4
Montagem de Groupware e de Serviços Colaborativos

Neste capítulo é descrita a abordagem utilizada, discutindo os tipos de componentes adotados, os *component kits* propostos, a arquitetura e o modelo de componentes utilizados, a maneira de descrever os componentes, de instanciar groupware e de utilizar frameworks do domínio para complementar a solução proposta.

4.1.
Componentes de Groupware e de Colaboração

Um groupware normalmente oferece ao participante um ambiente com um conjunto de ferramentas colaborativas, utilizadas nos diferentes momentos da colaboração. A Tabela 4.1 apresenta as ferramentas encontradas nos sistemas de groupware: AulaNet[8], TelEduc[9], AVA[10], WebCT[11] e Moodle[12], do domínio educacional, e GroupSystems[13], YahooGroups[14], OpenGroupware[15] e BSCW[16], voltados para o trabalho em grupo.

[8] http://www.eduweb.com.br
[9] http://teleduc.nied.unicamp.br
[10] http://ava.unisinos.br
[11] http://www.webct.com
[12] http://www.moodle.org
[13] http://www.groupsystems.com
[14] http://groups.yahoo.com
[15] http://www.opengroupware.org
[16] http://bscw.fit.fraunhofer.de
Data de consulta: 15 de janeiro de 2006

	Serviços de Comunicação							Serviços de Coordenação								Serviços de Cooperação											
	Correio	Lista de Discussão	Fórum	Mural	Brainstorming	Chat	Mensageiro	Agenda	Relat de Atividades	Acomp. da Particip.	Questionário	Tarefas	SubGrupos	Gerenc. de recursos	Orientação	Votação	Reposit. de Conteúdos	Quadro Branco	Busca	Glossário	Links	Jornal Cooperativo	Classificador	Wiki	Gerenc. de contatos	Revisão em pares	FAQ	Anotações	RSS
AulaNet	X	X	X			X	X		X	X	X	X	X			X	X				X							X	
TelEduc	X		X	X		X		X	X	X	X	X	X				X				X						X	X	
AVA	X	X	X	X		X		X	X	X	X	X			X		X			X	X						X	X	
WebCT	X		X			X		X	X		X	X				X	X	X	X	X								X	
Moodle			X			X	X	X	X	X	X	X	X			X	X		X	X	X	X		X		X	X		X
GroupSystems			X		X			X		X	X	X	X			X							X					X	
YahooGroups		X				X		X	X		X					X	X		X	X	X								X
OpenGroupware	X			X				X				X		X											X				
BSCW			X					X	X			X	X			X	X		X		X				X					

Tabela 4.1. Ferramentas colaborativas encontradas em groupware

Conforme observado na tabela, diversas ferramentas similares são utilizadas em groupware. Por exemplo, a maioria dos sistemas analisados oferece fórum, chat, agenda, relatórios de atividades, questionários, gerenciamento de tarefas, votação, repositório e links. Cada ferramenta pode ser vista de forma relativamente isolada dentro do ambiente. Estas características são propícias à aplicação de técnicas de desenvolvimento baseado em componentes, onde as ferramentas colaborativas são os componentes do groupware. Os ambientes oferecem um conjunto de componentes que é instanciado e customizado para cada grupo e dinâmica da colaboração.

Nesta tese, as ferramentas colaborativas são chamadas de serviços. Conforme discutido na seção anterior, é possível classificar os serviços de acordo com seus propósitos e características em função do modelo 3C. Os serviços da Tabela 4.1 são organizados em serviços de comunicação, coordenação e cooperação. Em um ambiente baseado em componentes, a partir das necessidades da colaboração no grupo, o desenvolvedor seleciona os serviços mais adequados. Um modelo de componentes unificado para os diversos groupwares possibilita ao desenvolvedor selecionar dentre os diversos serviços de mesmo propósito o mais adequado. Conforme pode-se notar na Tabela 4.1, há serviços que são oferecidos em apenas um groupware, mas que, a princípio, poderiam ser disponibilizados para utilização nos demais. Para os usuários, é interessante intercambiar serviços, de modo a complementar os ambientes e reusar experiências.

Além dos groupwares possuírem serviços similares, os serviços possuem funcionalidades similares. Nos serviços dos diversos groupwares analisados, as funcionalidades são recorrentes. A Figura 4.1 apresenta os chats do Moodle e do WebCT. Pode-se notar que ambos possuem uma área compartilhada onde as mensagens são apresentadas, uma lista de participantes conectados e uma área para elaboração das mensagens. Ao utilizar a componentização, estas características são reusadas.

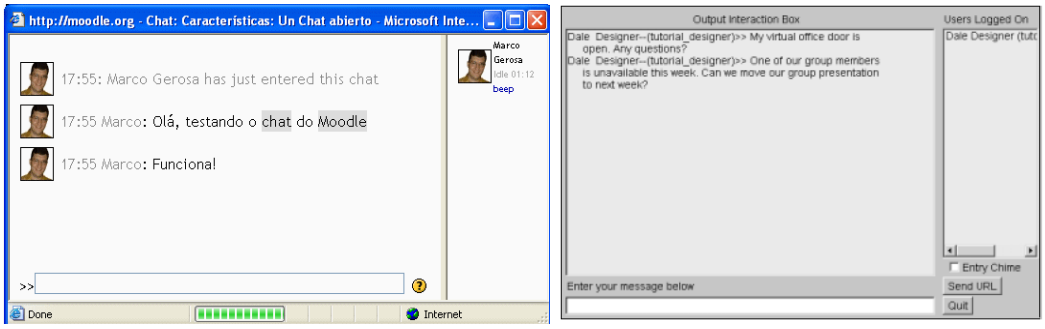

Figura 4.1. Chat do Moodle e do WebCT

Diferentemente dos demais, o chat do Moodle apresenta o tempo em que o participante está inativo, um recurso para chamar a atenção de um participante (beep) e a foto de cada um. O WebCT apresenta suporte para mensagens privadas e notificação de quando alguém entra na sala de bate papo. Encapsular estas funcionalidades em componentes possibilita ao desenvolvedor da aplicação disponibilizar para reuso os diversos aspectos do suporte à colaboração, de modo que outros desenvolvedores utilizem-nos na seleção das funcionalidades mais apropriadas para os grupos e atividades em questão.

O serviço Conferências e o serviço Correio para Turma do AulaNet, exibidos na Figura 4.2, compartilham o suporte ao envio, ao recebimento e à exibição de mensagens, à categorização, à avaliação da participação e ao bloqueio do canal de comunicação, entre outras funcionalidades. Encapsular as funcionalidades recorrentes em componentes propicia também o reuso do suporte computacional à colaboração de cada serviço, aumentando o reuso de código. Passa também a ser possível evoluir, ajustar e construir serviços variando e reconfigurando os componentes de colaboração.

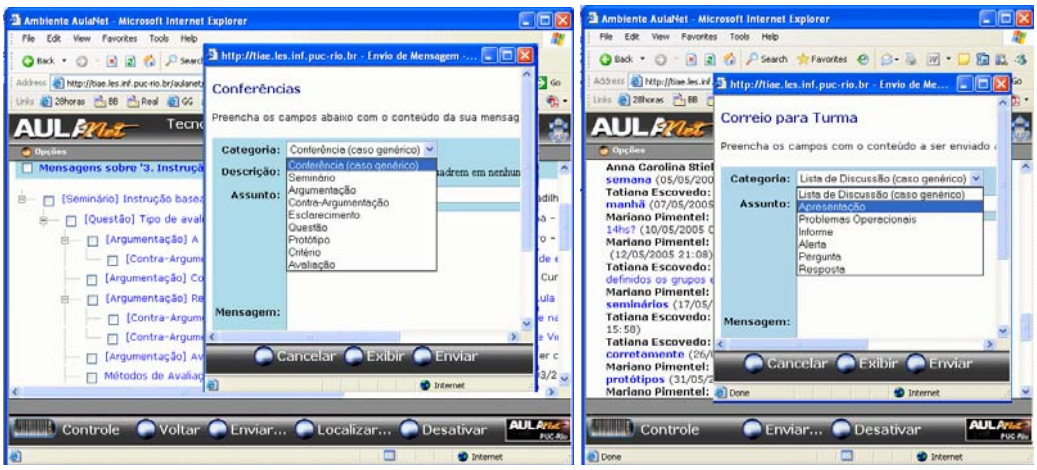

Figura 4.2. Serviços Conferências e Correio para Turma do AulaNet

A Figura 4.3 apresenta os serviços Bate Papo e Debate do AulaNet. A principal diferença entre estes dois serviços é o suporte à coordenação, que no primeiro é implementado apenas pela lista de participantes, enquanto no segundo é implementado pelo controle do espaço compartilhado, pelas técnicas de conversação e pela definição da ordem de participação. Em um serviço baseado em componentes, o suporte computacional à comunicação é compartilhado e os componentes que oferecem suporte computacional à coordenação são acrescentados, provendo reuso e especialização.

