós-produção. Esses documentos são reproduzidos em um *player*, navegador ou *plugin* apropriado (por exemplo, para o SMIL é usado o RealOne PlayerTM e para o XHTML+SMIL, o Internet Explorer 5.5 ou superior). Na Figura 5.10 é mostrada uma aula sendo apresentada em SMIL com *slides* e linha do tempo, onde cada marcador é uma ligação para a visita de um *slide*. Os traços desenhados pelo professor são ligações para o fluxo de áudio e permitem, quando clicados, a reprodução do áudio correspondente ao momento em que foram criados.

5.2.5 Comparação eClass — iClass

Sendo o eClass e o iClass sistemas que apresentam funcionalidades semelhantes e escritos na mesma linguagem de programação (Java), um dado comparativo interessante entre os mesmos pode ser obtido na contagem de linhas de código não comentadas (NCSS — *Non Commented Source Statements*). Utilizando-se o programa JavaNCSS¹⁰ e analisando-se somente as porções de código correspondentes às etapas de captura e armazenamento, foi obtido o resultado expresso na Tabela 5.1.

⁶<http://www.w3.org/audiovideo>.

⁷<http://www.w3.org/TR/XHTMLplusSMIL>.

⁸<http://www.w3.org/MarkUp>.

⁹Trabalho de mestrado também realizado no contexto do Projeto InCA-SERVE.

¹⁰<http://www.kclee.com/clemens/java/javancss>.

5.2 iClass

Figura 5.9: Interface gráfica de acesso do iClass: as linhas representam aulas e as colunas possuem ligações para diferentes formatos de apresentação.

Figura 5.10: Interface de apresentação SMIL da informação de uma sessão capturada.

Tabela 5.1: Comparação entre os sistemas eClass e iClass.

Sistema	Classes	Funções	NCSS	Javadocs	Pacotes
eClass	56	532	11256	115	2
iClass	8	70	986	50	1

Dessa análise é possível observar que o código-fonte do iClass, composto por 986 NCSS, é bem mais conciso que o do eClass, com 11256 NCSS. Esse tipo de resultado já era esperado, considerando-se o fato do iClass utilizar-se das infra-estruturas INCA e xINCA e do serviço StRES, delegando grande parte do trabalho a ser realizado a suas respectivas APIs.

5.3 CAT

Embora facilitem a construção de aplicações de C&A, as infra-estruturas e os serviços do Projeto InCA-SERVE foram desenvolvidos para serem usados por programadores e exigem noções de programação em linguagem Java, uma vez que o desenvolvedor da aplicação deve ser capaz de instanciar e agrupar os módulos e componentes necessários.

O CAT¹¹ (*Capture and Access Toolkit*) (Baldochi et al., 2002b) é um *toolkit* para geração automática de aplicações de C&A que permite que qualquer pessoa, com ou sem experiência em programação, seja capaz de construir e usar uma aplicação de C&A (representada por uma sessão de captura) pela simples seleção das funcionalidades desejadas. Como resultado, é produzido um *front-end* (na forma de um *applet*) correspondente à combinação escolhida.

O CAT fornece uma página *Web* em que o usuário provê informação contextual e seleciona as funcionalidades desejadas para a aplicação a ser gerada. A informação contextual é composta por dados como título da sessão, participantes, local, tipo e controle de acesso. As funcionalidades são escolhidas dentre uma gama de componentes existentes: texto (*chat*), *whiteboard*, áudio, vídeo e *web logging*. Uma vez realizado esse processo, o usuário recebe de volta um *applet* (*front-end*) com todas as funcionalidades escolhidas e pronto para execução.

Na Figura 5.11, é mostrada a interface de criação e configuração de uma nova aplicação/sessão de captura. É importante notar aqui que os conceitos de aplicação e de sessão de captura são intimamente ligados e, portanto, correspondentes. Quando um usuário cria uma aplicação, ele está, na verdade, criando uma sessão de captura para as funcionalidades selecionadas. Na Figura 5.12, é apresentado o *front-end* resultante para a combinação texto + *whiteboard*.

Além de criar aplicações, o usuário também pode participar (fazer *join*) de sessões em andamento. Nesse caso, o usuário herda as características da sessão e recebe diretamente o *front-end* correspondente. Não é necessário nenhum processo de configuração. Através do mecanismo de *join*, os usuários são capazes de interagir uns com os outros, uma vez que alguns componentes, como o texto e a *whiteboard*, são compartilhados. No entanto, há algumas restrições: por exemplo, um usuário só pode participar ou de sessões públicas, ou de sessões privadas às quais

¹¹<http://iclass.icmc.usp.br/cat>.

5.4 Sobre a Avaliação de Infra-estruturas

Figura 5.11: Interface de criação e configuração de uma nova aplicação (Baldochi et al., 2002b).

tenha permissão de acesso (configurada pelo usuário criador da sessão). Também é o usuário criador quem determina o fim da sessão (a sessão termina quando o usuário que a criou realiza *logout*).

Quando a sessão termina, é produzido um documento XML correspondente que é armazenado em uma base de dados e disponibilizado para apresentação. A informação é recuperada através das suas informações contextuais. Na Figura 5.13, é apresentado um exemplo de uma possível interface de acesso para uma sessão capturada.

