Capítulo

3

Engenharia de Groupware: Desenvolvimento de Aplicações Colaborativas

Hugo Fuks[1], Alberto B. Raposo[2], Marco A. Gerosa[1]

[1]LES—Laboratório de Engenharia de Software
[2]TECGRAF—Laboratório de Tecnologia em Computação Gráfica
Departamento de Informática, PUC-Rio
R. Marquês de São Vicente, 225
Gávea, Rio de Janeiro – RJ, 22453-900

{hugo,gerosa}@inf.puc-rio.br, abraposo@tecgraf.puc-rio.br

Resumo

A tecnologia gera ambientes que dão suporte às diferentes formas de relacionamento humano e, por conseguinte, revoluciona o modo de se trabalhar na sociedade conectada. A criação de espaços de compartilhamento e troca de informação propicia o trabalho colaborativo distribuído e descentralizado. A Engenharia de Software, que muito avançou no desenvolvimento de aplicações mono-usuário, foi buscar na área de ***CSCW*** *— trabalho colaborativo apoiado por computadores — os conceitos necessários para esta nova realidade.*

Este curso é uma introdução à tecnologia de groupware, software para apoiar o trabalho colaborativo. O assunto é apresentado do ponto de vista conceitual, propondo um modelo de colaboração baseado nos aspectos de Comunicação, Coordenação e Cooperação. Também é formulada uma proposta de Engenharia de Groupware, com a qual espera-se identificar os elementos necessários à criação de aplicações colaborativas, trabalhando com requisitos de groupware e técnicas de UML (Unified Modeling Language) estendida para esta finalidade. As abordagens conceitual e ferramental são discutidas usando exemplos de aplicações de groupware.

Abstract

The connected society works in different ways, partly because digital technology shapes new environments that, on their turn, shape new forms of interaction between humans. Distributed and decentralized collaborative work is favored by the creation of shared information spaces on the Internet. Although Software Engineering achieved good results on single user applications, for developing multiple user applications, new insights and concepts coming from CSCW - Computer Supported Cooperative Work - are necessary.

This course is an introduction to groupware technology, which is designed to support collaborative work. Starting from a collaboration model based upon the notions of Communication, Coordination and Cooperation, groupware is presented from a conceptual point of view. Then, a programming approach to Groupware Engineering is proposed to help identify the essential elements for developing collaborative applications, using requirements and extended UML (Unified Modeling Language) techniques specifically tailored for this purpose. A few groupware applications are presented in order to discuss both approaches.

3.1. Introdução

É apropriado começarmos este texto com uma definição para Engenharia de Groupware. Porém, sendo groupware um tipo de software, a definição de Engenharia de Software dada pelo IEEE Standard Glossary of Software Engineering Terminology [IEEE, 1991] é o nosso ponto de partida:

> "**software engineering.** (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1)."

Conforme Dan Berry [Berry, 1992], a definição inclui a operação e a manutenção do software, não se limitando ao seu desenvolvimento; diz que a abordagem deve ser sistemática, disciplinada e quantificável e não fala nada de ciência da computação e matemática. Além disto, a definição inclui o estudo e a busca por abordagens que possibilitem a realização das atividades da engenharia de software, isto é, o estudo e pesquisa de métodos, técnicas, ferramentas, etc.

O próprio Berry reformula esta definição e propõe a seguinte:

1. Engenharia de Software é aquela engenharia que aplica:

- uma abordagem sistemática, disciplinada e quantificável,

- os princípios da ciência da computação, design, engenharia, administração, matemática, psicologia, sociologia e outras disciplinas se necessário for,

- e às vezes pura invenção,

para criar, desenvolver, operar e manter de forma econômica, confiável e correta, soluções de alta qualidade para problemas que envolvam software.

2. Engenharia de Software também é o estudo e a busca por abordagens para a realização das atividades (1) acima.

Examinando a nova definição, vemos que ela inclui conceitos de outras disciplinas como administração, psicologia, sociologia, entre outras. Isto é conveniente, pois a base de conhecimento necessária para se projetar um bom groupware é o objeto da pesquisa da área de CSCW (*Computer Supported Cooperative Work*, que pode ser traduzido por trabalho colaborativo apoiado por computadores). Segundo [Greif, 1988], CSCW estuda as funções e as relações de trabalho entre grupos de pessoas e sistemas de computação. Groupware pode ser entendido como a tecnologia baseada em mídia digital que dá suporte às atividades de pessoas organizadas em grupos que podem variar em tamanho, composição e local de trabalho.

Um pouco mais à vontade com o título, é hora de examinarmos o sub-título deste trabalho, Desenvolvimento de Aplicações Colaborativas, que será abordado com base na experiência dos autores no desenvolvimento do ambiente AulaNet. O AulaNet é um *learningware*, groupware dedicado à aprendizagem colaborativa, desenvolvido desde 1997 no Laboratório de Engenharia de Software (LES) do Departamento de Informática da PUC-Rio[1].

3.1.1. Ciclo de desenvolvimento de groupware

Nesta seção, vamos contextualizar o ciclo de desenvolvimento de software para groupware. Tendo em vista que este documento não se propõe a ser um livro-texto sobre o tema, somente algumas das fases serão tratadas. De acordo com esta mesma filosofia, somente certos aspectos das fases escolhidas serão discutidos. As fases de desenvolvimento de software [Pressman, 1992] são apresentadas na Figura 3.1, juntamente com os aspectos abordados neste documento.

[1] O AulaNet vem na forma de um servidor gratuito que atualmente está na sua versão 2.0. Mais informações sobre o Ambiente AulaNet, como publicações e uso, podem ser consultadas a partir de http://www.les.inf.puc-rio.br/aulanet.

Todas as URLs que aparecerem no texto foram visitadas em maio de 2002.

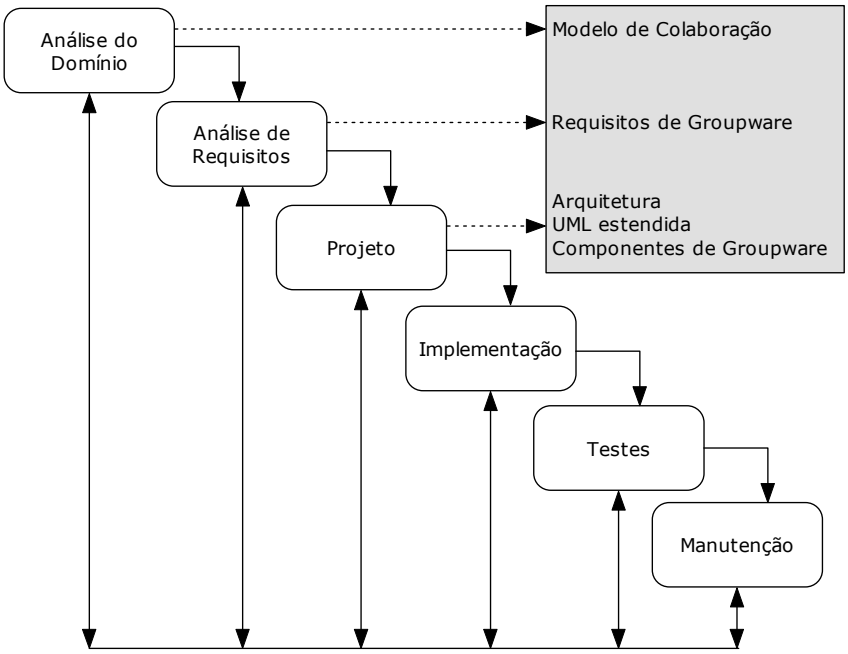

Figura 3.1. Ciclo de desenvolvimento de software indicando as fases e aspectos abordados neste documento

Para embasar a fase de análise do domínio, onde se estuda o domínio da aplicação a ser desenvolvida, é proposto o modelo de colaboração baseado nos aspectos de comunicação, coordenação e cooperação (Seção 3.2). Para a fase de análise de requisitos, onde as atenções são concentradas no software, são listados requisitos de groupware (Seção 3.3). Para instrumentar a fase de projeto, onde o software é concebido de forma a atender os requisitos, são apresentados o conceito de componente de groupware, arquiteturas de componentes e extensões da linguagem UML (Seção 3.5).