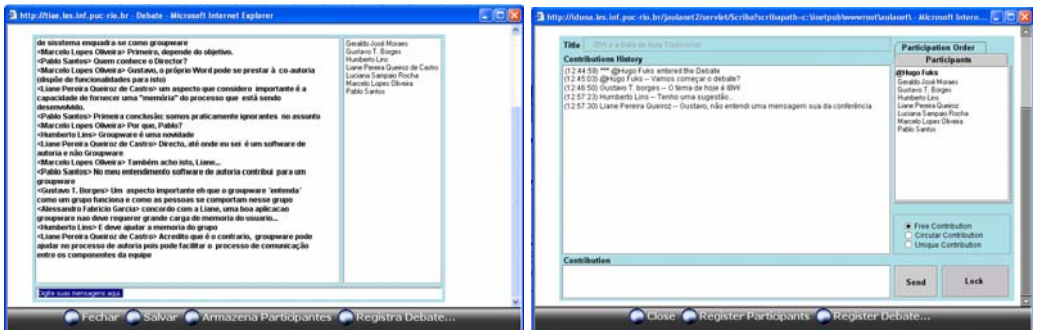

Figura 4.3. Serviços Bate-Papo e Debate do ambiente AulaNet

Estes cenários indicam a necessidade de se adotar a componentização de software em dois níveis. O primeiro nível contempla os componentes que provêem os serviços colaborativos, utilizados para oferecer suporte computacional à dinâmica da colaboração como um todo. O segundo nível contempla os componentes utilizados para montar os serviços, oferecendo suporte a determinados aspectos da colaboração dentro da dinâmica de uma atividade em particular. Na abordagem proposta, os componentes que implementam as ferramentas colaborativas são chamados de *serviços* e os componentes utilizados

para implementar o suporte computacional à colaboração dos serviços são chamados de *componentes de colaboração*.

Conforme ilustrado na Figura 4.4, um groupware é composto de serviços, que são reusáveis em diversos outros groupwares. Os serviços compartilham componentes de colaboração que implementam o suporte à colaboração, que nesta tese é modelada através do modelo 3C. A partir de component kits organizados em função do modelo 3C, o desenvolvedor monta uma aplicação para dar suporte à dinâmica da colaboração. Além do reuso, esta abordagem favorece também a capacidade de extensão da solução, ao possibilitar a inclusão de novos componentes.

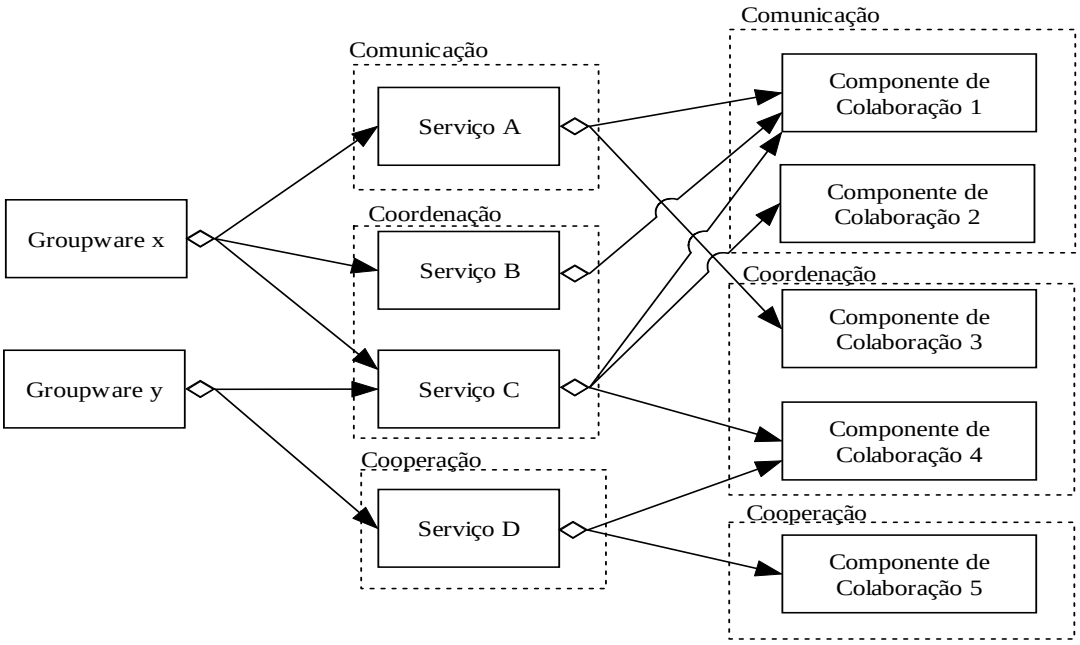

Figura 4.4. Groupwares montados a partir de serviços, e serviços montados a partir de componentes de colaboração

Conforme discutido no capítulo anterior, mesmo um serviço de comunicação, como um fórum de discussão, além dos componentes de comunicação, utiliza componentes de coordenação e de cooperação. Os componentes de colaboração de um C são reusados nos serviços dos demais Cs.

4.2. O Collaboration Component Kit

O objetivo da abordagem proposta é instrumentar o desenvolvedor com *component kits* para serem usados na montagem dos groupwares e dos serviços colaborativos. Para obter o conjunto de componentes é necessária uma engenharia do domínio. A análise do domínio, primeiro passo da engenharia, foi elaborada neste trabalho com base na experiência do grupo desenvolvedor do AulaNet, que atua há 8 anos no desenvolvimento de ferramentas para colaboração, na literatura e na análise de ferramentas colaborativas. Entre outras, foram analisadas as ferramentas de comunicação dos diversos groupware apresentados na Tabela 4.1. A análise do domínio foi restrita às ferramentas de comunicação, que além dos elementos de comunicação, apresentam uma representatividade de elementos de coordenação e de cooperação.

A análise das ferramentas foi guiada pelo modelo 3C, sendo analisadas sob a perspectiva da comunicação, coordenação e cooperação. Os modelos de características foram diagramados na notação Odyssey-FEX, que é implementada no ambiente de modelagem Odyssey [Oliveira et al., 2005]. O Odyssey é um ambiente voltado para o suporte à engenharia de domínio e à engenharia de aplicações e oferece suporte computacional ao processo CBD-Arch-DE [Blois et al., 2004]. Na notação Odyssey-FEX um retângulo fechado simboliza uma característica obrigatória e um tracejado, uma característica opcional.

O modelo de características (*feature model*) apresentado na Figura 4.5 sumariza a análise do domínio da comunicação. Uma ferramenta de comunicação possui mensagens, que utilizam mídia textual, vídeo, áudio ou pictórica [Daft & Lengel, 1986]. Eventualmente, as mídias apresentam alguma variabilidade, como restrições ao tamanho do texto ou ao vocabulário disponível, no caso da mídia textual, e à taxa de captura e de envio de dados, no caso do áudio e vídeo. Algumas ferramentas disparam e-mail aos destinatários, possibilitam anexar arquivos às mensagens e disponibilizam um corretor ortográfico para auxiliar na elaboração do texto. Algumas ferramentas, como as Conferências e o Correio para Turma do AulaNet, oferecem a categorização de mensagens [Gerosa et al., 2001]. Também é encontrado em ferramentas de comunicação o suporte a carteiras de compromissos, como no ACCORD [Laufer & Fuks, 1995], e a caminhos de

conversação, como no Coordinator [Winograd & Flores, 1987]. As mensagens de uma ferramenta são organizadas em uma estrutura de diálogo, linear, hierárquica ou em rede [Stahl, 2001]. As mensagens são transmitidas em blocos ou continuamente.