5.4 Sobre a Avaliação de Infra-estruturas

Enquanto que medidas tradicionais para avaliação, como desempenho, escalabilidade, segurança, robustez, usabilidade, entre outras, podem ser diretamente aplicadas para aplicações de C&A, avaliar a eficiência da infra-estrutura em que as mesmas se baseiam é uma tarefa bem mais complicada. (Edwards et al., 2003) enumeram um conjunto de questões sobre o tema, alguns dos quais experimentados e utilizados no desenvolvimento das aplicações iClass e CAT:

- *Os aspectos técnicos da infra-estrutura são expressos nas características e no estilo das aplicações, e o desenvolvedor deve ser capaz de abstrair a partir deles:* ambas as apli-

Figura 5.12: *Front-end* para aplicação com combinação texto + *whiteboard* (Baldochi et al., 2002b).

Figura 5.13: Interface de acesso para uma sessão capturada (Baldochi et al., 2002b).

5.5 Considerações Finais

cações desenvolvidas usam a abstração de funcionalidades baseada em seleção de componentes provida pela xINCA;

- *O único modo de se avaliar a experiência proporcionada ao usuário pela infra-estrutura é através da construção e do uso de aplicações que se utilizem dela e da correspondente avaliação dessas aplicações:* no caso do iClass, embora não tenha sido realizado nenhum experimento de usabilidade, o sistema vem sendo usado a um longo período de tempo (3 semestres só no ICMC-USP);
- *Os protótipos de aplicação construídos para avaliação devem expressar as características básicas e os objetivos da infra-estrutura, com os usos permitidos da mesma e dentro de um escopo limitado:* nesse caso a colaboração ficou restrita ao modelo de comunicação da INCA (sem *streaming* de áudio e vídeo) e com a interface de acesso baseada na estrutura dos documentos XML produzidos pelos componentes xINCA e pelo serviço StRES.

5.5 Considerações Finais

Neste capítulo foram descritas duas aplicações que se beneficiaram das infra-estruturas e serviços desenvolvidos no contexto do Projeto InCA-SERVE, particularmente a xINCA, por meio da instanciação dos componentes xINCA necessários: o Sistema iClass, para captura de aulas em ambiente educacional instrumentado, e o CAT, um *toolkit* para geração de sessões de captura distribuídas. Finalmente, foram feitas algumas considerações sobre o processo de avaliação de infra-estruturas de software por meio de protótipos de aplicação.

Trabalhos Relacionados

6.1 Considerações Iniciais

A xINCA é uma proposta relativamente inovadora e não foram encontrados na literatura outros trabalhos que abordem, especificamente, a recorrência de funcionalidades em aplicações de captura e acesso (C&A). No entanto, alguns projetos podem servir de base para algumas comparações.

No contexto de captura e acesso, o LiteMinutes (Chiu et al., 2001) faz uso de módulos fracamente acoplados para facilitar a manutenção do sistema. JETS (Shirmohammadi & Georganas, 1997) e Habanero (NCSA Software Development Division, 1996) fazem uso de componentes que interagem uns com os outros em ambientes distribuídos, porém tem como foco maior a área de trabalho cooperativo (CSCW — *Computer Supported Cooperative Work*).

Já para o contexto de computação ubíqua, existem pesquisas para o desenvolvimento de espaços domésticos e de trabalho interativos e programáveis em que captura e acesso é apenas uma entre muitas aplicações que exploram os dispositivos integrados ao ambiente. O EasyLiving (Brumitt et al., 2000) é voltado para o desenvolvimento de infra-estruturas e tecnologias para ambientes inteligentes, por exemplo uma sala de estar instrumentada com sensores, compostos pela agregação de diversos dispositivos, tais como *displays* de parede, câmeras ou teclados. O Gaia (Roman et al., 2002b; Hess et al., 2002) e o Aura (Garlan et al., 2002) provêm infra-estruturas de *middleware* para espaços físicos capazes de interagir e se adaptar às necessidades de seus usuários. O Interactive Workspaces (Johanson et al., 2002) tem como foco a expansão

(*augmenting*) de um espaço de reuniões utilizando tecnologias tais como grandes *displays* e dispositivos sem fio, bem como sua intersecção com a área de interação homem-computador. O Roomware (Streitz et al., 2001) propõe uma visão para espaços de trabalho em que espaços de informação virtual e espaços físicos possuem um projeto integrado.

No restante deste capítulo, são apresentados alguns detalhes técnicos sobre os trabalhos relacionados citados.

6.2 LiteMinutes

O Sistema LiteMinutes (Chiu et al., 2001) apresenta uma arquitetura constituída de módulos fracamente acoplados. Esses módulos provêm flexibilidade para apoiar a combinação de diferentes fluxos de mídia para tipos distintos de reuniões, podendo ser estendidos. No caso, um módulo de anotações combina metadados com as anotações feitas pelo usuário, produzindo um documento HTML que referencia os outros tipos de mídia (vídeo, *slides*, etc.). Na Figura 6.1, é ilustrada a integração entre mídias a partir dos módulos. Apesar de poderem ser usados em outras aplicações, os módulos não foram projetados com esse propósito, mas sim para facilitar o gerenciamento do sistema LiteMinutes.

Figura 6.1: Integração entre mídias a partir dos módulos fracamente acoplados da arquitetura do LiteMinutes (Chiu et al., 2001).

A combinação das mídias em um documento HTML, como feito pelo LiteMinutes, limita a etapa de acesso a dispositivos de visualização restritos a esse formato. Além disso, acaba sendo menos flexível do que a solução obtida pela criação de um documento XML, como feito na abordagem xINCA-StRES, que permite a produção de documentos derivados em qualquer formato a partir de folhas de estilo.