3.2. Colaboração

Trabalhando colaborativamente, pelo menos potencialmente, pode-se produzir melhores resultados do que se os membros do grupo atuassem individualmente. Em um grupo pode ocorrer a complementação de capacidades, de conhecimentos e de esforços individuais, e a interação entre pessoas com entendimentos, pontos de vista e habilidades complementares [Fuks et al., 2002]. Colaborando, os membros do grupo têm retorno para identificar precocemente inconsistências e falhas em seu raciocínio e, juntos, podem buscar idéias, informações e referências para auxiliar na resolução dos problemas. O grupo também tem mais capacidade de gerar criativamente alternativas, levantar as vantagens e desvantagens de cada uma, selecionar as viáveis e tomar decisões [Turoff and Hiltz, 1982].

Trabalhar em grupo também traz motivação para o membro, pois seu trabalho vai estar sendo observado, comentado e avaliado por pessoas de uma comunidade da qual ele faz parte [Benbunan-Fich and Hiltz, 1999]. Ao argumentar suas idéias com os

outros membros, o participante trabalha ativamente seus conceitos, raciocinando sobre os mesmos e refinando-os.

Apesar de suas vantagens, trabalhar colaborativamente demanda um esforço adicional para a coordenação de seus membros. Sem coordenação, boa parte dos esforços de comunicação não será aproveitada na cooperação, isto é, para que o grupo possa operar em conjunto de forma satisfatória, é necessário que os compromissos assumidos nas conversações entre os participantes sejam realizados durante a cooperação. A coordenação deve evitar conflitos inter-pessoais que possam prejudicar o grupo.

Para possibilitar a colaboração, são necessárias informações sobre o que está acontecendo. Estas informações são fornecidas através de elementos de percepção que capturam e condensam as informações coletadas durante a interação entre os participantes. A percepção em si é relativa ao ser humano, enquanto os elementos de percepção estão relacionados à interface do ambiente.

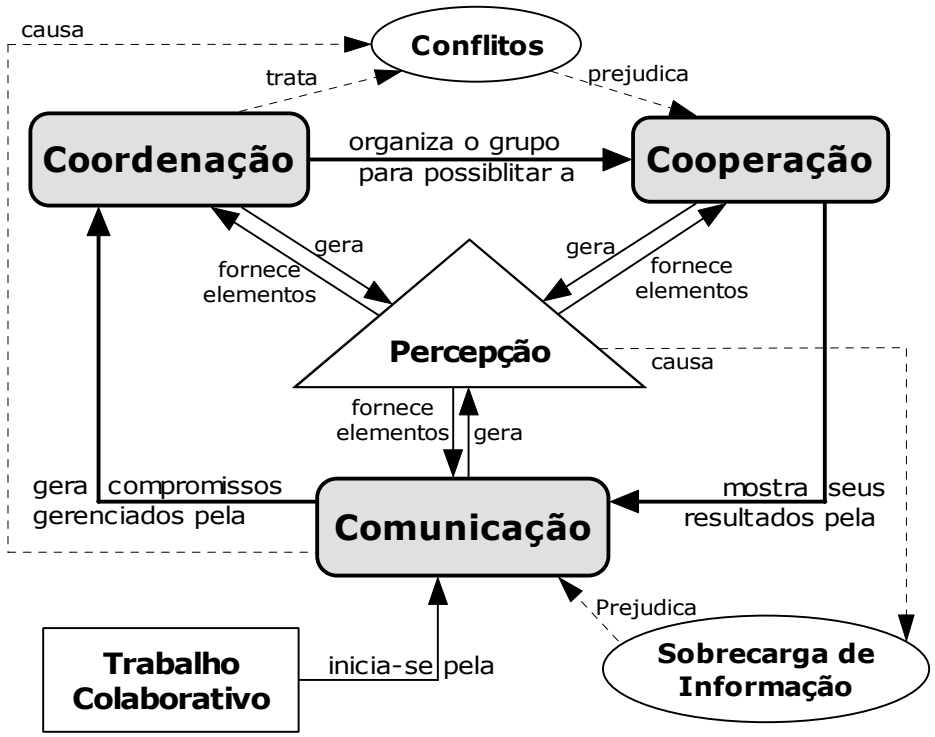

Figura 3.2. Modelando a Colaboração

O diagrama da Figura 3.2 sumariza os principais conceitos abordados. Este diagrama é um refinamento do modelo apresentado em [Fuks and Assis, 2001], que é baseado em [Ellis et al., 1991]. Veremos agora em mais detalhes os principais elementos do diagrama e suas inter-relações. Vale lembrar que apesar de estarmos separando estes conceitos para efeito de análise, não é possível considerá-los monoliticamente, uma vez que são intimamente dependentes e inter-relacionados.

3.2.1. Comunicação

Durante a comunicação, as pessoas almejam construir um entendimento comum e compartilhar idéias, discutir, negociar e tomar decisões. Os participantes de uma equipe de trabalho devem se comunicar para conseguir realizar tarefas interdependentes, não completamente descritas ou que necessitem de negociação [Fussel et al, 1998].

Delvin e Rosenberg ressaltam a importância do conhecimento individual e das práticas cooperativas, como a linguagem das mãos na comunicação face-a-face, que as pessoas desenvolvem de forma a coordenar a variedade de conhecimentos individuais e atingir o entendimento mútuo. O contexto cultural, o domínio em questão e os conhecimentos individuais influenciam como as expressões de linguagem são produzidas pelo comunicador e interpretadas pelo receptor [Delvin and Rosenberg, 1996].

Como o ambiente define o espaço compartilhado de informação entre os indivíduos, ele pode fornecer elementos adicionais não-verbais à estrutura de linguagem utilizada na conversação. Isto simplifica a comunicação verbal, que é complementada pelos elementos presentes no ambiente [Gutwin and Greenberg, 1999].

As informações são transmitidas através de um canal de percepção criado no espaço compartilhado onde ocorre a conversação. Este canal de percepção fica implícito no canal de comunicação. Por exemplo, em uma conversa face-a-face, as informações são transmitidas através do som, dos gestos e das expressões dos indivíduos, entre outros.

Quando se comunicam, as pessoas geralmente não estão cientes das expressões, da conversação em sua totalidade ou dos elementos de percepção e de expressão utilizados, porque sua atenção está voltada para o propósito e para os efeitos das mensagens. Entretanto, quando há algum tipo de confusão ou problema, as estruturas de linguagem e os elementos de percepção utilizados são trazidos para o foco central, em uma tentativa de reparar o desentendimento.

Para haver entendimento e a comunicação cumprir seu objetivo, é necessário o conhecimento de todos sobre a utilização das mídias de transmissão e de recebimento dos dados, bem como a participação ativa do receptor, que deve estar atento às informações transmitidas e aos elementos utilizados, para que seja viabilizado o canal de percepção. Todos os envolvidos na comunicação devem estar cientes das estruturas de linguagem e expressões utilizadas.

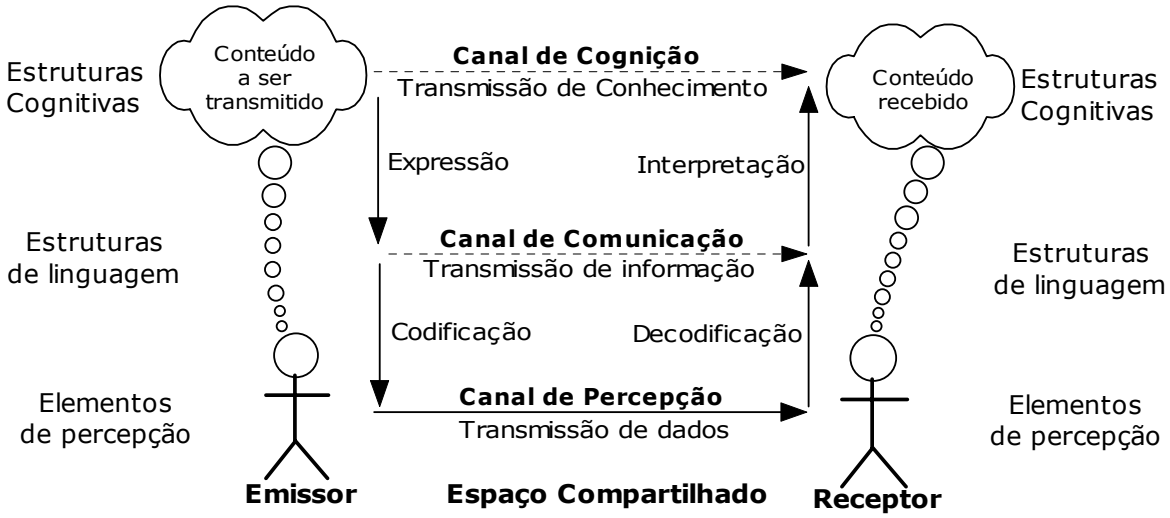

Figura 3.3. Modelando a Comunicação

Na Figura 3.3, encontra-se o diagrama que retrata algumas destas colocações. Para o emissor transmitir o conteúdo desejado, que é formado nas suas estruturas cognitivas a partir de um processo de raciocínio com base em seus conhecimentos, ele necessita expressar-se através da estrutura de linguagem utilizada naquela conversação. Montadas as expressões de linguagem, o emissor as codifica nos recursos oferecidos pelo ambiente (espaço compartilhado), que as transmite para o receptor. Este as recebe através da decodificação dos dados obtidos dos elementos de percepção disponíveis no espaço compartilhado, e as interpreta de forma a agregar novos conhecimentos em suas estruturas cognitivas. A comunicação é bem sucedida se houver o entendimento da mensagem transmitida e se o conteúdo recebido for equivalente ao transmitido, no sentido de causar o efeito esperado.