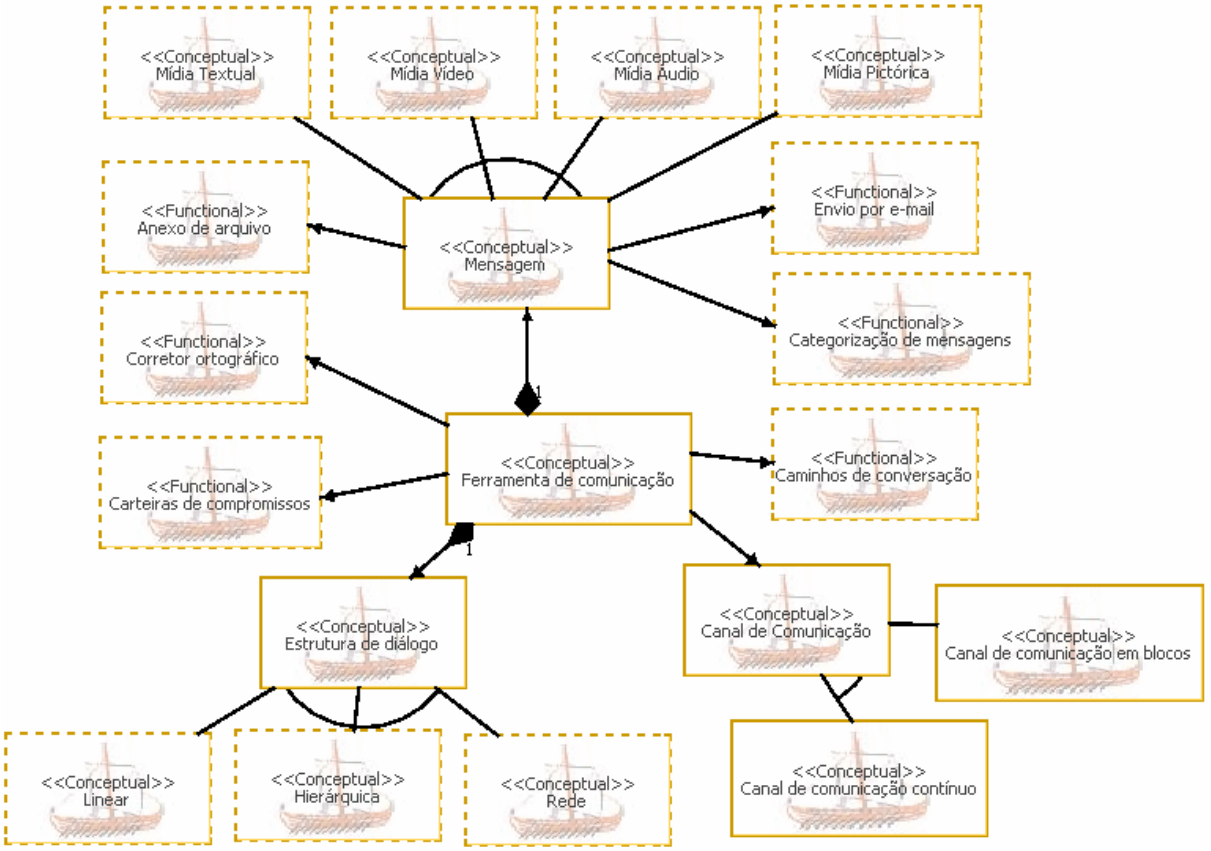

Figura 4.5. Modelo de características da comunicação nas ferramentas de comunicação

Conforme discutido no capítulo anterior, o suporte à coordenação em uma ferramenta de comunicação está relacionado às políticas de acesso ao canal, à gestão de tarefas e participantes e ao acompanhamento da participação. O modelo da Figura 4.6 sumariza a análise da coordenação nas ferramentas de comunicação. As ferramentas de comunicação apresentam um gerenciamento de permissões de acesso, associado a participantes ou a papéis. Os papéis são utilizados associados a tarefas, no escopo de uma atividade. Eventualmente, as tarefas são acompanhadas através de uma máquina de workflow. As participações dos indivíduos, principalmente em ferramentas de comunicação ligadas ao ensino-aprendizagem, são avaliadas, de modo a prover dados para o acompanhamento qualitativo da participação [Fuks et al., 2003]. A avaliação de mensagens provê subsídio para a gestão de competência dos participantes, que é utilizada para definir a dinâmica das atividades, a associação de papéis e a definição de

subgrupos. A participação ocorre no contexto de uma sessão e é embasada em informações de percepção, como por exemplo, a informação de que um determinado participante está escrevendo uma mensagem. Algumas ferramentas disponibilizam informações sobre a presença e a disponibilidade dos participantes, de modo a propiciar uma melhor sincronização da participação dos interlocutores. Um mecanismo de coordenação implementado por software encontrado em algumas ferramentas de comunicação, como o Debate do AulaNet, é o controle de palco, que possibilita implantar técnicas de conversação e políticas de acesso ao canal [Rezende et al., 2003].

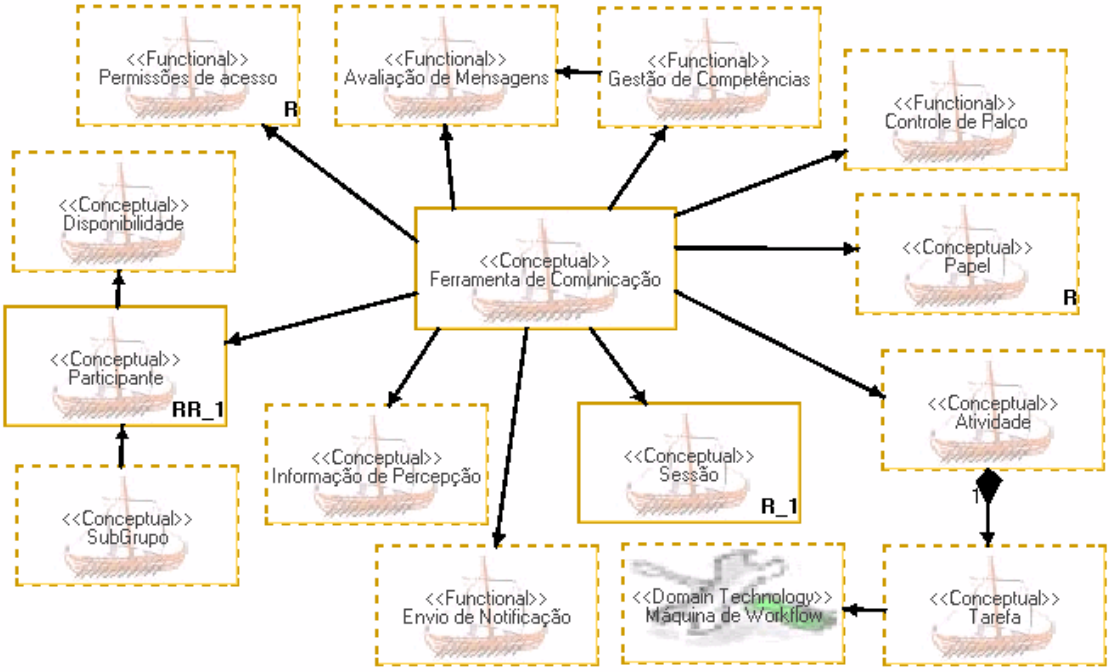

Figura 4.6. Modelo de características da coordenação nas ferramentas de comunicação

O suporte à cooperação em uma ferramenta de comunicação está relacionado ao registro e à manipulação das informações. O modelo de características da cooperação está apresentado na Figura 4.7. As mensagens ou as sessões das ferramentas de comunicação são registradas em repositórios na forma de objetos de cooperação. Estes objetos são associados a um gerenciamento de versões, a um registro de acesso, a análises estatísticas, a uma lixeira, a um sistema de recomendação, a um mecanismo de busca ou a um mecanismo de ranqueamento. As operações sobre os objetos são registradas na forma de um log, possibilitando a realização de auditoria e a volta a estados anteriores.

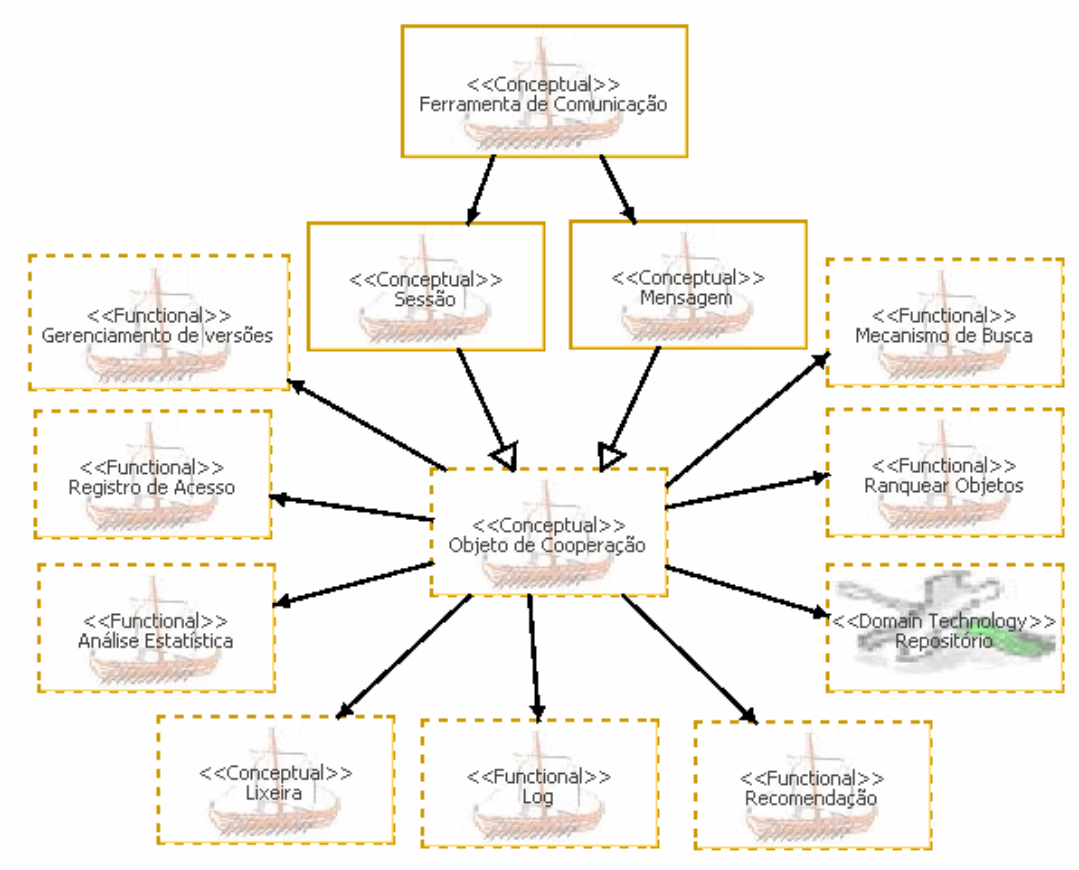

Figura 4.7. Modelo de características da cooperação nas ferramentas de comunicação

Os modelos de características são utilizados ao projetar uma nova aplicação para o domínio. As características são mapeadas em componentes que implementam as funcionalidades definidas. Os componentes são selecionados, implantados e configurados para oferecer suporte computacional às necessidades estabelecidas. Os componentes interoperáveis são organizados em *component kits*.