6.3 JETS

O JETS (Shirmohammadi & Georganas, 1997) é um sistema genérico, multi-usuário e colaborativo que permite o compartilhamento, em tempo real, de *applets* através da Internet. Trata-se de uma API para construção de *applets* compartilhados. Da mesma maneira que a xINCA permite a instanciação de componentes compartilhados em um ambiente distribuído, o JETS permite a construção de *applets* compartilhados, provendo consistência, controle de acesso e passagem de dados. No JETS, cada aplicação possui um servidor próprio que, por sua vez, consiste de servidores menores responsáveis pela manipulação das requisições dos clientes. O fluxo de informação ocorre através desse servidor, de maneira similar a uma sessão xINCA, e pode estar sujeito a um controle de acesso. A exemplo dos componentes xINCA, clientes JETS também podem ser adicionados ou removidos em tempo de execução. O JETS não provê nenhuma abstração ou mecanismo para estruturação e armazenamento da informação trocada entre suas aplicações.

6.4 Habanero

O Habanero (NCSA Software Development Division, 1996) é um *framework* de objetos distribuídos que facilita a construção de software para comunicação síncrona na Internet. Consiste de um ambiente colaborativo composto por um conjunto de aplicações que compartilham eventos e estado. O ambiente inclui ainda um servidor que hospeda sessões e um cliente que interage com elas usando uma variedade de aplicações chamadas *hablets*. Um *hablet* pode ser visto como uma entidade correspondente a um componente xINCA. Um cliente Habanero é capaz de interagir entre múltiplas sessões da mesma maneira que uma aplicação desenvolvida com a xINCA. A exemplo do JETS, o Habanero não provê estruturação ou abstrações de armazenamento para a informação que transmite.

6.5 EasyLiving

O Projeto EasyLiving (Brumitt et al., 2000), conduzido no Microsoft Research Center, é dedicado ao desenvolvimento de uma arquitetura que agrega dispositivos distribuídos e heterogêneos à experiência vivida pelo usuário em um ambiente inteligente. Tal arquitetura inclui (i) *middleware* para comunicação assíncrona, (ii) modelagem geométrica para representar a existência de e o relacionamento entre objetos no mundo físico, (iii) percepção em que dados são obtidos de sensores e processados em tempo de execução, e (iv) descrição de serviços que

permite interações automáticas pela exposição de um conjunto de atributos ou comandos reconhecidos. O modelo de comunicação utilizado no EasyLeaving apresenta semelhanças com o modelo de comunicação utilizado pela INCA como a transmissão de dados em mensagens XML e independência de endereçamento. Nesse último caso, o EasyLeaving utiliza um serviço de nomes e de busca que identifica unicamente cada componente presente no sistema, enquanto que a xINCA utiliza identificadores de sessão que referenciam grupos de um ou mais componentes.

6.6 Gaia

O Projeto Gaia (Roman et al., 2002b; Hess et al., 2002), desenvolvido na Universidade de Illinois, engloba uma infra-estrutura de *middleware* distribuída, baseada em CORBA, que coordena os recursos computacionais em um espaço físico, transformando-o em um sistema computacional programável. Tal infra-estrutura é similar à de um sistema operacional tradicional a medida que gerencia as tarefas comuns a todas as aplicações construídas para espaços físicos. Os serviços providos pela infra-estrutura do Gaia incluem eventos, presença (de dispositivos, usuários e serviços), notificação de contexto, *discovery* e serviço de nomes, cada qual especificado por meio de interfaces.

6.7 Aura

Sediado na Universidade Carnegie Mellon, o Projeto Aura (Garlan et al., 2002) propõe o desenvolvimento de arquiteturas de sistema, algoritmos, interfaces e técnicas de avaliação que forneçam ao usuário um conjunto de serviços computacionais de maneira transparente e independente de localização. Sua infra-estrutura se baseia na adaptação de sistemas pré-existentes como o Coda, que provê acesso a arquivos com adaptação de rede, e o Odyssey, que suporta monitoração de recursos e adaptação de aplicações. Além disso, investiga os problemas associados ao desenvolvimento de novos sistemas — como o Spectra, que é um mecanismo de execução remota que utiliza informação de contexto para decidir como melhor executar uma chamada remota, e o Prism, que é uma camada de sistema responsável pela captura e gerenciamento das intenções (objetivos) do usuário.

6.8 Interactive Workspaces

O Projeto Interactive Workspaces (Johanson et al., 2002), da Universidade de Stanford, é voltado ao desenvolvimento e uso de salas instrumentadas contendo grandes *displays* com a capacidade de integrar dispositivos portáteis. O projeto engloba o desenvolvimento de um protótipo de espaço de trabalho interativo, chamado iRoom, uma infra-estrutura de software para esse ambiente, chamada iROS (*interactive Room Operating System*), e experimentos que analisem a interação dos usuários com o espaço de trabalho. A iROS possui três sub-sistemas: o Data Heap, que permite a movimentação de dados entre aplicações; o iCrafter, que provê um sistema para *advertisement* e invocação de serviços; e o Event Heap, que coordena toda as operações de comunicação entre aplicações e os demais sub-sistemas do iROS, armazenando e enviando mensagens (chamadas eventos) na forma de campos nome-tipo-valor.