Ao contrário de algumas situações onde o importante é saber apenas se o receptor recebeu uma mensagem, na colaboração é importante assegurar-se do entendimento da mesma. Sem um entendimento compartilhado, os participantes terão dificuldade em se coordenar de modo a somar seus esforços para a conclusão das tarefas. Porém, não há como inspecionar se o conteúdo recebido é equivalente ao enviado e se ele foi assimilado pelo receptor. A única forma de se obter indícios do entendimento é através das ações (e reações) do receptor, pois as mesmas são guiadas por seus conhecimentos. Uma falha na comunicação seria então uma discordância entre as expectativas do emissor e as ações do receptor. Algumas vezes estas falhas são identificadas no discurso do receptor, e outras, em suas atitudes.

Dada a importância da percepção para viabilizar o canal de transmissão de informação na comunicação, deve-se projetar e avaliar cuidadosamente nos ambientes virtuais colaborativos os elementos disponíveis para o emissor codificar sua mensagem e os elementos de percepção para que o receptor receba a mesma.

Os membros de uma equipe de trabalho têm necessidade de se comunicar de diversas maneiras. O coordenador da equipe deve escolher uma ferramenta de comunicação apropriada para cada situação e objetivo. O groupware utilizado deve

fornecer uma gama de ferramentas, pois algumas vezes uma ferramenta de comunicação assíncrona é mais apropriada, enquanto em outras, uma síncrona atende melhor. Ferramentas de comunicação assíncrona são utilizadas quando se deseja valorizar a reflexão dos participantes, pois estes terão mais tempo antes de agir. Em uma ferramenta de comunicação síncrona, valoriza-se a interação, visto que o tempo de resposta entre a ação de um participante e a reação de seus companheiros é curto.

Deve-se também considerar, ao se escolher uma ferramenta de comunicação, a estruturação do discurso que ela possibilita e a sua interface. Algumas ferramentas de comunicação são voltadas para uma conversa desestruturada, enquanto outras favorecem uma estruturação em lista, em árvore ou em grafo [Gerosa et al., 2001]. A interface da ferramenta normalmente é projetada para uma forma de utilização específica. Portanto, deve-se avaliar se a forma de estruturação e a interface da ferramenta atendem às necessidades do grupo em determinado momento.

Alguns exemplos de ferramentas de comunicação atualmente utilizadas são: e-mail, lista de discussão, fórum, ferramentas de CSCA (*Computer Supported Collaborative Argumentation*), ferramentas de votação, mensagem instantânea, chat, *eletronic brainstorming*, vídeo-conferência, tele-conferência, telefone, etc. [Long and Baecker, 1997].

3.2.2. Coordenação

Conversação para ação gera compromissos [Winograd and Flores, 1987] [Winograd, 1988]. Para garantir o cumprimento destes compromissos e a realização do trabalho colaborativo através da soma dos trabalhos individuais, é necessária a coordenação das atividades. Esta coordenação organiza o grupo para evitar que esforços de comunicação e cooperação sejam perdidos e que as tarefas sejam realizadas na ordem correta, no tempo correto e cumprindo as restrições e objetivos [Raposo et al., 2001].

Trabalho colaborativo foi definido por Karl Marx como “múltiplos indivíduos trabalhando juntos de maneira planejada no mesmo processo de produção ou em processos de produção diferentes, mas conectados” (citado em [Bannon and Schmidt, 1991]). No âmago desta definição está a noção de planejamento, garantindo que o trabalho coletivo seja resultante do conjunto de tarefas individuais.

A noção de planejamento presente na definição de Marx é realizada em CSCW pelo chamado trabalho de articulação, que é o esforço adicional necessário para a colaboração ser obtida a partir da soma dos trabalhos individuais. Fazem parte do trabalho de articulação a identificação dos objetivos, o mapeamento destes objetivos em tarefas, a seleção dos participantes, a distribuição das tarefas entre eles e a coordenação da realização das atividades.

A coordenação envolve tanto a pré-articulação das atividades, que corresponde às ações necessárias para preparar a colaboração, normalmente concluídas antes do trabalho colaborativo se iniciar, e o gerenciamento do aspecto dinâmico da colaboração, renegociada de maneira quase contínua ao longo de todo o tempo. Olhando apenas para esse aspecto dinâmico e contínuo da coordenação, ela pode ser definida como “o ato de

gerenciar interdependências entre as atividades realizadas para se atingir um objetivo" [Malone and Crowston, 1990]. Apesar da interdependência normalmente positiva entre as tarefas na colaboração (um participante desejando que o trabalho do outro seja bem sucedido), ela nem sempre é harmoniosa. Sem coordenação, há o risco de os participantes se envolverem em tarefas conflitantes ou repetitivas.

Algumas atividades envolvendo múltiplos indivíduos não exigem um planejamento formal. Atividades ligadas às relações sociais são bem controladas pelo chamado protocolo social, caracterizado pela ausência de qualquer mecanismo de coordenação explícito entre as atividades e pela confiança nas habilidades dos participantes de mediar as interações. Exemplos de ferramentas para este tipo de atividade são a maioria dos *chats* e as áudio e videoconferências.

Por outro lado, atividades mais diretamente voltadas para o trabalho colaborativo (e não para as relações sociais) exigem sofisticados mecanismos de coordenação para garantir o sucesso da colaboração. Exemplos de ferramentas que dão suporte a este tipo de atividade são gerenciamento de fluxo de trabalho (*workflow)*, *learningware*, jogos multi-usuários e ferramentas de autoria e de desenvolvimento de software colaborativo.

Na prática, entretanto, nem sempre é claro o que deve ficar a cargo do protocolo social e o que deve ter um mecanismo de coordenação associado. É tarefa do desenvolvedor de groupware a decisão sobre como será feita a coordenação de cada uma das atividades a serem realizadas. O ideal é que sistemas colaborativos não imponham padrões rígidos de trabalho ou de comunicação. Deve-se prover facilidades que permitam aos usuários interpretar e explorar estes padrões, decidir usá-los, modificá-los ou rejeitá-los [Schmidt, 1991]. O grande desafio ao se propor mecanismos de coordenação para o trabalho colaborativo consiste em torná-los suficientemente flexíveis para se adequar ao dinamismo da interação entre os participantes.

Para a coordenação do grupo são essenciais informações de percepção. É importante que cada um conheça o progresso do trabalho dos companheiros: o que foi feito, como foi feito, o que falta para o término, quais são os resultados preliminares, etc. As informações de percepção são necessárias principalmente durante a fase dinâmica da coordenação, para transmitir mudanças de planos e ajudar a gerar o novo entendimento compartilhado. Elas ajudam a medir a qualidade do trabalho com respeito aos objetivos e progressos do grupo e a evitar duplicação desnecessária de esforços [Dourish and Belloti, 1992].

Estas informações são especialmente úteis para o coordenador do grupo, que precisa saber, por exemplo, quem está ou não está trabalhando, entre quem estão ocorrendo conflitos de interesse, assim como as habilidades e as experiências de cada um. Um ambiente de groupware deve prover elementos de percepção que forneçam estas informações. Deve-se, porém, atentar para o fluxo de informações disponibilizadas para o coordenador. A princípio, quase todas as informações sobre o que acontece, aconteceu ou acontecerá no grupo têm alguma importância. Mas um excesso de informações pode dificultar a tomada de decisões.

Conflitos podem ocorrer devido a problemas de comunicação ou de percepção, ou por diferenças na interpretação da situação ou de interesse [Putnam and Poole, 1987]. A coordenação deve tratar os conflitos que prejudiquem o grupo, como competição, desorientação, problemas de hierarquia, difusão de responsabilidade, etc. [Salomon and Globerson, 1989].