Conforme discutido no Capítulo 2, um *component kit* é uma coleção de componentes que foram projetados para trabalhar em conjunto [D'Souza & Wills, 1998]. De um kit de componentes gera-se uma família de aplicações, fazendo diferentes arranjos e eventualmente desenvolvendo alguns sob demanda. Nesta tese, para instrumentar o desenvolvedor de groupware, é oferecido o *Collaboration Component Kit*, com os componentes de colaboração utilizados para compor os serviços, implementando os aspectos da colaboração. Os componentes de colaboração foram obtidos a partir da análise do domínio, representada pelos modelos de características.

Conforme discutido no Apêndice A, um *component kit* não necessita ser exaustivo. Os *component kits* são extensíveis para acomodar novos componentes

necessários. No capítulo seguinte, algumas instanciações são analisadas. Componentes de software que sejam realmente reusáveis são refinados iterativamente até atingir a maturidade, a confiabilidade e a adaptabilidade desejadas [Gimenes & Huzita, 2005]. Os componentes de comunicação do *Collaboration Component Kit* são apresentados na Tabela 4.2.

Convencionou-se nomear os elementos arquiteturais na língua inglesa, pois no re-desenvolvimento do ambiente AulaNet, que serve de motivação para esta tese, o código está seguindo a padronização adotada pelos sistemas de código aberto, que são escritos em inglês, de modo a prepará-lo para futuras colaborações internacionais. O Apêndice B apresenta a descrição completa de todos os componentes propostos no kit. Para cada componente são descritos: nome, intenção, aplicabilidade, variabilidade, estrutura, usos conhecidos e componentes relacionados. Os componentes foram nomeados seguindo a recomendação de Cheesman & Daniels [2000] de acrescentar o sufixo "Mgr", que representa a abreviação de "Manager".

Componente	**Descrição**
MessageMgr	Oferece suporte à construção de mensagens. Este componente interage com o canal de comunicação para a transmissão dos dados. Para compor a mensagem podem ser utilizadas várias mídias (pelo menos uma), associando o MessageMgr aos componentes TextualMediaMgr, VideoMediaMgr, AudioMediaMgr ou PictorialMediaMgr. O componente também possibilita o anexo de arquivos na mensagem e seu envio por correio eletrônico.
TextualMediaMgr	Utilizado para mensagens com corpo textual. É possível configurar o tamanho do texto e o vocabulário disponível, através de filtros.
VideoMediaMgr	Utilizado para o tratamento de vídeo. Podem ser configuradas as taxas de captura e de envio de dados.
AudioMediaMgr	Utilizado para o tratamento de áudio. Podem ser configurados a taxa de captura e de envio de dados e o volume.
PictorialMediaMgr	Utilizado para incorporar imagens como parte da comunicação.
DiscreteChannelMgr	Utilizado para transmitir mensagens através de blocos de informação. Pode ser configurado para modo síncrono ou assíncrono, que influencia a maneira como as mensagens são tratadas; push ou pull, que define como a mensagem é entregue ao participante; e o atraso na entrega, que é utilizado para regular a interação em uma ferramenta síncrona [Pimentel et al., 2005].
ContinuousChannelMgr	Utilizado para transmissão contínua de informações.
MetaInformationMgr	Gerencia as meta-informações associadas às mensagens.
CategorizationMgr	Gerencia a categorização de mensagens.
DialogStructureMgr	Gerencia as inter-relações entre as mensagens, que podem ser na forma de lista, árvore ou grafo.
ConversationPathsMgr	Possibilita a utilização de uma máquina de estados para definir os caminhos disponíveis na conversação.
CommitmentMgr	Possibilita o gerenciamento de compromissos negociados durante a comunicação a partir de clichês de conversação.

Tabela 4.2. Componentes de comunicação do Collaboration Component Kit

Cada componente apresenta interfaces fornecidas e requeridas. A Figura 4.8 ilustra o componente de categorização de mensagens (CategorizationMgr). Este componente apresenta a interface ICategorizationMgr, que oferece os serviços relativos à utilização da categorização de mensagens, como atribuir categoria, modificar categoria, listar objetos de uma determinada categoria, etc.; ICategorizationMgrConfig, que oferece os serviços de configuração do componente, como manutenção do conjunto de categorias, estabelecimento da categoria default, etc.; Category, que representa a categoria em si; e ICategorizableObj, que é implementada no serviço colaborativo pelo respectivo objeto a ser categorizado.

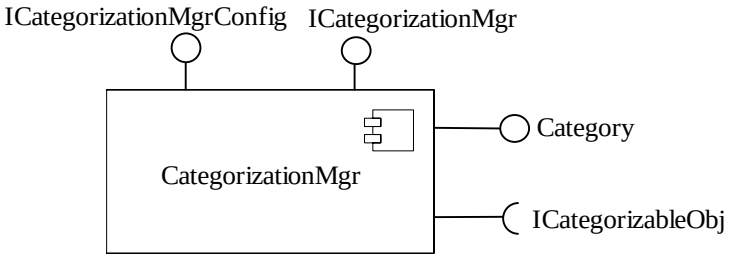

Figura 4.8. Componente de categorização de mensagens

A Tabela 4.3 apresenta os componentes de coordenação do Collaboration Component Kit. Estes componentes objetivam oferecer suporte computacional à organização do grupo, lidando com o acesso dos participantes, com as tarefas atribuídas e com os recursos alocados. O componente AssessmentMgr lida com a avaliação de mensagens; o RoleMgr, com a gestão dos papéis dos participantes; o PermissionMgr, com as permissões; o ParticipantMgr, com os participantes do grupo; o GroupMgr, com a criação de grupos e subgrupos; o SessionMgr, com a gerência das sessões; o FloorControlMgr, com as técnicas de conversação e políticas de acesso; o TaskMgr, com as tarefas e atividades; e o CompetencyMgr, com a gestão das competências dos participantes do grupo. Também são tratadas, através dos componentes AwarenessMgr e AvailabilityMgr, as informações de percepção voltadas para o feedback e feedthrough. O componente NotificationMgr é responsável pelo envio das notificações aos participantes.

Componente	Descrição
AssessmentMgr	Oferece suporte à avaliação qualitativa, possibilitando a associação de notas e conceitos aos objetos compartilhados.
RoleMgr	Cuida do gerenciamento dos papéis atribuídos aos participantes.
PermissionMgr	Gerencia as permissões dos participantes, que podem ser relativas ao papel ou ao indivíduo.
ParticipantMgr	Gerencia os participantes dos grupos.
GroupMgr	Possibilita formar grupos e subgrupos com os participantes.
SessionMgr	Gerencia a sessão de colaboração, incluindo mecanismos para convite e controle de acesso.
FloorControlMgr	Gerencia a ordem de participação. Podem ser alocadas várias políticas de acesso.
TaskMgr	Possibilita o gerenciamento das tarefas do grupo.
AwarenessMgr	Gerencia as informações de percepção em geral, registrando e operando sobre os eventos ocorridos no serviço.
CompetencyMgr	Oferece funcionalidade relativa à gestão das competências dos participantes.
AvailabilityMgr	Possibilita que o participante configure sua disponibilidade.
NotificationMg	Gerencia o envio das notificações aos participantes.

Tabela 4.3. Componentes de coordenação do Collaboration Component Kit

A Tabela 4.4 apresenta os componentes de cooperação do Collaboration Component Kit. Os componentes de cooperação lidam com o registro, a recuperação e o compartilhamento dos objetos. O CooperationObjMgr provê os mecanismos de registro e de concorrência de acesso aos objetos compartilhados. O SearchMgr provê mecanismos de busca. O VersionMgr provê controle de versões, possibilitando a recuperação de edições anteriores e a comparação dos objetos. O StatisticalAnalysisMgr provê funcionalidades para análise estatísticas dos objetos. O RankingMgr possibilita a atribuição de uma avaliação e uma classificação aos objetos, no intuito de ranqueá-los. O RecommendationMgr fornece um mecanismo de recomendação para objetos. O ActionLogMgr oferece recursos para registro das operações realizadas nos objetos. O AccessRegistrationMgr possibilita registrar os acessos que cada participante fez nos objetos, de modo a identificar os objetos já acessados e os novos. O TrashBinMgr oferece uma lixeira para manter temporariamente os objetos removidos, antes da eliminação definitiva.