6.9 Roomware

O Projeto Roomware (Streitz et al., 2001), do Centro de Pesquisa Alemão para Tecnologia da Informação (GMD-IPSI), propõe um abordagem centralizada no usuário para integração entre espaços virtuais de informação e espaços físicos reais em ambientes chamados *cooperative buildings*. A abordagem principal consiste no uso de objetos aumentados com capacidades computacionais resultantes da integração de elementos espalhados pelo ambiente, como paredes, portas e mobília, com dispositivos computacionais de informação. O sistema envolve uma infra-estrutura de rede que provê conectividade entre componentes, uma infra-estrutura de software com recursos de cooperação e compartilhamento; e uma infra-estrutura sensorial e de localização permitindo a mobilidade de usuários e componentes e a determinação de sua posição dentro do ambiente.

6.10 Considerações Finais

A xINCA procura suprir uma lacuna existente no desenvolvimento de aplicações de C&A justamente ao propor uma solução especializada que explora a recorrência de funcionalidades nessas aplicações. Abordagens semelhantes já foram adotadas em sistemas colaborativos, mas não tem sido investigadas pelas infra-estruturas de *middleware* para computação ubíqua atuais, dedicadas à modelagem e representação de informações, arquiteturas, tecnologias e soluções abrangentes para ambientes ubíquos.

Neste capítulo foram apresentados os principais trabalhos relacionados ao tema abordado.

Em uma primeira categoria, foram citados sistemas específicos para o domínio de captura e acesso, como o LiteMinutes, e colaborativos, como o JETS e o Habanero. Tais sistemas apresentam algumas semelhanças com a solução apresentada pela xINCA, como arquiteturas fracamente acopladas e o uso de componentes para composição de aplicações. Em uma segunda categoria foram abordados trabalhos com perspectivas mais generalizadas e ambiciosas, envolvendo a área de computação ubíqua como um todo e relegando captura e acesso a um segundo plano, apenas como mais uma das possibilidades de aplicação oferecidas. Entre eles: EasyLiving, Gaia, Aura, Interactive Workspaces e Roomware.

Conclusões

7.1 Considerações Iniciais

No presente trabalho foi apresentada a infra-estrutura xINCA, constituída por um conjunto de componentes de software reutilizáveis que implementam algumas das funcionalidades recorrentes comuns em aplicações de captura e acesso (C&A), incluindo lousa eletrônica (*white-board*), troca de mensagens textuais, captura de áudio, captura de vídeo e captura de histórico de navegação hipertexto (*web logging*). Idealizada a partir da demanda existente por mecanismos que auxiliem o desenvolvimento de aplicações de C&A, a xINCA foi desenvolvida no contexto do Projeto InCA-SERVE, que procura colaborar com a área de computação ubíqua no sentido de prover abstrações que facilitem a construção de aplicações de C&A. Integrada com outras abstrações, como a camada de comunicação provida pela infra-estrutura INCA e o modelo de armazenamento do serviço StRES, a xINCA compõe os passos (i), (ii) e (iii) de um modelo simplificado para o desenvolvimento de aplicações de C&A baseado na (i) criação de abstrações para a classe de aplicações especificada, (ii) no desenvolvimento de soluções arquiteturais para dar apoio ao projeto e à construção dessas aplicações, (iii) na implementação de infra-estruturas e serviços que englobem essas abstrações, e (iv) na validação das abstrações, arquitetura e infra-estruturas por meio do desenvolvimento de aplicações dentro do espaço de projeto delimitado. O passo (iv) corresponde ao desenvolvimento das aplicações iClass e CAT, implementadas sobre as infra-estruturas e serviços desenvolvidos.

7.2 Resultados e Contribuições

Dentre os principais resultados e contribuições resultantes da conclusão dos objetivos propostos, estão:

- Validação dos objetivos propostos pela infra-estrutura INCA, salientando algumas de suas vantagens — consistência no projeto de novas aplicações e existência de serviços auxiliares para monitoração e gerenciamento — e algumas de suas desvantagens — como possíveis problemas de escalabilidade;
- Implementação da infra-estrutura xINCA, como uma camada de abstração acima da INCA, permitindo o desenvolvimento de aplicações de C&A por meio do agrupamento e configuração de componentes de software reutilizáveis;
- Desenvolvimento de aplicações reais, em especial o Sistema iClass, que tem sido utilizado no ICMC-USP desde o segundo semestre de 2002, período em que foram capturadas mais de 180 aulas de 10 disciplinas de graduação e pós-graduação. Esse uso estendeu-se para outras instituições, como as Faculdades COC de Ribeirão Preto, e há previsões de instalações em outras unidades da USP, no Georgia Tech e na Universidade de Wisconsin Milwaukee. O iClass também possui um versão para demonstração, com lousa eletrônica SmartBoardTM, instalada no Centro de Computação Eletrônica da USP;
- Contribuição, para o Projeto InCA-SERVE, como uma das infra-estruturas desenvolvidas, compondo parte de uma solução de *middleware* para construção de aplicações de C&A (Cattelan et al., 2003c; Baldochi et al., 2003);
- Colaboração na definição do modelo de estruturação da informação capturada em documentos XML especificados por esquemas XML e na caracterização de aplicações de C&A (Cattelan et al., 2003b);
- Contribuição para divulgação da área de computação ubíqua (Baldochi et al., 2002a);
- Co-autoria na realização de um mini-curso de divulgação documentando o uso do iClass nos cursos do ICMC-USP. Esse mini-curso foi realizado na UFRJ como parte do programa do Simpósio Brasileiro de Informática na Educação (SBIE 2003), resultando ainda na publicação de um capítulo no livro de mini-cursos do evento (Pimentel et al., 2003).