3.2.3. Cooperação

Cooperação é a operação conjunta dos membros do grupo no espaço compartilhado. Em um espaço virtual de informação, os indivíduos cooperam produzindo, manipulando e organizando informações, bem como construindo e refinando artefatos digitais, como documentos, planilhas, gráficos, etc. O ambiente pode fornecer ferramentas de gerenciamento destes artefatos, como por exemplo, registro e recuperação de versões, controle e permissões de acesso, etc.

O registro da informação visa aumentar o entendimento entre as pessoas, reduzindo a incerteza (relacionada com a ausência de informação) e a equivocalidade (relacionada com a ambigüidade e com a existência de informações conflitantes) [Daft and Lengel, 1986]. Os indivíduos trabalham as informações e se comunicam na tentativa de solucionar os desentendimentos.

A forma de garantir a "memória" do grupo nos projetos colaborativos é preservando, catalogando, categorizando e estruturando a documentação produzida pelos participantes. Este tipo de conhecimento pode ser encarado como conhecimento formal. Entretanto, o conhecimento dito informal, isto é, idéias, fatos, questões, pontos de vista, conversas, discussões, decisões, etc. que ocorrem durante o processo e acabam por defini-lo, é difícil de ser capturado, porém permite recuperar o histórico da discussão e o contexto em que as decisões foram tomadas.

Ao registrar, organizar e ligar as informações trocadas durante o projeto colaborativo aos artefatos digitais, pode-se investigar o raciocínio que levou a um determinado artefato (*design rationale*) e averiguar posteriormente, em um novo contexto, se os motivos pelos quais as decisões de projeto foram tomadas continuam sendo válidos. Se o novo contexto invalida os argumentos que embasaram as decisões, pode-se reprojetar o artefato. Quando este raciocínio por trás das decisões não está disponível, a identificação dos motivos pelos quais o artefato foi projetado e das técnicas utilizadas fica dificultada.

Há diversas ferramentas na literatura que utilizam o hipertexto para a organização da memória do grupo [Shum and Hammond, 1994]. Algumas destas ferramentas possibilitam *linkar* os artefatos digitais ao espaço compartilhado, explicitando nestas ligações as interações que os originaram. Com isto, os contextos dos artefatos e das interações são preservados, facilitando o seu entendimento e a posterior recuperação. A memória do grupo passa a ser formada então pelos artefatos (memória do produto) e pelas redes de informações compostas pelos fatos, hipóteses, restrições, decisões, argumentos, significados dos conceitos, etc. (memória do processo).

Algumas destas ferramentas são voltadas para o apoio ao desenvolvimento de software. O Beyond-Sniff [Bischofberger et al., 1994] permite aos usuários de um software registrar anotações associadas a suas funcionalidades. O Evolving Artifact [Ostwald, 1995] integra documentação baseada em hipertexto com protótipos e possibilita, a partir da documentação, executar o protótipo, que fica inserido no contexto da documentação e os usuários podem registrar seus comentários e críticas ao interagir com o software. O artefato-protótipo possibilita que os envolvidos no processo do desenvolvimento reflitam sobre as conseqüências do projeto [Schön, 1983] [Schön and Bennet, 1996] [Schrage, 1996].

3.2.4. Percepção

Perceber[2], neste contexto, é adquirir informação, por meio dos sentidos, do que está acontecendo e do que as outras pessoas estão fazendo, mesmo sem se comunicar diretamente com elas [Brinck and McDaniel, 1997]. A percepção, que é inerente ao ser humano, torna-se central para a comunicação, coordenação e cooperação de um grupo de trabalho. Os indivíduos tomam ciência das mudanças causadas no ambiente pelas ações dos participantes, e redirecionam as suas atitudes.

Na interação entre pessoas e ambiente dentro de uma situação face-a-face, a obtenção de informações é rica e natural, visto que os sentidos estão presentes em sua plenitude. Em ambientes virtuais, o suporte à percepção fica menos claro, pois os meios de transmitir as informações aos órgãos sensoriais dos seres humanos são restritos. Estações de trabalho típicas são limitadas a fornecer informações em uma tela com apenas duas dimensões e, em alguns casos, através de caixas de som. Por outro lado, em um ambiente virtual pode-se filtrar os eventos de forma a reduzir dispersões com informações irrelevantes, que normalmente permeiam uma situação face-a-face.

Elementos de percepção são os elementos do espaço compartilhado por onde são transmitidas as informações destinadas a prover percepção. Estas informações auxiliam os indivíduos a dirigir suas ações, interpretar eventos e prever possíveis necessidades. Perceber as atividades dos outros indivíduos é essencial para garantir o fluxo e a naturalidade do trabalho, assim como para diminuir as sensações de impessoalidade e distância, comuns nos ambientes virtuais.

Um projeto adequado dos elementos de percepção possibilita que os participantes tenham disponíveis informações necessárias para prosseguir seu trabalho, sem ter que interromper seus colegas para solicitá-las. Os ambientes de colaboração devem prover informações necessárias para o trabalho coletivo e o individual, de forma que os participantes possam criar um entendimento compartilhado e construir o seu contexto de trabalho. Alguns exemplos de informações de percepção que podem ser providas são: o objetivo comum, o papel de cada um dentro do contexto, o que fazer, como proceder, qual o impacto das ações, até onde atuar, quem está por perto, o que o companheiro pode fazer, o que as outras pessoas estão fazendo, a localização, a origem, a importância, as relações e a autoria dos objetos de cooperação.

---

[2] O termo "percepção" é uma tradução do inglês *awareness*.

O projetista de um ambiente virtual deve prever quais informações de percepção são importantes, como elas podem ser capturadas ou geradas, onde elementos de percepção são necessários, de que forma apresentá-los e como dar aos indivíduos o controle sobre eles. A escolha da forma adequada de implementar estes elementos e a utilização de filtros e personalização do recebimento das informações ajuda a evitar a má interpretação dos dados e a sobrecarga de informação.

Uma quantidade não gerenciável de informações dificulta a organização dos membros do grupo, ocasionando desentendimentos [Fussel et al, 1998]. Vale ressaltar, que a existência da sobrecarga de informação está extremamente ligada ao indivíduo. Uns conseguem lidar com mais informações simultâneas do que outros, dependendo, entre outros fatores, da maturidade, das capacidades e das habilidades de cada um, bem como das características e do nível de conhecimento sobre o assunto em questão. Deve haver um controle para que o fluxo de informações não seja maior do que a capacidade do indivíduo de processá-la e digeri-la, apesar desta capacidade não ser facilmente mensurável.

Para evitar a sobrecarga, é necessário balancear a necessidade de fornecer informações com a de preservar a atenção sobre o trabalho. O fornecimento de informações na forma assíncrona, estruturada, filtrada, agrupada, resumida e personalizada facilita esta tarefa [Kraut and Attewell, 1997]. Deve-se fornecer uma visão geral para que o indivíduo selecione em que parte da informação deseja trabalhar, e mais detalhes são obtidos quando forem demandados. A redução da sobrecarga de informação na comunicação, por exemplo, pode se dar através da estruturação do diálogo e do fornecimento de informações simples e representativas que ajudem os participantes a identificar a relevância e o contexto das mensagens, sem que estas sejam lidas completamente [Gerosa et al, 2001].

O modelo de colaboração apresentado nesta seção e nas anteriores foi aplicado no projeto do AulaNet, conforme será visto mais adiante.

3.3. Requisitos de Groupware

Nesta seção são levantados requisitos de funcionalidades de um groupware com relação a seus usuários e desenvolvedores, adaptados de [Tietze, 2001]. Tanto os aspectos da colaboração elaborados na seção anterior quanto os requisitos listados a seguir fornecem subsídios para analisar, avaliar e desenvolver ferramentas de groupware.

3.3.1. Requisitos do Usuário

O levantamento de requisitos será iniciado pelo ponto de vista dos usuários finais, ou seja, dos indivíduos que irão utilizar o groupware. Os requisitos são apresentados através de cenários de uma empresa fictícia, cujos funcionários André, Bia e Carlos estão situados em localidades distintas e utilizam ferramentas de groupware para colaborar na construção de artefatos digitais.

RU1 – Acesso aos objetos compartilhados e às ferramentas de colaboração

Ao chegar em sua sala, André liga sua estação de trabalho e entra no ambiente de groupware. Na área de trabalho do ambiente, aparecem alguns documentos compartilhados com os quais ele estava trabalhando no dia anterior, pastas compartilhadas que servem de repositórios para artefatos, e algumas ferramentas, como um calendário com seus compromissos diários, sua caixa de mensagens e outras.