Componente	Descrição
CooperationObjMgr	Provê mecanismos de compartilhamento de objetos.
SearchMgr	Provê mecanismos de busca para os objetos compartilhados.
VersionMgr	Controla versões dos objetos, possibilitando recuperar uma versão antiga, comparar alterações, entre outras funcionalidades.
StatisticalAnalysisMgr	Oferece funcionalidades de análise estatística dos objetos compartilhados.
RankingMgr	Gerencia a avaliação e a classificação dos objetos compartilhados.
RecommendationMgr	Recomenda objetos para os participantes ou para o grupo a partir das características dos objetos, dos participantes e das tarefas.
ActionLogMgr	Possibilita registrar o histórico de ações no serviço. Pode ser configurado o nível e o tipo de log a ser gerado, possibilitando fazer auditoria, acompanhar o uso, etc.
AccessRegistrationMgr	Registra os acessos dos participantes para cada objeto, possibilitando um controle do que já foi visitado.
TrashBinMgr	Implementa uma lixeira utilizada para armazenar objetos removidos.

Tabela 4.4. Componentes de cooperação do Collaboration Component Kit

Os componentes do Collaboration Component Kit são utilizados na composição dos serviços colaborativos. O conjunto de componentes é iterativamente refinado embasado na realimentação obtida pelo reuso no desenvolvimento das aplicações e pela experimentação das diversas configurações no suporte às características dos grupos e tarefas envolvidos.

Para implementação dos componentes não foram usados os mecanismos para geração de código disponibilizados pelo ambiente Odyssey [Oliveira et al., 2005], pois o modelo de componentes e a infra-estrutura de execução adotados nesta tese possuem especificidades distintas das disponíveis no ambiente. A arquitetura técnica é tratada em mais detalhes na dissertação de Barreto [2006].

4.3. A Arquitetura de Aplicação

Para oferecer suporte ao gerenciamento e à execução dos componentes, são utilizados *component frameworks* [Syzperski, 1997]. Na arquitetura proposta, utiliza-se um *component framework* para cada um dos tipos de componentes propostos, de modo a lidar com as peculiaridades de cada um. No Service Component Framework são acoplados os serviços, oferecendo suporte à montagem do groupware, e no Collaboration Component Framework são acoplados os componentes de colaboração, utilizados na montagem do serviço. A Figura 4.9 ilustra a implantação dos componentes nos *component frameworks*

correspondentes, que estabelecem as condições ambientais para as instâncias dos componentes e oferecem serviços relativos ao ciclo de vida.

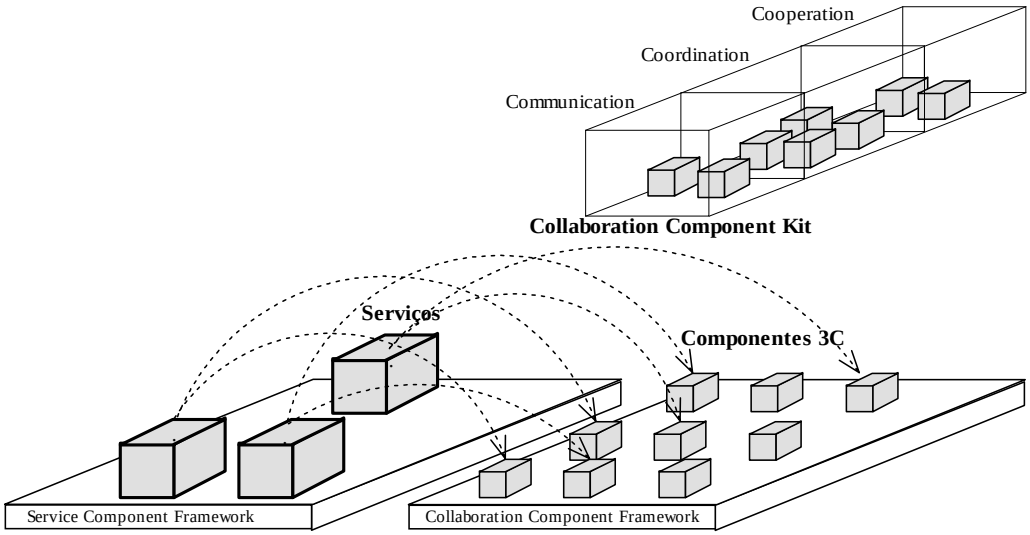

Figura 4.9. Component frameworks com serviços e componentes 3C

Os *component frameworks* são responsáveis por tratar a instalação, remoção, atualização, ativação, desativação, localização, configuração, monitoramento, importação e exportação de componentes. O Service Component Framework gerencia as instâncias dos serviços e a ligação com os componentes de colaboração correspondentes. O Collaboration Component Framework gerencia as instâncias dos componentes de colaboração, que são provenientes do Collaboration Component Kit.

Grande parte das funcionalidades dos *component frameworks* é recorrente e reusável. Um framework pode ser utilizado para a instanciação de uma família de sistemas [Pinto, 2000]. Nesta tese, é utilizado um framework para instanciar os *component frameworks*. Este tipo de framework é chamado de *component framework framework* (CFF) [Szyperski, 1997, p.277]. Um *component framework framework* é visto como um *component framework* de segunda ordem, onde seus componentes são *component frameworks* [Szyperski, 1997, p.276]. Da mesma forma que um componente interage com outros diretamente ou mediado pelo *component framework*, o mesmo pode ser dito dos *component frameworks*, cujo suporte de mais alto nível é o *component framework framework* [Szyperski, 1997, p.277]. A Figura 4.10, estendendo a notação utilizada por Szyperski [1997, p.278], ilustra a arquitetura de aplicação, incluindo o Groupware Component Framework Framework, como o *component framework* de segunda ordem.

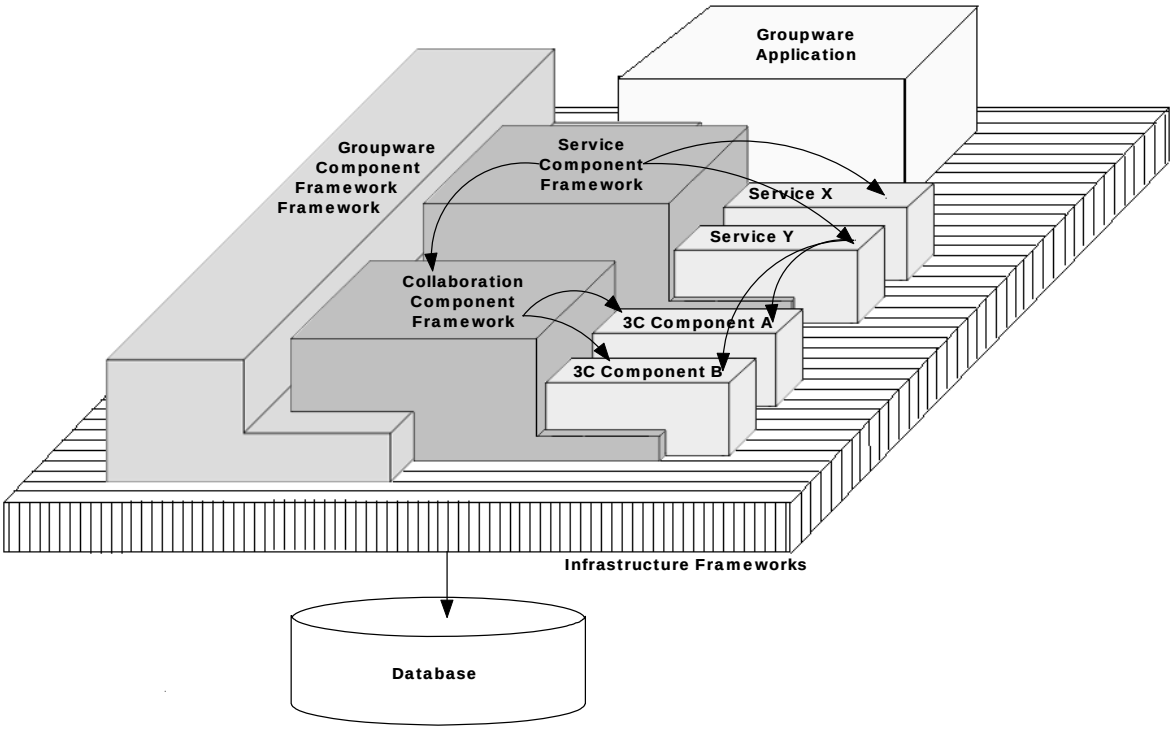

Figura 4.10. A arquitetura de aplicação proposta

A arquitetura adotada segue uma divisão em camadas, onde se distingue a camada de apresentação (não representada na Figura 4.10), que é responsável pela captura e apresentação de dados e pela interação com o usuário; a camada de negócio, que captura o modelo da lógica de negócio do domínio da aplicação; e a camada de infra-estrutura que implementa os serviços técnicos de baixo nível. A divisão em camadas é importante para tratar a complexidade de sistemas componentizados [Szyperski, 1997, p.277].