7.3 Limitações

Como parte integrante da solução de *middleware* desenvolvida no contexto do Projeto InCA-SERVE, a infra-estrutura xINCA apresenta algumas limitações:

- O componente xINCA de *whiteboard* não suporta o uso de animações (por exemplo, GIFs animados ou apresentações PowerPointTM) nas imagens carregadas em *background*, restrição compensada em parte pelas amplas possibilidades de interação fornecidas pelo componente que permite ao usuário desenhar a animação durante a sessão, o que geralmente economiza tempo na preparação de material na fase de pré-produção;
- Devido a limitações no modelo de comunicação empregado pela INCA, os componentes xINCA de áudio e de vídeo não suportam transmissão “ao vivo” (*streaming*) para audiências remotas;
- Embora não tenha sido conduzido estudo sobre desempenho da solução desenvolvida, a mesma deve enfrentar problemas no que diz respeito à escalabilidade de sistemas envolvendo muitos componentes, uma vez que a infra-estrutura de rede segue o modelo *publish-subscribe* com toda comunicação passando por uma entidade central, o Registry, que eventualmente pode se tornar um gargalo da infra-estrutura. Isso pode ser contornado, em parte, pela replicação de Registries e por uma distribuição equalitária dos componentes entre os mesmos, mas com a inconveniência de que ambos são procedimentos manuais.

7.4 Trabalhos Futuros

Uma possível contituidade deste trabalho pode se dar com uma avaliação de desempenho da solução adotada, principalmente com análise da escalabilidade do sistema como um todo frente a condições de estresse e ao uso de muitos componentes.

A arquitetura desenvolvida pode ainda ser adequada ao modelo de computação *peer-to-peer*, visando formas de interação mais complexas e a aquisição de um comportamento autônomo, com o estabelecimento de conexões entre componentes de maneira dinâmica e a eliminação total ou parcial de entidades centralizadas.

A API da xINCA também pode ser estendida e portada para suportar dispositivos móveis. Neste caso, são necessárias algumas adaptações no código para sua adequação à especificação de uma máquina virtual portátil, tal como J2ME¹ ou SuperWaba². Também são necessárias

¹<http://java.sun.com/j2me>.

²<http://www.superwaba.com.br>.

alterações na camada de rede, principalmente para a inclusão de funcionalidade para redes sem fio.

Referências

- Abowd, G. (1999a). Classroom 2000: An experiment with the instrumentation of a living educational environment. *IBM Systems Journal*, 38(4):508–530.
- Abowd, G. (1999b). Software engineering issues for ubiquitous computing. In *Proceedings of the 21st International Conference on Software Engineering*, pages 75–83.
- Abowd, G., Atkeson, C., Feinstein, A., Hmelo, C., Kooper, R., Long, S., Sawhney, N., & Tan, M. (1996). Teaching and learning as multimedia authoring: The classroom 2000 project. In *Proceedings of the 1996 ACM Multimedia Conference*, pages 187–198.
- Abowd, G. D. & Mynatt, E. D. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction*, 7(1):29–58.
- Andrade, A. (2003). Visualização e apresentação de sessões capturadas em ambientes de computação ubíqua. Monografia de Qualificação de Mestrado. 36p.
- Baldochi, L., Cattelan, R., Arruda Jr, C., & Pimentel, M. (2002a). Computação ubíqua: Fundamentos e exemplos. In *Anais da IV Escola Regional de Informática*.
- Baldochi, L., Cattelan, R., & Pimentel, M. (2003). Building a middleware infrastructure for capture and access applications. In *Anais do XXX Seminário Integrado de Software e Hardware*, pages 299–313.
- Baldochi, L., Cattelan, R., Pimentel, M., & Truong, K. (2002b). Automatic generation of capture and access applications. In *Proceedings of the 8th Brazilian Symposium on Multimedia and Hypermedia Systems*, pages 100–115.
- Berque, D. (1999). Using a variation of the wysiwiw shared drawing surface paradigm to support electronic classrooms. In *Human Computer Interaction '99: The 8th International Conference on Human Computer Interaction*.

- Bianchi, M. (1998). Autoauditorium: A fully automatic, multi-camera system to televise auditorium presentations. In *Proceedings of DARPA/NIST Smart Spaces Technology Workshop*.
- Boehm, B. & Scherlis, W. (1992). Megaprogramming. In *Proceedings of the DARPA Software Technology Conference*, pages 63–82.
- Brotherton, J. & Abowd, G. (1998). Rooms take note: Room take notes! In *Working Papers of AAAI Spring Symposium*.
- Brotherton, J., Abowd, G., & Truong, K. (1999). Supporting capture and access interfaces for informal and opportunistic meetings. Technical report GIT-GVU-99-06, GVU Center, Georgia Institute of Technology.
- Brotherton, J., Bhalodia, J., & Abowd, G. (1998). Automated capture, integration, and visualization of multiple media streams. In *Proceedings of the IEEE Multimedia*, pages 54–63.
- Brumitt, B., Meyers, B., Krumm, J., Kern, A., & Shafer, S. (2000). Easyliving: Technologies for Intelligent Environments. In *Proceedings of the Second International Symposium on Hand-held and Ubiquitous Computing (HUC)*, pages 12–29.
- Bulterman, D. C. A. (2003). Using smil to encode interactive, peer-level multimedia annotations. In *Proceedings of the 2003 ACM symposium on Document engineering*, pages 32–41. ACM Press.
- Cattelan, R., Andrade, A., Rocha, C., & Pimentel, M. (2003a). iclass: um sistema para captura e acesso de sessões em ambiente educacional. *Revista Eletrônica de Iniciação Científica da SBC*, 3(1):10–28.
- Cattelan, R., Baldochi, L., & Pimentel, M. (2003b). Experiences on building capture & access applications. In *Proceedings of the 9th Brazilian Symposium on Multimedia System and Web*, pages 298–312.
- Cattelan, R., Baldochi, L., & Pimentel, M. (2003c). Processing and storage middleware support for capture and access applications. In *Companion Proceedings of the 2003 ACM/IFIP/USENIX International Middleware Conference*, page 315.
- Chen, A., Muntz, R., Yuen, S., Locher, I., Sung, S., & Srivastava, M. (2002). A support infrastructure for the smart kindergarten. *IEEE Pervasive Computing*, 1(2):49–57.
- Chiu, P., Boreczky, J., Girgensohn, A., & Kimber, D. (2001). LiteMinutes: an Internet-Based System for Multimedia Meeting Minutes. In *Proceedings of the 2001 International World Wide Web Conference*, pages 140–149.