Neste cenário, pode-se levantar um requisito de groupware, que deve prover fácil acesso aos objetos e às ferramentas de colaboração, fornecendo compartilhamento e persistência.

RU2 – Auxílio na escolha das ferramentas apropriadas

O objetivo do grupo de trabalho do qual André faz parte é desenvolver colaborativamente um complexo relatório técnico. André está encarregado de um dos capítulos, que é acessível através de um dos ícones de sua área de trabalho. Ao indicar ao ambiente que ele deseja abrir o documento, automaticamente é acionada a ferramenta capaz de lidar com aquele documento.

Neste cenário, pode-se observar um requisito para o groupware: ele deve saber qual a ferramenta apropriada para executar determinada tarefa em um tipo de objeto. Caso haja mais de uma ferramenta, o sistema deve auxiliar o usuário a escolher qual a mais adequada às suas necessidades e preferências.

RU3 – Elementos de percepção

Enquanto André estava trabalhando no seu texto, notou um novo ícone na lista de usuários conectados, indicando que Bia entrou no ambiente. Um dos itens da área de trabalho de Bia é o capítulo em que André está trabalhando, isto é sinalizado a ela por um indicador ao lado do ícone.

Este cenário ilustra a necessidade de prover informações de percepção do grupo através de elementos de percepção do ambiente. Por meio destas informações, os participantes se coordenam e montam seu contexto de trabalho, tendo indicativos das ações e presença de seus companheiros.

RU4 – Colaboração síncrona e assíncrona

André encontra um problema no texto em que está trabalhando. Como o histórico do documento informa que Bia o utilizou recentemente, ele resolve contatá-la. Bia recebe então um convite para que participe da edição cooperativa do documento. Ao aceitá-lo, automaticamente o editor apropriado é aberto em seu ambiente. André também abriu um canal de voz com Bia para os dois se comunicarem durante a edição. André explica o problema para Bia, que escreve no documento uma proposta de solução. Os dois se comunicam e editam o documento (cooperam) até que entendem que o problema foi resolvido. Deixam então uma nota para Carlos, o gerente do projeto, informando-o da alteração que fizeram.

Este cenário ilustra a necessidade de que o ambiente forneça serviços de colaboração entre os participantes, tanto de forma síncrona quanto assíncrona.

RU5 – Acesso ao ambiente independente da estação de trabalho

Ao continuar colaborando com Bia, André notou que faltavam alguns dados a serem obtidos. Saiu de sua estação de trabalho e se locomoveu até o laboratório. Através de um dos computadores, ele entrou no ambiente, que permanecia da mesma forma que havia deixado ao sair de sua sala. Desta forma, pôde continuar colaborando com Bia e colocar os dados coletados no laboratório diretamente no documento.

Este cenário ilustra a necessidade de independência do ambiente com relação à máquina onde ele está sendo executado. Quando este requisito for atendido, os usuários terão o ambiente da mesma forma que o deixaram, mesmo que ele esteja sendo utilizado em uma máquina diferente.

RU6 – Espaço privativo e público e a transição entre eles

Enquanto colaborava com André, Bia estava com outra sessão do editor aberta, visível apenas a ela, onde registrava algumas idéias. Em um certo momento, ela organiza suas idéias e decide compartilhá-las com André. Ela libera seu acesso à sessão de edição e após colaborarem no texto, resolvem incorporá-lo como um novo parágrafo no capítulo que estão editando.

Em certos momentos, as pessoas preferem atuar individualmente, e em outros, coletivamente. O ambiente deve prover recursos para facilitar as duas formas de trabalho e a transição entre elas.

RU7 – Extensão dinâmica do ambiente

Ao trabalhar em conjunto no capítulo do relatório técnico, André e Bia têm necessidade de incorporar algumas figuras que ilustrem determinado conceito. Porém, não há no ambiente nenhuma ferramenta de edição gráfica. André havia feito recentemente o *download* (transferência) de uma ferramenta desta natureza. Para usá-la no trabalho colaborativo, carrega-a no ambiente, que por sua vez a disponibiliza para todos os usuários.

Neste cenário, percebe-se a necessidade de o ambiente ser capaz de incorporar e disponibilizar aos usuários novas ferramentas, sem ser reinicializado.

RU8 – Sincronização entre ferramentas diferentes

Bia abre o documento que estava editando com André em uma ferramenta que fornece uma visão global do mesmo, mostrando a estrutura de seções do documento de forma gráfica. Ela resolve então alterar de posição uma das seções. Quando faz isto, o texto que estava no editor compartilhado é atualizado, tanto em sua máquina quanto na de André.

Este cenário ilustra a necessidade de que diferentes ferramentas, operando sobre um mesmo objeto compartilhado, façam a sincronização e atualização do mesmo.

RU9 – Mobilidade

Carlos, o gerente da equipe, checa o andamento do trabalho, se conectando ao sistema através de seu PDA (*Personal Digital Assistant*), no intervalo de uma reunião. Aparece para ele uma versão simplificada de sua área de trabalho, dentro dos limites de exibição de seu dispositivo. Através de elementos de percepção disponíveis na interface, ele pode identificar quem está conectado e o que cada um está fazendo. Ele abre a ferramenta que mostra a estrutura do relatório para ver o que foi alterado desde sua última visita. À medida que André e Bia mexem na estrutura do capítulo que estão editando, as mudanças são refletidas na tela de Carlos.

Este cenário ilustra a necessidade de prover acesso através de dispositivos móveis, como PDAs e celulares, de forma a possibilitar a conexão mesmo longe de uma estação de trabalho. Como a interface neste tipo de dispositivo geralmente é mais limitada do que as presentes nas estações de trabalho, o ambiente deve prever simplificações que possam tornar a exibição possível.

RU10 – Agrupamento de ferramentas

Durante a colaboração com André, Bia utilizou um editor de documentos, um editor gráfico, uma ferramenta de comunicação por voz e outra por texto. Para não ter que remontar este contexto na próxima colaboração, ela cria uma nova ferramenta incorporando as ferramentas utilizadas. Na próxima vez que for colaborar com André, ela poderá utilizar diretamente a nova ferramenta e automaticamente será recomposto o contexto de trabalho.

O sistema deve prover recursos para que os usuários agrupem ferramentas. Quando determinada situação de colaboração ocorrer novamente, as diversas ferramentas envolvidas poderão ser abertas de uma vez e assumirão a configuração previamente estabelecida.

RU11 – Alta performance

Ao colaborar, os usuários esperam que o seu trabalho flua sem atrasos devido à latência do sistema e que as modificações feitas nos artefatos do ambiente de trabalho sejam instantaneamente refletidas para seus companheiros. Entretanto, diferentes tipos de ferramenta têm requisitos distintos de *feedback* (modificações devido a ações do indivíduo) e de *feedthrough* (modificações devido a ações dos companheiros). De forma geral, as ferramentas de colaboração síncronas têm requisitos mais fortes de *feedthrough* do que as ferramentas assíncronas, e operações interativas têm requisitos mais fortes de *feedback* do que operações em lote.

RU12 – Uso das ferramentas de trabalho individual para o coletivo

André resolve abrir um tipo de documento que ainda não havia sido usado no ambiente colaborativo. Para trabalhar com este documento, ele deseja usar a mesma ferramenta com a qual ele trabalha individualmente neste tipo de documento.

Este cenário ilustra a necessidade de o ambiente prover recursos para que ferramentas de trabalho mono-usuárias sejam usadas de forma colaborativa, evitando que os usuários tenham que aprender a trabalhar com ferramentas distintas das usuais.

3.3.2. Requisitos do Desenvolvedor

O desenvolvedor utiliza a infra-estrutura disponível, seja ela uma arquitetura de componentes ou mesmo a extensão de uma linguagem de programação, visando a criação e manutenção de ferramentas colaborativas. Nesta seção, são analisados requisitos do groupware do ponto de vista dos desenvolvedores.

RD1 – Reuso da experiência e conhecimento anteriores

Ao construir ferramentas colaborativas, a experiência e o conhecimento dos desenvolvedores sobre programação de aplicações mono-usuários devem ser aproveitados ao máximo para reduzir o tempo necessário para que um novo membro se integre à equipe de desenvolvimento.

RD2 – Aproveitamento do modelo de dados

A infra-estrutra em questão deve prover recursos para que os desenvolvedores aproveitem modelos de dados já existentes. Ao reaproveitá-los, a integração de diferentes ferramentas colaborativas é facilitada e o tempo de projeto e implementação é reduzido, pois se aproveitam esforços anteriores.