A mesma infra-estrutura desenvolvida para a camada de negócio pode ser utilizada para mais de uma apresentação, como por exemplo, PDA, desktop implementado em HTML e desktop implementado em Flash (*Rich Internet Application*). Quando os serviços da camada de negócio necessitam de acesso remoto, como por exemplo, em um cliente PDA, são disponibilizados web services que encapsulam a fachada da camada de negócio. Nos demais casos, a apresentação acessa diretamente a fachada do negócio. Esta solução segue os padrões de projeto *Façade* [Gamma et al., 1994], *Data Transfer Object* e *Business Delegate* [Alur et al., 2001].

O ferramental desenvolvido nesta tese instrumenta o desenvolvimento da camada de negócio, implementando os conceitos do modelo 3C de colaboração. A arquitetura de aplicação expressa a estrutura dos componentes do domínio, representando um projeto lógico de alto nível independente da tecnologia de suporte [D'Souza & Wills, 1998]. Já a camada de infra-estrutura é independente do domínio de aplicação e é tratada separadamente na dissertação de Celso Gomes Barreto [2006], que faz parte do consórcio de pesquisa onde esta tese está inserida.

Os *component frameworks* estendem o suporte oferecido pelos frameworks de infra-estrutura. Por exemplo, o container utilizado (JBoss) e os frameworks Spring e Hibernate oferecem suporte a transação declarativa, persistência, integração MVC, injeção de dependências, localização, etc. Na dissertação de Barreto [2006] a arquitetura técnica é tratada em mais detalhes.

O suporte computacional oferecido pelos componentes é reusável para oferecer suporte à colaboração nos diversos grupos da organização, como por exemplo, diretoria, gerência, administração e suporte. Desta forma, em um ambiente educacional, além do suporte à interação em sala de aula, podem ser disponibilizadas ferramentas para mediar a colaboração entre os professores de um curso, de um departamento e de uma instituição. Cada aplicação de groupware implementa sua organização específica.

4.4. O Modelo de Componentes

Um modelo de componentes define as características dos componentes e de seu ciclo de vida. O modelo de componentes utilizado nesta tese é uma extensão do modelo Java Beans e do modelo oferecido pelo framework Spring. A estes modelos são acrescentadas a definição do ciclo de vida dos componentes, a forma de empacotamento, a sintaxe do arquivo descritor e as regras de interação entre os componentes e os *component frameworks.*

O ciclo de vida de um serviço é ilustrado na Figura 4.11. O componente é posto no sistema de arquivos, em um diretório pré-determinado (d*eployment*). A atividade de *deployment* consiste em implantar um componente para ser instalado

pelo sistema [Szyperski, 2003]. São previstos diretórios específicos para cada um dos Cs do modelo de colaboração. No *deployment*, o componente pode ser customizado editando seu arquivo descritor.

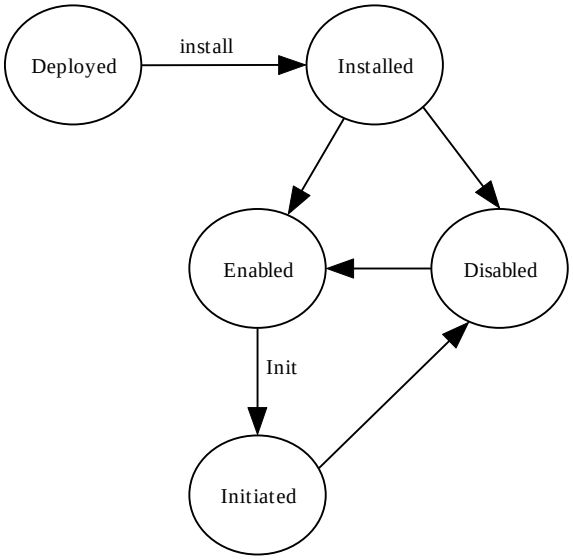

Figura 4.11. Ciclo de vida dos componentes

Para passar do estado de *deployed* para *installed* e se tornar pronto para execução, não há intervenção do desenvolvedor, visto que a atividade de *deployment* não faz parte do desenvolvimento [Szyperski, 2003]. Ao iniciar o servidor de aplicações, o *component framework* verifica a existência de novos serviços e, ao encontrar um novo serviço, executa sua instalação. A instalação do componente consiste em copiar os arquivos, registrar o componente e executar o script de instalação ou de atualização. Neste ponto, as dependências dos componentes e seus estados são lidos e verificados, levando-os para o estado habilitado (*enabled*) ou desabilitado (*disabled*). Os componentes são carregados, inicializados e se tornam prontos para uso.

Um mesmo serviço pode possuir várias instâncias, que são independentes entre si. Por exemplo, no caso do ambiente AulaNet, para cada curso que utiliza um serviço, é criada uma instância do componente[17]. O *Service Component Framework* gerencia as instâncias dos serviços e guarda o estado atual de cada uma, possibilitando a posterior restauração. Na criação de uma nova instância, são utilizados os valores padrões definidos no arquivo descritor.

[17] Alguns serviços possuem escopo de turma, sendo necessária uma instância para cada turma.

A instanciação de um novo serviço implica na instanciação dos componentes de colaboração utilizados na composição do serviço. O *Service Component Framework* interage com o *Collaboration Component Framework* para possibilitar a instanciação e a associação entre as instâncias dos componentes. Visando reduzir o acoplamento entre os dois *component frameworks*, é utilizada uma interface que provê um contrato de utilização entre eles.

A instalação e o gerenciamento dos componentes de colaboração seguem um procedimento similar ao descrito para os serviços. Os componentes de colaboração contam com arquivos descritores que definem configurações padrões, utilizadas na instanciação dos componentes. O *Collaboration Component Framework* gerencia a configuração dos componentes de colaboração.

O modelo de componentes define a maneira de tratar os componentes como unidades executáveis de produção, aquisição e instalação independente, definindo a forma de empacotamento do componente [Szyperski, 1997]. Os serviços são empacotados em um arquivo ZIP, com os diretórios CLASSES, DOC, LIBS, SCRIPTS e WEB. Os diretórios CLASSES, WEB e LIBS contêm os arquivos que implementam o componentes e são copiados para os diretórios correspondentes do servidor de aplicações. O padrão J2EE define os diretórios WEB-INF\classes e WEB-INF\libs para receber os arquivos class e jar, e o diretório raiz da aplicação para receber os arquivos que implementam a interface web (html, jsp, js, gif, etc.). O diretório DOCS do empacotamento do serviço disponibiliza a documentação do componente, que inclui a documentação voltada para utilização e para o desenvolvimento e manutenção. Por fim, o diretório SCRIPTS contém os scripts que são executados na instalação, atualização e remoção do serviço. Estes scripts atualizam a base de dados, criam diretórios e alocam os recursos necessários.

O modelo de componentes define também a sintaxe do arquivo descritor. O arquivo descritor é colocado na raiz da estrutura de diretórios do componente. O arquivo descritor é codificado na linguagem XML e segue a estrutura apresentada na Figura 4.12. Optou-se por utilizar a linguagem XML, pois este é o padrão adotado nos frameworks de infra-estrutura utilizados na arquitetura técnica [Barreto et al., 2005]. São definidos no arquivo descritor o identificador, o nome, a descrição, a versão, o tipo e as configurações do serviço e os componentes de