-
- Dey, A. (2001). Understanding and using context. *Personal and Ubiquitous Computing Journal*, 1(5):4–7.
- Dey, A., Futakawa, M., Salber, D., & Abowd, G. (1999). The Conference Assistant: Combining Context-Awareness with Wearable Computing. In *Proceedings of the 1999 International Symposium on Wearable Computers*, pages 21–28.
- D’Souza, D. & Wills, A. (2001). *Objects, Components, and Frameworks with UML*. Addison Wesley, fourth edition.
- Edwards, W., Bellotti, V., Dey, A., & Newman, M. (2003). Stuck in the middle: The challenges of user-centered design and evaluation for infrastructure. *CHI Letters*, 5(1):297–304.
- Fleck, M., Frid, M., Kindberg, T., O’Brien-Strain, E., Rajani, R., & M., S. (2002). Rememberer: A tool for capturing museum visits. In *Proceedings of the 2002 International Conference on Ubiquitous Computing*, pages 48–55.
- Garlan, D., Allen, R., & Ockerbloom, J. (1995). Architectural mismatch or why it’s hard to build systems out of existing parts. In *Proceedings of the 1995 International Conference on Software Engineering*, pages 179–185.
- Garlan, D., Siewiorek, D., Smailagic, A., & Steenkiste, P. (2002). Project Aura: Toward Distraction-Free Pervasive Computing. *IEEE Pervasive Computing*, 2(1):22–31.
- Greenberg, S., Boyle, M., & LaBerge, J. (1999). Pdas and shared public displays: Making personal information public, and public information personal. *Personal Technologies*, 1(3):54–64.
- Griswold, W. G., Boyer, R., Brown, S. W., & Truong, T. M. (2003). A component architecture for an extensible, highly integrated context-aware computing infrastructure. In *Proceedings of the 25th international conference on Software engineering*, pages 363–372. IEEE Computer Society.
- Harrison, B., Fishkin, K., Gujar, A., Mochon, C., & Want, R. (1998). Squeeze me, hold me, tilt me! an exploration of manipulative user interfaces. In *Proceedings of the 1998 Conference on Human Factors in Computing Systems*, pages 17–24.
- Hauptmann, A. (1995). Speech recognition in the informedia digital video library: Uses and limitations. In *Proceedings of the 1995 IEEE International Conference on Tools with AI*.
- Heiner, J., Hudson, S., & Tanaka, K. (1999). The information percolator: Ambient information display in a decorative object. In *Proceedings of the 1999 Symposium on User Interface Software and Technology*, pages 141–148.

- Hess, C. K., Román, M., & Campbell, R. H. (2002). Building applications for ubiquitous computing environments. In *Proceedings of the 2002 International Conference on Pervasive Computing*, pages 16–29.
- Ishii, H. & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the 1997 Conference on Human Factors in Computing Systems*, pages 234–241.
- Johanson, B., Fox, A., & Winograd, T. (2002). The interactive workspaces project: Experiences with ubiquitous computing rooms. *IEEE Pervasive Computing*, 2(1):71–78.
- Mankoff, J., Dey, A., Hsieh, G., Kientz, J., Lederer, S., & Ames, M. (2003). Heuristic evaluation of ambient displays. In *Proceedings of the 2003 ACM Conference on Human Factors in Computing Systems*, pages 169–176.
- Meier, W. (2003). eXist open source XML database. Available in: <http://exist-db.org>.
- Minneman, S., Harrison, S., Janseen, B., Kurtenback, G., Moran, T., Smith, I., & Van Melle, B. (1995). A confederation of tools for capturing and accessing collaborative activity. In *Proceedings of the 1995 ACM Multimedia Conference*, pages 523–534.
- Mukhopadhyay, S. & Smith, B. (1999). Passive capture and structuring of lectures. In *Proceedings of the 1999 ACM Multimedia Conference*, pages 477–487.
- Mynatt, E., T., I., Edwards, W., & LaMarca, A. (1999). Flatland: New dimensions in office whiteboards. In *Proceedings of the 1999 ACM Conference on Human Factors in Computing Systems*, pages 346–353.
- NCSA Software Development Division (1996). Habanero(TM) NCSA Home Page. Available in: <http://havefun.ncsa.uiuc.edu/habanero>.
- Pedersen, E., McCall, K., Moran, T., & Halasz, F. (1993). Tivoli: An electronic whiteboard for informal workgroup meetings. In *Proceedings of the 1993 ACM INTERCHI Conference on Human Factors in Computing Systems*, pages 391–398.
- Pimentel, M., Abowd, G., & Ishiguro, Y. (2000). Linking by interacting: a paradigm for authoring hypertext. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*, pages 39–48.
- Pimentel, M., Cattelan, R., & Fortes, R. (2003). Computação ubíqua em sala de aula: experiências e oportunidades. In *Mini-cursos do XIV SBIE - Inclusão digital como instrumento de inclusão social*, pages 49–72. Ed. Rio de Janeiro.