RD3 – Compartilhamento transparente de dados

Não deve ficar a cargo do programador a preocupação com a distribuição e compartilhamento de dados pelo sistema. A infra-estrutura deve prover recursos para gerenciar o acesso, a alocação e o compartilhamento dos dados.

RD4 – Suporte a dados locais e compartilhados

Nem todos os dados, mesmo em uma ferramenta colaborativa, devem ser compartilhados. Alguns podem ficar localizados na máquina do cliente e outros no servidor. O desenvolvedor deve decidir se o dado vai ser local ou compartilhado, e a infra-estrutura deve prover recursos para ambos os casos.

RD5 – Acesso às informações de percepção

O desenvolvedor das aplicações colaborativas deve ter fácil acesso às informações relativas ao contexto da colaboração, ou seja, às informações necessárias para prover ao indivíduo percepção sobre as atividades e localização de seus companheiros. Ele pode

incrementar sua aplicação com informações que auxiliem os participantes a se comunicarem, se coordenarem e cooperarem.

RD6 – Disponibilização de novas ferramentas

Assim que uma nova ferramenta ou uma nova versão de uma ferramenta existente ficar pronta, o desenvolvedor não deve ter dificuldades em distribuí-la para os usuários finais. O acesso à nova ferramenta pelos usuários deve-se dar simultaneamente para evitar utilização de versões diferentes da mesma ferramenta.

RD7 – Escalabilidade

A performance do sistema não deve se degradar perceptivelmente na medida que novos usuários se conectam.

RD8 – Integração com ferramentas externas

A infra-estrutura deve ser adaptável o suficiente para possibilitar a integração do ambiente com ferramentas externas, adquiridas de terceiros ou não.

RD9 – Suporte às ferramentas localizadas no servidor

Algumas operações em um ambiente colaborativo são melhor realizadas quando centralizadas devido, por exemplo, à necessidade de acesso a determinados recursos ou de execução ininterrupta. A infra-estrutura deve prover, portanto, recursos para que algumas ferramentas sejam executadas no servidor, sendo que a forma de invocar estas ferramentas não deve diferir da forma de invocar ferramentas presentes nas máquinas dos usuários.

Os requisitos listados acima são retomados nas próximas seções, onde exemplos e arquiteturas de groupware são analisados.

3.4. O Modelo de Colaboração e os Requisitos em Ferramentas de Groupware

Esta seção aborda três classes de ferramentas de groupware tomando como base o modelo de colaboração e os requisitos apresentados anteriormente. Sistemas de videoconferência e ambientes virtuais colaborativos serão analisados de uma maneira mais genérica, sem enfocar uma ferramenta específica. O exemplo de *learningware*, groupware dedicado à aprendizagem, será analisado mais detalhadamente considerando o AulaNet, um ambiente de ensino e aprendizagem desenvolvido pelos autores.

3.4.1. Videoconferência

Videoconferência é a denominação de sistemas que tenham as seguintes características: usa imagens de vídeo (de pessoas, lugares ou objetos) para enriquecer o processo de comunicação, e a transmissão desse vídeo (e respectivo áudio) é feita nas duas direções,

interativamente [Rhodes, 2001]. A Figura 3.4 mostra a interface integrada do VIC[3], um software de videoconferência de código aberto desenvolvido pelo University College de Londres.

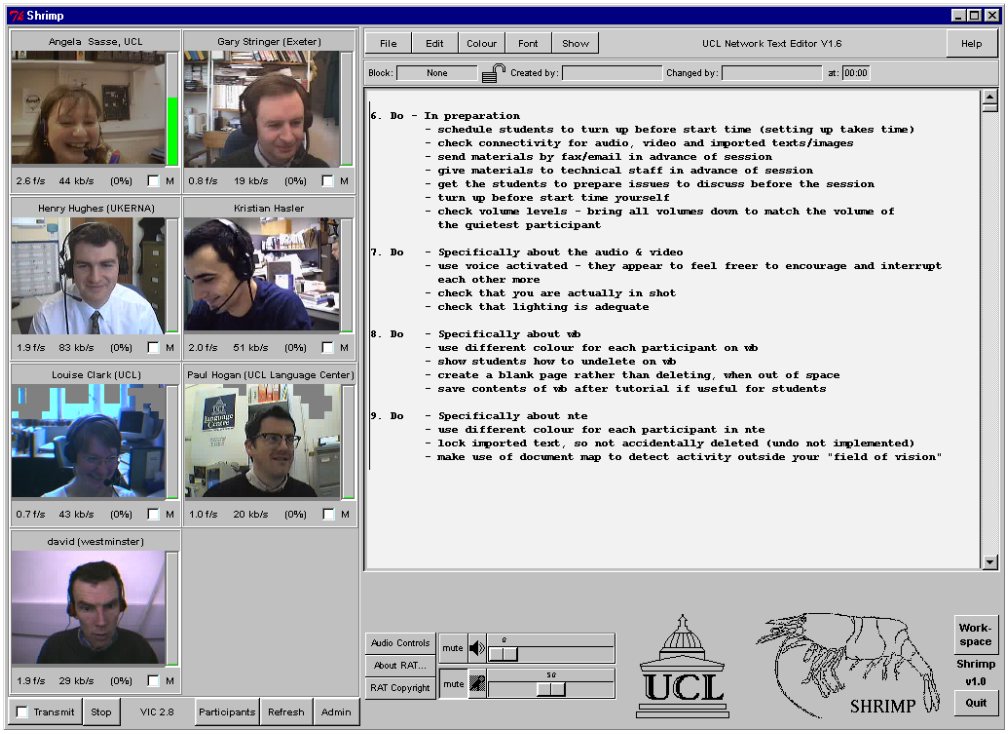

Figura 3.4. Interface integrada do VIC

A videoconferência surgiu basicamente como ferramenta de comunicação. Atualmente, muitos equipamentos e aplicações de videoconferência são ferramentas de colaboração completas. Além da capacidade de comunicação, está incluída a cooperação, realizada por meio do compartilhamento de documentos, imagens, programas, etc., e a coordenação, que aparece na forma da interface de controle do equipamento de videoconferência, incluindo funções como a discagem para a conexão, o gerenciamento de sessão, passagem de controle de aplicações compartilhadas, etc. Exemplos de ferramentas nessa categoria são o NetMeeting[4], o Webex[5] e o Click to Meet[6].

Sistemas de videoconferência englobam uma gama de equipamentos, que podem variar desde simples conferências *desktop* pela Internet, cujo custo é basicamente o de uma *Webcam*, até sofisticadas salas projetadas especificamente para esse fim, cujos custos podem chegar a milhões de dólares.

---

[3] http://www-mice.cs.ucl.ac.uk/multimedia/software

[4] http://www.microsoft.com/netmeeting

[5] http://www.webex.com

[6] http://www.cuseeme.com/products/clicktomeet.htm

A utilização de videoconferência é uma solução relativamente simples para se conseguir uma interação imediata com pessoas geograficamente distantes. Em grandes empresas, esta solução está sendo cada vez mais utilizada para diminuir custos, tempo e riscos envolvidos nas viagens. Ela possibilita a participação de pessoas que não estariam presentes caso a reunião ocorresse em outros locais, seja devido à limitação com gastos de viagens ou por falta de disponibilidade de tempo para deslocamento. Em cenários mais sofisticados, a videoconferência ainda pode ser usada para simular a telepresença. Isso é útil, por exemplo, em situações de emergência onde especialistas podem controlar a situação a partir da sede, tendo contato direto com operadores no local da situação.

O grande problema com relação aos sistemas de videoconferência é que eles ainda são relativamente caros, e os resultados podem não corresponder às expectativas, principalmente nas transmissões via Internet.

Sistemas de videoconferência, quando integrados ao ambiente de trabalho (por exemplo, o Netmeeting integrado ao *desktop* Windows e o Webex integrado ao navegador Web), possibilitam o acesso aos objetos compartilhados e às ferramentas de colaboração (RU1), bem como determinam a ferramenta apropriada para cada tipo de documento (RU2), que muitas vezes é a ferramenta mono-usuário usada no trabalho individual (RU12). Algumas destas ferramentas provêem um espaço privativo de trabalho (RU6). A percepção (RU3) é uma conseqüência natural da transmissão de vídeo, dado o contato visual com os outros usuários. De uma maneira geral, ferramentas de videoconferência não são projetadas para dar suporte à colaboração assíncrona (RU4), embora algumas permitam a gravação do vídeo.

Um outro requisito que começa a ser atendido por certos sistemas de videoconferência é o da mobilidade (RU9), pois já existem sistemas compactos para serem transportados e instalados em qualquer plataforma. Os requisitos de desenvolvedor não serão considerados, pois neste caso estamos tratando apenas da visão do usuário de videoconferência.