colaboração utilizados, com suas respectivas customizações. No exemplo da figura, o serviço Conferências está utilizando o componente CategorizationMgr.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <service>
    <!--                        -->
  - <service-handler>
      <service-name>Conferência</service-name>
      <service-description>Fórum textual de discussão.</service-description>
      <service-type>Asynchronous</service-type>
      <!--                        -->
      <service-participation-URL>participation/InitialPage.jsp</service-participation-URL>
      <service-configuration-Initial-URL>config/ConfigPage.jsp</service-configuration-Initial-URL>
    </service-handler>
    <!--                                  -->
  - <specific-configuration>
      <allowDesactivation>true</allowDesactivation>
    </specific-configuration>
    <!--                                          -->
  - <collaboration-components>
    - <collabComponent name="categorizationManager">
        <allowNonCategorizedMessage>false</allowNonCategorizedMessage>
      - <defaultCategory>
          <name>Questão</name>
          <descricao>Use a categoria para propor questões</descricao>
        </defaultCategory>
      - <defaultCategory>
          <name>Argumentação</name>
          <descricao>Argumentar um ponto de vista.</descricao>
        </defaultCategory>
      </collabComponent>
    - <collabComponent name="taskManager">
      - <tasks>
        - <task name="Send message" type="collaborative" permissionClass="All">
            <taskComplement>CategorizationManager</taskComplement>
            <taskComplement>ConversationSessionManager</taskComplement>
          </task>
          <task name="Remove message" permissionClass="Mediator" />
          <task name="Alter category" permissionClass="Mediator" />
          <task name="Create conference" permissionClass="Coordinator" />
        </tasks>
      </collabComponent>
    </collaboration-components>
</service>
```

Figura 4.12. Arquivo descritor de um serviço

Os conectores definidos no modelo de componentes são específicos do domínio e são dependentes do modelo de negócio com o qual os componentes trabalham [Wills, 2001, p.310]. Cada componente disponibiliza um modelo de objetos com os quais os clientes do componente irão lidar. Eventualmente um componente possui uma implementação diferente para um determinado objeto. Nestes casos, adota-se um adaptador que faz a conversão entre eles [Wills, 2001, p.313].

4.5. Instanciação de Groupware

Instrumentado pelo modelo 3C de colaboração, o desenvolvedor modela a aplicação e seus requisitos, e seleciona os serviços e seus componentes de modo a oferecer suporte às necessidades de colaboração. O desenvolvedor seleciona os componentes desejados a partir dos *component kits*, implantando-os nos

component frameworks correspondentes. Se o sistema já estiver em operação, o desenvolvedor identifica os problemas a serem resolvidos a partir da análise da colaboração. As soluções são mapeadas em funcionalidades, que são implementadas pelos componentes de colaboração [Pimentel et al., 2005].

O groupware é montado a partir do levantamento de requisitos, que identifica as necessidades de colaboração e as características do grupo e das tarefas. O trabalho em grupo é modelado e a dinâmica da colaboração é estabelecida. Com base nestas informações, são selecionados os serviços que irão oferecer o suporte computacional à colaboração, definindo a arquitetura da solução. Após a montagem do ambiente, são efetuados os testes e as avaliações, de modo a refinar o ambiente iterativamente. Caso seja necessário modificar a implementação de um serviço existente ou desenvolver um novo, adota-se um ciclo similar ao anterior. Inicia-se com a modelagem das necessidades de colaboração e das características do grupo e das tarefas, no escopo do serviço. A partir destas análises, embasadas pelo modelo 3C, são selecionadas as características relevantes para a solução. Estas características são mapeadas em componentes de colaboração que implementam as funcionalidades requeridas, e o serviço é montado a partir deles.

Caso seja identificada uma necessidade não implementada pelos componentes pré-existentes há três opções: ela pode ser implementada no código da aplicação, de uma forma específica ao domínio em questão; no código do serviço; ou como um novo componente 3C, de modo a disponibilizar a nova funcionalidade para futuro reuso. O novo componente de colaboração é desenvolvido em conformidade com o *Collaboration Component Framework* e com o modelo de componentes estabelecido.

4.6. Uso de Frameworks de Domínio

Com a evolução do suporte à colaboração, podem ser utilizados frameworks para instanciar famílias de componentes. As funcionalidades de diversos componentes similares são generalizadas e um framework especializável e customizável é implementado. O framework de classes propicia o reuso do código

e fornece ao desenvolvedor mecanismos de especialização, que são utilizados para instanciar famílias de componentes, serviços e aplicações.

No exemplo da Figura 4.13, está representado um framework para geração de diferentes chats. O framework de domínio fundamenta o conhecimento sobre a família de serviços e propicia o reuso do código recorrente. As ligações internas dos componentes são encapsuladas e são oferecidos *hot spots*[18] que ficam disponíveis para o desenvolvedor implementar as especificidades do serviço.

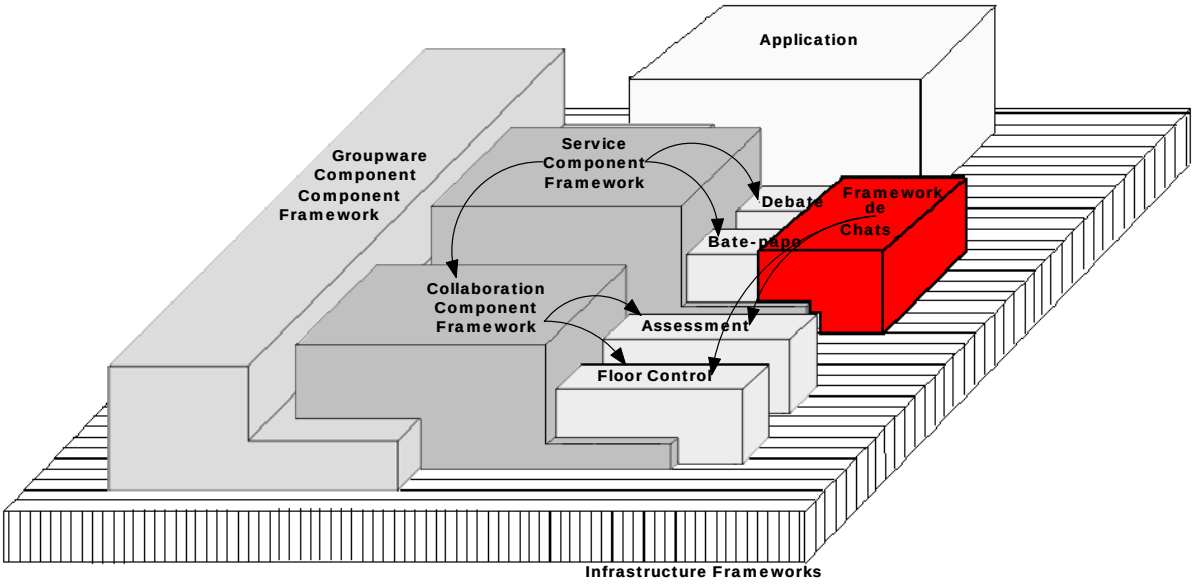

Figura 4.13. Framework de domínio para instanciação de diferentes chats

Podem ser disponibilizados também frameworks para consolidar e gerar uma família de componentes de colaboração, tratando das diversas variações de um mesmo elemento. Por exemplo, ao invés de oferecer ao desenvolvedor diversos componentes de avaliação, pode ser oferecido um framework para instanciar componentes de avaliação, reusando o código recorrente, conforme ilustrado na Figura 4.14.

[18] Para mais detalhes sobre a terminologia associada a frameworks, consulte o Apêndice A.

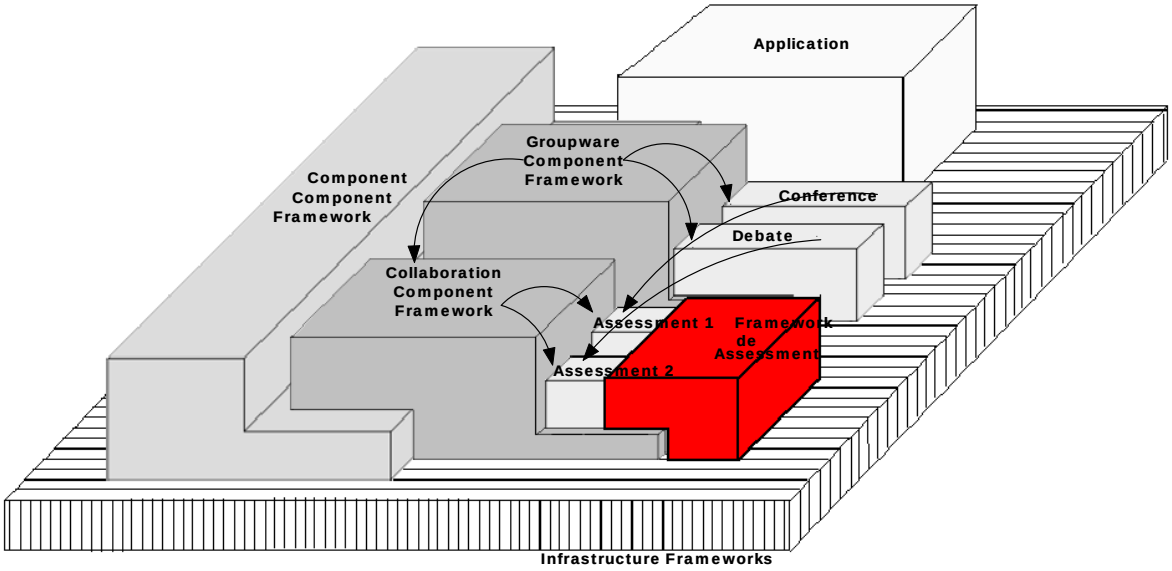

Figura 4.14. Framework de domínio para instanciação de componentes de colaboração voltados para avaliação da participação

A granularidade dos frameworks de domínio é um fator importante a ser considerado. Quanto mais abstrato e abrangente for o framework mais difícil se torna seu uso. Quando mais pontos de extensão, maior a aplicabilidade dentro de um domínio de aplicação, porém sua utilização requer uma longa curva de aprendizado e sua instanciação requer um trabalho maior de customização [Oliveira, 2001]. A granularidade dos componentes e frameworks é iterativamente refinada ao longo do projeto [Gimenes & Huzita, 2005].

Um groupware trata de uma área de aplicação específica, como por exemplo, aprendizagem colaborativa, comércio eletrônico, acompanhamento de projetos, desenvolvimento de software, etc. Frameworks também podem ser utilizados para gerar uma família de groupware de uma mesma área de aplicação, de modo a reusar funcionalidades recorrentes, conforme ilustrado na Figura 4.15. Pode ser desenvolvido, por exemplo, um framework para instanciar ambientes de aprendizagem, a partir do qual é instanciado o AulaNet, entre outros.