-
- Pimentel, M., Fortes, R., & Moreira, E. (2001a). Development and understanding of automated capture environments to support long-term use: Proposal 1999. Technical report 149, ICMC-USP.
- Pimentel, M., Ishiguro, Y., Kerimbaev, B., Abowd, G., & Guzdial, M. (2001b). Supporting educational activities through dynamic web interfaces. *Interacting with Computers*, 13(3):353–374.
- Richter, H., Abowd, G., Geyer, W., Fuchs, L., Daijavad, S., & Poltrock, S. (2001). Integrating meeting capture within a collaborative team environment. In *Proceedings of the 3rd International Conference on Ubiquitous Computing*, pages 123–138.
- Roman, G.-C., Julien, C., & Huang, Q. (2002a). Network abstractions for context-aware mobile computing. In *Proceedings of the 24th international conference on Software engineering*, pages 363–373. ACM Press.
- Roman, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R., & Nahrstedt, K. (2002b). Gaia: A middleware platform for active spaces. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(4):65–67.
- Roseman, M. & Greenberg, S. (1996). Teamrooms: Network places for collaboration. In *Proceedings of the 1996 ACM Conference on Computer Supported Cooperative Work*, pages 325–333.
- Sante, D. (2003). Autore: suportando autoria evolucionária em ambientes de captura. Dissertação de Mestrado do ICMC-USP. 95p.
- Schilit, B., Golovchinsky, G., & Price, M. (1998). Beyond paper: Supporting active reading with free form digital ink annotations. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 249–256.
- Shi, Y., Xie, W., Xu, G., Shi, R., Chen, E., Mao, Y., & Liu, F. (2003). The Smart Classroom: Merging Technologies for Seamless Tele-Education. *IEEE Pervasive Computing*, 2(2):47–55.
- Shirmohammadi, S. & Georganas, N. (1997). JETS: a java-enabled telecollaboration system. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 541–547.
- Streitz, N., Geibler, J., Haake, J., & Hol, J. (1994). DOLPHIN: integrated meeting support across local and remote desktop environments and liveboards. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 345–358.

- Streitz, N., Tandler, P., Müller-Tomfelde, C., & Konomi, S. (2001). Roomware: Towards the next generation of human-computer interaction based on an integrated design of real and virtual worlds. In Carroll, J., editor, *Human-Computer Interaction in the New Millenium*, pages 553–578. Addison-Wesley.
- Truong, K. & Abowd, G. (2002). Enabling the generation, preservation & use of records and memories of everyday life. Technical report GATECH GIT-GVU-02-02, GVU Center, Georgia Institute of Technology.
- Truong, K. & Abowd, G. (2004). Inca: A software infrastructure to facilitate the construction and evolution of ubiquitous capture & access applications. In *Proceedings of the 2004 International Conference on Pervasive Computing*. To appear.
- Truong, K., Abowd, G., & Brotherton, J. (1999). Personalizing the capture of public experiences. In *Proceedings of the 1999 Symposium on User Interface Software and Technology*, pages 121–130.
- Truong, K., Abowd, G., & Brotherton, J. (2001). Who, what, when, where, how: Design issues of capture & access applications. In *Proceedings of the 2001 International Conference on Ubiquitous Computing*, pages 209–224.
- Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265(3):94–104.
- Weiser, M. & Brown, J. (1996). The coming age of calm technology. Xerox PARC.
- White, D., Camacho-Guerreiro, J., Truong, K., Abowd, G., M., M., Vekaria, P., & Gromala, D. (2003). Mobile capture and access for assessing language and social development in children with autism. In *Extended Abstracts of the 2003 International Conference on Ubiquitous Computing*, pages 137–140.
- Wilcox, L., Schilit, B., & Sawhney, N. (1997). Dynamite: A dynamic organized ink and audio notebook. In *Proceedings of the 1997 ACM Conference on Human Factors in Computing Systems*, pages 186–193.

Diagramas de Classes UML da xINCA

Figura A.1: Diagrama de classes UML do componente xINCA de áudio.

Figura A.2: Diagrama de classes UML do modelo de dados do componente xINCA de áudio.

Figura A.3: Diagrama de classes UML do componente xINCA de texto.

Figura A.4: Diagrama de classes UML do modelo de dados do componente xINCA de texto.

Figura A.5: Diagrama de classes UML do componente xINCA de vídeo.

Figura A.6: Diagrama de classes UML do modelo de dados do componente xINCA de vídeo.

Figura A.7: Diagrama de classes UML do componente xINCA de *web logging*.

Figura A.8: Diagrama de classes UML do modelo de dados do componente xINCA de *web logging*.

Figura A.9: Diagrama de classes UML do componente xINCA de *whiteboard*.

Figura A.10: Diagrama de classes UML do modelo de dados do componente xINCA de *white-board*.