3.4.2. Ambientes Virtuais Colaborativos

Os ambientes virtuais colaborativos, ou CVEs (*Collaborative Virtual Environments*), são simulações de mundos reais ou imaginários onde vários usuários podem interagir em tempo real, compartilhar informações e manipular objetos no ambiente [Hagsand, 1996], [Singhal and Zyda, 1999]. Os CVEs vão além da metáfora de *desktop* da maioria das aplicações atuais, propondo comunidades virtuais onde as interações são modeladas de acordo com as do mundo real, utilizando recursos da realidade virtual.

Sistemas experimentais do tipo CVE já são usados há décadas, mas apenas recentemente começaram a sair das esferas acadêmicas e militares. Esse aumento de popularidade se deve principalmente ao rápido aumento da capacidade de processamento das máquinas e sua redução de custos. Entre os conhecidos estão o

Active Worlds[7] (Figura 3.5), The Palace[8], o Blaxxun[9] e o DIVE[10], sendo estes dois últimos plataformas para o desenvolvimento de CVEs.

Figura 3.5. Interface do Active Worlds

Apesar da crescente popularidade, CVEs ainda apresentam uma série de desafios em seu desenvolvimento. Entre esses desafios, podem ser mencionados os problemas relacionados ao gerenciamento de recursos de rede (controle de concorrência, perda de dados, escalabilidade, etc.), às aplicações gráficas em tempo real (por exemplo, alocação de CPU para geração das imagens) e às aplicações multi-usuários (por exemplo, manutenção de consistência entre os usuários). Também há as dificuldades específicas da área de aplicação do CVE, tais como a integração com grandes bases de dados (por exemplo, para as informações geográficas de simulações militares) e a autenticação de usuários (para aplicações de comércio eletrônico, por exemplo).

Quando o campo de aplicação é especificamente a realização de trabalhos colaborativos, somam-se os desafios relacionados a CSCW. Para mencionar alguns, há a dificuldade de se trabalhar com os objetos do mundo virtual [Benford et al., 1994] e a necessidade de se criar avatares[11] realistas para ampliar a capacidade de comunicação entre os participantes e seu sentimento de presença [Joslin et al., 2001].

---

[7] http://www.activeworlds.com

[8] http://www.thepalace.com

[9] http://www.blaxxun.com/solutions/applications/virtualworlds/index.shtml

[10] http://www.sics.se/dive

[11] Avatar é a representação do usuário no ambiente virtual. Normalmente esta representação é feita por meio de uma entidade que lembre a figura humana.

Apesar das dificuldades, os CVEs apresentam grande potencial para o suporte ao trabalho colaborativo e, por conseguinte, têm sido desenvolvidos dando grande atenção aos resultados de CSCW. Em particular, eles apresentam características bastante interessantes no que diz respeito a comunicação, cooperação e percepção. A comunicação aparece na forma de ferramentas de bate-papo, que acompanham os CVEs mais simples, ou na forma de áudio e videoconferência, que acompanham os CVEs mais sofisticados. A capacidade de comunicação é ampliada em CVEs pela noção de espaço oferecida (por exemplo, um avatar se aproximando do outro pode indicar que ele deseja se comunicar). A cooperação faz parte da própria natureza dos CVEs, que são literalmente espaços de trabalho compartilhados. Por imitarem metáforas do mundo real, CVEs trazem possibilidades de percepção que não são triviais em aplicações de *desktop*. Por exemplo, os avatares e suas localizações fornecem informações diretas sobre os usuários presentes no ambiente e suas atividades. A movimentação dos mesmos antecipa intenções dos usuários e revela a expectativa que os demais usuários têm a seu respeito.

Com relação à coordenação, entretanto, ainda existe uma certa limitação por parte dos CVEs [Raposo et. al, 2001a]. Por causa dessa deficiência, eles têm sido utilizados basicamente para atividades coordenadas pelo protocolo social, como por exemplo, para navegação por ambientes 3D e comunicação com usuários remotos. Os caros CVEs para aplicações militares e treinamento são notórias exceções.

Os CVEs, de uma maneira geral, atendem bem aos requisitos de usuários de RU1 a RU6. Ao entrar em um ambiente virtual, o usuário tem acesso aos objetos compartilhados (RU1) e o ambiente pode ajudar na escolha das ferramentas apropriadas para se trabalhar com eles, através de chamadas a programas externos quando necessários (RU2). Os elementos de percepção (RU3), como comentado anteriormente, estão presentes em CVEs. Há CVEs com persistência de estado, de modo que a interação também possa ocorrer de forma assíncrona (RU4), o que possibilita o acesso ao ambiente virtual independente da máquina em que se trabalha (RU5). Alguns CVEs possibilitam a criação de áreas particulares (RU6). Problemas como a utilização em dispositivos móveis (RU9) e otimização de performance (RU11) também são preocupações dos desenvolvedores de CVEs, embora ainda não se possa dizer que eles são plenamente atendidos na maioria dos atuais.

Com relação aos requisitos do desenvolvedor, os CVEs não atendem ao requisito RD1, visto que CVEs não reusam o modelo de programação das aplicações mono-usuários. Já os requisitos RD2 e RD3 são satisfeitos na maioria dos CVEs que provêem suporte para o compartilhamento e distribuição dos dados. Vários CVEs têm arquitetura híbrida, com dados centralizados e replicados (RD4 e RD9). O RD5 (percepção) é plenamente satisfeito, enquanto o RD7 (escalabilidade) é meta de projeto de vários CVEs, embora não se possa dizer que seja plenamente atendido na maior parte dos CVEs atuais. A integração com ferramentas externas (RD8) é possível em alguns casos.

3.4.3. Learningware

No projeto Internet2[12], *learningware* é a expressão utilizada para denominar o groupware dedicado à aprendizagem colaborativa na Web. O AulaNet é um ambiente gratuito baseado em uma abordagem *learningware* para a criação, aplicação e gerenciamento de cursos pela Internet. Ele vem sendo desenvolvido desde Junho de 1997 pelo Laboratório de Engenharia de Software da Universidade Católica do Rio de Janeiro (PUC-Rio)[13].

O AulaNet é organizado em serviços de comunicação, de coordenação e de cooperação. Os serviços são colocados à disposição do docente durante a criação e a atualização do curso, permitindo a ele selecionar e configurar quais ficarão disponíveis aos participantes. Estes serviços ficam acessíveis através de um menu representado graficamente como um controle remoto (Figura 3.6).

Figura 3.6. Interface do AulaNet destacando o controle remoto e um conteúdo na forma de vídeo

Os serviços de comunicação fornecem as facilidades que permitem a troca e o envio de informações. Estes serviços incluem ferramentas de discussão textual assíncrona no estilo de fórum (Conferências), de bate-papo síncrono textual no estilo de *chat* (Debate), de troca de mensagens instantâneas com participantes simultaneamente conectados (Mensagens para Participantes), e de correio eletrônico individual com o mediador (Contato com os Docentes) e com toda a turma (Lista de Discussão).

---

[12] http://www.internet2.org

[13] http://www.les.inf.puc-rio.br/aulanet

Um indivíduo trabalhando sozinho lida com sua própria base de conhecimento e de fontes de informação, e deve se organizar para resolver as tarefas. Ao trabalhar em grupo, diversos problemas complexos de coordenação aparecem. Os serviços de coordenação visam minimizar estes problemas, organizando o grupo para possibilitar a cooperação, com mecanismos de gerenciamento da agenda do grupo e da competência, entre outros. No AulaNet os serviços de coordenação incluem uma ferramenta de notificação (Avisos), uma ferramenta de coordenação básica do fluxo do curso (Plano de Aulas), ferramentas de avaliação (Tarefas e Exames) e uma ferramenta de acompanhamento da participação do grupo (Relatórios de Participação).

Os serviços de cooperação fornecem meios à aprendizagem colaborativa [Harasim et al, 1997], à resolução de problemas e à co-autoria de cursos. Apesar de a performance do grupo na resolução das tarefas ser extremamente dependente de fatores contextuais, como composição do grupo, características e habilidades dos membros, tipo do problema e suporte tecnológico, os serviços de cooperação dão suporte à interação entre os participantes de forma a minimizar as dificuldades contextuais. No AulaNet, os serviços de cooperação incluem uma lista de referências do curso (Bibliografia e Webliografia), uma lista de conteúdos transferíveis para consumo desconectado (Download) e facilidades de co-autoria, tanto de docentes (Co-autoria de Docente) quanto de aprendizes (Co-autoria de Aprendiz).