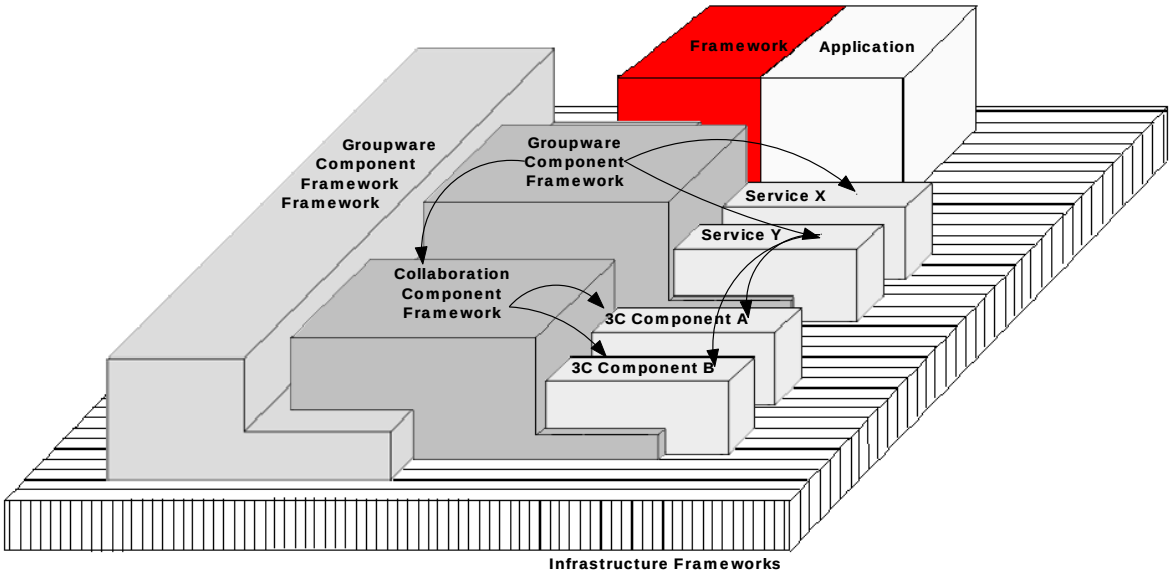

Figura 4.15. Framework de domínio para instanciar uma determinada família de aplicações

Frameworks e componentes são tecnologias complementares [Gimenes & Huzita, 2005; Szyperski, 1997, p.278]. Na abordagem proposta, os frameworks podem ser usados para instanciar os componentes e os ambientes. A abordagem de componentes oferece maior flexibilidade, visto que um framework só oferece variabilidade nos pontos previstos para tal; e maior extensibilidade, visto que pode-se desenvolver e acrescentar novos componentes ou substituir algum em uso, sem impactar a família de aplicações. Entretanto, os frameworks consolidam o conhecimento sobre um subdomínio e possibilitam um maior reuso de código para componentes e ambientes similares.

4.7. Considerações Finais

Groupware é difícil de desenvolver e de manter [Grudin, 1989]. Desenvolver groupware requer diversas áreas de conhecimento, como informática, psicologia, pedagogia, sociologia e cognição. Envolve também várias complexidades técnicas inerentes a sistemas distribuídos e multi-usuário. Um conjunto de componentes interoperáveis fundamentados no modelo 3C de colaboração encapsula parte das dificuldades técnicas e dos aspectos multidisciplinares na implementação dos componentes. Com isto, o desenvolvedor foca sua atenção na investigação dos aspectos específicos de

interação e colaboração, projetando um groupware mais adequado às necessidades do grupo e menos susceptível às falhas de usabilidade.

No desenvolvimento de groupware, os requisitos raramente são claros o suficiente para possibilitar uma especificação precisa antecipada do comportamento do sistema. É difícil prever como um determinado grupo colabora e cada grupo tem características e objetivos consideravelmente distintos [Gutwin & Greenberg, 2000]. Por envolver um grupo, multiplicam-se as possibilidades de interações e aumenta a demanda por sincronismo e concorrência de acesso, o que dificulta a construção de mecanismos de interação adequados e a condução de testes. Usar um conjunto de componentes que são reusados em diversas situações aumenta a confiabilidade e a estabilidade do sistema, além de possibilitar substituir ou reimplementar componentes com impactos limitados no restante do sistema. O desenvolvedor também pode experimentar e prototipar diversas configurações iterativamente para refinar os requisitos do sistema e o suporte à colaboração.

O uso de componentes oferece capacidade de customização e extensão. Oferecer um conjunto de componentes torna desnecessário prever todas as possibilidades de utilização e dar suporte a elas. São oferecidos blocos de granularidade média que o desenvolvedor usa para compor a aplicação [Szyperski, 1997]. Estes blocos também oferecem suporte à pluralidade de plataformas a partir das quais são acessados os sistemas colaborativos. Isto inclui computadores desktop e dispositivos móveis, como celulares e PDAs, que estão cada vez mais com recursos suficientes para incrementar a comunicação, a coordenação e a cooperação do grupo [Filippo et al., 2005].

Software é evolutivo, e groupware é evolutivo por natureza [Tam & Greenberg, 2004]. A composição e as características dos grupos de trabalho se alteram ao longo do tempo, assim como as tarefas executadas. O grupo aprende, surgem afinidades e conflitos entre os membros, entram e saem pessoas, levando o grupo a mudar continuamente. A componentização provê a capacidade de montar e evoluir um ambiente de trabalho específico, selecionando e configurando um conjunto de ferramentas colaborativas específicas para suas necessidades.

Esta pesquisa propõe a estruturação de um sistema colaborativo em componentes que encapsulam as dificuldades técnicas de sistemas distribuídos e

multi-usuário e refletem os conceitos da colaboração, modelados pelo modelo 3C. Este ferramental instrumenta o desenvolvimento da camada de negócio da aplicação, possibilitando trocar a camada de interface de um serviço, mantendo a mesma lógica de aplicação. No contexto deste trabalho, a engenharia de domínio é baseada no modelo 3C de colaboração, de modo a guiar o desenvolvimento de aplicações colaborativas. A transição entre as atividades do desenvolvimento e o mapeamento dos conceitos de análise para as estruturas do código é estreitado, facilitando o desenvolvimento iterativo e a posterior manutenção da aplicação.

"Sem uma arquitetura adequada, a construção de groupware e sistemas interativos em geral é difícil de obter, o software resultante é difícil de manter e o refinamento iterativo é dificultado" [Calvary et al., 1997]. Uma arquitetura componentizada possibilita que componentes sejam selecionados para montar um groupware voltado aos interesses específicos de um grupo. Os componentes são customizados e combinados na medida da necessidade, tendo em mente futuras manutenções. O uso desta abordagem propicia a prototipação e a experimentação, que são fundamentais em CSCW, visto que ainda são escassos e pouco documentados os casos de sucesso. A prototipação deve ser rápida, pois ocorre enquanto o grupo trabalha, para se obter um feedback oportuno e proceder com os ajustes. O uso de componentes auxilia a adaptação dinâmica do ambiente e do suporte à colaboração através da recomposição e reconfiguração do sistema.

Entretanto, vale ressaltar que a solução proposta não elimina a necessidade de um desenvolvedor consciente e conhecedor do assunto em questão, pois não basta sair ligando os componentes aleatoriamente para produzir um sistema colaborativo eficaz. O uso de um processo sistematizado ajuda a reduzir estas dificuldades, ao instrumentar o desenvolvedor com catálogos de problemas, soluções, elementos e componentes, bem como as atividades a serem executadas e os artefatos a serem produzidos para obter o produto final.

Um processo é um conjunto de políticas, tecnologias, procedimentos, artefatos e estruturas organizacionais utilizado para conceber, desenvolver, implantar e manter um produto de software [Fuggetta, 2000]. No consórcio de pesquisa no qual esta tese está inserida, Mariano Pimentel [2006] propõe um processo de desenvolvimento de groupware que leva em consideração a arquitetura e a abordagem propostas nesta tese. Este processo instrumenta e guia o

desenvolvedor, que passa a contar com uma abordagem sistematizada para o desenvolvimento incremental de groupware baseado no modelo 3C [Pimentel et al., 2005]. O processo estabelece a ligação entre os problemas e soluções, bem como o mapeamento para as funcionalidades e componentes correspondentes. O processo define as atividades e artefatos que são gerados durante o desenvolvimento e instrumenta o desenvolvedor para que a partir dos sintomas possa identificar quais elementos precisam ser refinados. O processo estabelece técnicas e abordagens para avaliar uma ferramenta, com o objetivo de identificar problemas e proceder em novos ciclos de desenvolvimento.