Esquemas XML dos Componentes xINCA

B.1 Esquema XML do Componente xINCA de Áudio

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="audioml" type="AudiomlType"/>
  <xsd:complexType name="AudiomlType">
    <xsd:sequence>
      <xsd:element name="audio" minOccurs="0" maxOccurs="unbounded" type="Audio"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Audio">
    <xsd:sequence>
      <xsd:element name="title" type="xsd:string" minOccurs="0"/>
      <xsd:element name="url" type="xsd:string"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:int" use="required"/>
    <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
    <xsd:attribute name="person-id" type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>
```

B.2 Esquema XML do Componente xINCA de Texto

```
<?xml version="1.0"?>
```

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="chatml" type="ChatmlType"/>
  <xsd:complexType name="ChatmlType">
    <xsd:sequence>
      <xsd:element name="msg" minOccurs="0" maxOccurs="unbounded" type="Msg"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Msg">
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute name="id" type="xsd:int" use="required"/>
        <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
        <xsd:attribute name="person-id" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:schema>

```

B.3 Esquema XML do Componente xINCA de Vídeo

```

<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="videoml" type="VideomlType"/>
  <xsd:complexType name="VideomlType">
    <xsd:sequence>
      <xsd:element name="video" minOccurs="0" maxOccurs="unbounded" type="Video"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Video">
    <xsd:sequence>
      <xsd:element name="title" type="xsd:string" minOccurs="0"/>
      <xsd:element name="url" type="xsd:string"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:int" use="required"/>
    <xsd:attribute name="video-width" type="xsd:int" use="required"/>
    <xsd:attribute name="video-height" type="xsd:int" use="required"/>
    <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
    <xsd:attribute name="person-id" type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>

```

B.4 Esquema XML do Componente xINCA de Web Logging

```

<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="weblogml" type="WeblogmlType"/>
  <xsd:complexType name="WeblogmlType">
    <xsd:sequence>

```

B.5 Esquema XML do Componente xINCA de Whiteboard

```
<xsd:element name="log" minOccurs="0" maxOccurs="unbounded" type="Log"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Log">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="id" type="xsd:int" use="required"/>
      <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
      <xsd:attribute name="person-id" type="xsd:string"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
</xsd:schema>
```

B.5 Esquema XML do Componente xINCA de Whiteboard

```
<?xml version="1.0"?>
<xsd:schema xmlns:jxb="http://java.sun.com/xml/ns/jaxb"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" jxb:version="1.0">
  <xsd:element name="whiteboardml" type="WhiteboardmlType"/>
  <xsd:complexType name="WhiteboardmlType">
    <xsd:sequence>
      <xsd:element name="slides" type="Slides"/>
      <xsd:element name="visits" type="Visits"/>
    </xsd:sequence>
    <xsd:attribute name="board-width" type="xsd:int" use="required"/>
    <xsd:attribute name="board-height" type="xsd:int" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="Slides">
    <xsd:sequence>
      <xsd:element name="slide" minOccurs="0" maxOccurs="unbounded" type="Slide"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Visits">
    <xsd:sequence>
      <xsd:element name="visit" minOccurs="0" maxOccurs="unbounded" type="Visit"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Slide">
    <xsd:sequence>
      <xsd:element name="title" type="xsd:string" minOccurs="0"/>
      <xsd:element name="text" minOccurs="0" type="Text"/>
      <xsd:element name="url" type="xsd:string"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:int"/>
  </xsd:complexType>
  <xsd:complexType name="Text">
    <xsd:sequence>
      <xsd:element name="sentence" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="annotation" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
```

```

</xsd:complexType>
<xsd:complexType name="Visit">
  <xsd:sequence>
    <xsd:element name="strokes" type="Strokes"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:int" use="required"/>
  <xsd:attribute name="duration" type="xsd:long" use="required"/>
  <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
  <xsd:attribute name="slide-id" type="xsd:int" use="required"/>
</xsd:complexType>
<xsd:complexType name="Strokes">
  <xsd:sequence>
    <xsd:element name="stroke" minOccurs="0" type="Stroke"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Stroke">
  <xsd:sequence>
    <xsd:element name="min-x" type="xsd:int"/>
    <xsd:element name="min-y" type="xsd:int"/>
    <xsd:element name="max-x" type="xsd:int"/>
    <xsd:element name="max-y" type="xsd:int"/>
    <xsd:element name="points" type="Points"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:int" use="required"/>
  <xsd:attribute name="time-offset" type="xsd:long" use="required"/>
  <xsd:attribute name="deletion-time-offset" type="xsd:long"/>
  <xsd:attribute name="color" type="Color" use="required"/>
  <xsd:attribute name="width" type="xsd:int" use="required"/>
  <xsd:attribute name="person-id" type="xsd:string" use="required"/>
</xsd:complexType>
<xsd:simpleType name="Color">
  <xsd:annotation>
    <xsd:appinfo>
      <jxb:javaType name="java.awt.Color"
        parseMethod="br.usp.icmc.xinca.data.xml.bind.TransColor.parseColor"
        printMethod="br.usp.icmc.xinca.data.xml.bind.TransColor.printColor"/>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:restriction base="xsd:string"/>
</xsd:simpleType>
<xsd:simpleType name="Points">
  <xsd:annotation>
    <xsd:appinfo>
      <jxb:javaType name="br.usp.icmc.xinca.data.Points"
        parseMethod="br.usp.icmc.xinca.data.xml.bind.TransPoints.parsePoints"
        printMethod="br.usp.icmc.xinca.data.xml.bind.TransPoints.printPoints"/>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:restriction base="xsd:string"/>
</xsd:simpleType>
</xsd:schema>

```