Faremos a seguir a análise dos requisitos de groupware satisfeitos pelo ambiente. A janela do controle remoto permite acesso a todos os serviços do ambiente (RU1), enquanto o fato de estar embutido em um ambiente Web garante o auxílio na escolha das ferramentas adequadas para cada tipo de objeto (*MIME type*) – RU2. Com relação aos elementos de percepção (RU3), eles estão espalhados por toda a interface do AulaNet. Por exemplo, o ambiente fornece uma lista de usuários conectados e o controle remoto indica se há novos conteúdos nas aulas ou novas mensagens, etc. Uma discussão completa sobre os elementos de percepção do AulaNet é feita em [Gerosa et al., 2001a].

O AulaNet dá suporte à comunicação síncrona e assíncrona (RU4) e, como a base de dados está centralizada em um servidor, o usuário pode ter acesso ao seu ambiente a partir de qualquer computador com acesso à Internet usando um navegador Web (RU5). O AulaNet possibilita a utilização de conteúdos gerados em ferramentas de trabalho individual (RU12), pois o ambiente não tem ferramentas de autoria. Há também a possibilidade de o coordenador do curso criar conteúdos não-certificados (não acessíveis aos aprendizes). Essas funcionalidades satisfazem parcialmente as necessidades de um membro do grupo no seu espaço privativo de trabalho (RU6). O ambiente AulaNet também provê a possibilidade de extensão dos formatos de conteúdos aceitos, por meio da inserção de novos *plugins* para o navegador (RU7).

Não é possível visualizar um mesmo tipo de documento com duas ferramentas diferentes pelo AulaNet (RU8), nem criar agrupamentos de ferramentas (RU10). Também não há suporte específico para dispositivos móveis (RU9) e otimização de performance (RU11), ficando restrito ao suporte oferecido pelos navegadores e *plugins* utilizados.

Com relação aos requisitos do desenvolvedor, o AulaNet permite tanto o reuso da experiência com programação mono-usuário (RD1) quanto dos modelos de dados existentes (RD2). O ambiente também gerencia os aspectos de compartilhamento de dados, que não ficam a cargo do desenvolvedor (RD3), e fornece acesso às informações de percepção (RD5).

Todos os objetos de cooperação estão localizados no servidor AulaNet, não havendo qualquer suporte a dados locais. Isso garante o cumprimento do requisito RD9 (suporte a componentes no servidor), mas vai de encontro ao RD4 (suporte a dados locais). O fato de ser totalmente centralizado não garante que a performance será mantida com um número elevado de usuários (RD7), pois o servidor pode ser o gargalo do sistema.

O AulaNet atualmente ainda não é baseado em componentes, daí o RD6 (disponibilização de novos componentes) ficar fora do escopo do ambiente. A integração com ferramentas externas (RD8) ocorre no nível de dados (no caso de integração de novos *plugins* para visualizar os conteúdos), mas não no nível de aplicação.

Nas seções anteriores, requisitos de groupware foram associados a aplicações colaborativas. Passaremos à fase de projeto, que é a seqüência do ciclo de desenvolvimento de software.

3.5. Engenharia de Groupware

Um groupware normalmente é composto por um conjunto de ferramentas colaborativas, que possibilitam a interação entre múltiplos usuários. Como os processos de trabalho entre os indivíduos são muito específicos e evoluem com o tempo, a tecnologia de groupware deve prover flexibilidade suficiente para ser adaptada às necessidades de cada grupo e à evolução dos processos de trabalho. O groupware deve prover ao grupo a possibilidade de montar seu contexto de trabalho, selecionando e configurando um conjunto de ferramentas colaborativas específicas para suas necessidades.

Uma aplicação normalmente está disponível como uma caixa-preta monolítica que provê um determinado conjunto de funcionalidades. Este tipo de software é pouco flexível, visto que há muitas inter-relações e interdependências amarradas em seu código fonte. Este engessamento impede que o usuário possa adaptá-lo às suas preferências e necessidades à medida que ganha experiência na utilização do software. Algumas aplicações oferecem extensibilidade ao invocar ferramentas externas, como por exemplo, um editor de programas que chama um compilador, ou um editor de HTML que chama um navegador para visualizar o arquivo. O resultado final vai continuar aparentando ser um conjunto de aplicações pouco integradas e com interface diversas.

Um pacote de aplicações (*application suite*) consiste em um conjunto de programas, normalmente de um mesmo fabricante, com algum nível de integração e uma aparência padronizada. Alguns exemplos de pacotes de aplicações são o Microsoft Office e o Netscape Communicator suite. Porém, nos pacotes não há muita flexibilidade

(nem sempre o usuário pode decidir quais partes do pacote deseja instalar) e não é possível a integração de ferramentas externas.

A utilização de técnicas de desenvolvimento baseado em componentes é uma forma de facilitar o desenvolvimento de groupware para que este seja mais flexível. Estas técnicas visam desenvolver sistemas modulares compostos de componentes de software, que podem ser adaptados e combinados na medida da necessidade, tendo em mente futuras manutenções. Um componente de software é definido como uma unidade de composição da qual se conhece apenas as interfaces, especificadas na forma de contratos, e as dependências de contexto [Szypersky, 1999]. Um componente pode fazer parte de outros componentes, e pode ser disponibilizado independentemente, bem como utilizado em composições por terceiros.

Ao utilizar um software cujo desenvolvimento foi baseado em componentes, o usuário aplica um conjunto de ferramentas de sua escolha a documentos e artefatos. Fazendo uma analogia, ao invés de apresentar ao usuário uma aplicação que coloca um prego ou parafuso na parede, ele é equipado com uma caixa de ferramentas contendo, entre outras coisas, martelo, chave de fenda e parafusadeira, de onde ele pode escolher qual a ferramenta mais adequada a suas preferências e à tarefa em questão. Adicionalmente, ferramentas existentes podem ser combinadas para formar novas ferramentas mais complexas para a realização de tarefas específicas.

Um *framework* agiliza o processo de desenvolvimento de software que lida com determinado tipo de problema, provendo recursos que fornecem flexibilidade para os desenvolvedores adequarem-no às suas necessidades, reutilizando a solução da parte comum dos problemas. De acordo com [Johnson, 1997], um *framework* pode ser definido como um projeto reutilizável de todo ou de parte de um sistema, fornecendo um conjunto de classes abstratas e a forma com que suas sub-classes interagem. Deste modo, um *framework* pode ser encarado como um esqueleto de uma aplicação que pode ser adaptado por um desenvolvedor. Um *framework* é adaptável a uma situação específica em pontos denominados *hot-spots,* apresentando implementações prototípicas para domínios de conhecimento.

O desenvolvimento dos componentes normalmente está associado a um *component framework*, que governa como os componentes serão criados e combinados entre si. Um *component framework* fornece uma base para a integração dinâmica de diferentes ferramentas (até de diferentes fabricantes), de acordo com as necessidades e preferências dos usuários e desenvolvedores. Cada ferramenta é encarada como um componente que pode realizar um conjunto de tarefas e é passível de ser acoplado em uma arquitetura que deve dar suporte ao acoplamento de novos componentes e à distribuição dinâmica dos mesmos. Além disto, os usuários devem ser auxiliados na localização e utilização do componente necessário para executar determinada tarefa.

OLE2, DCOM, OpenDOC, Enterprise JavaBeans e CORBA são algumas arquiteturas que visam, através da composição de componentes interoperáveis, tornar a aplicação mais flexível. Cabe ressaltar que mesmo utilizando uma arquitetura de componentes, a interface da aplicação pode ocultar dos usuários finais como o sistema foi composto. Com isto, a flexibilidade fica disponível apenas aos desenvolvedores, que

por sua vez liberam diferentes versões da aplicação alterando a configuração e o conjunto de componentes.

Um dos *component frameworks* mais utilizados é o OLE2 [Chapell, 1996], que serve para integrar em um documento de uma aplicação objetos gerados por outras. Contanto que tenha sido desenvolvida de acordo com as especificações do OLE2, uma aplicação pode servir tanto como contêiner quanto como componente. A forma de se estender a aplicação, incorporando novos componentes, é instalando no sistema operacional outras aplicações compatíveis com o OLE2.

No Microsoft Word, por exemplo, podemos inserir em um texto, objetos de outras aplicações através do OLE2. Na Figura 3.7, estão as aplicações que podem ser utilizadas para gerar estes objetos em uma determinada máquina (note que há aplicações desenvolvidas por outros fabricantes). O objeto a ser inserido pode conter outros objetos OLE2 em sua composição.

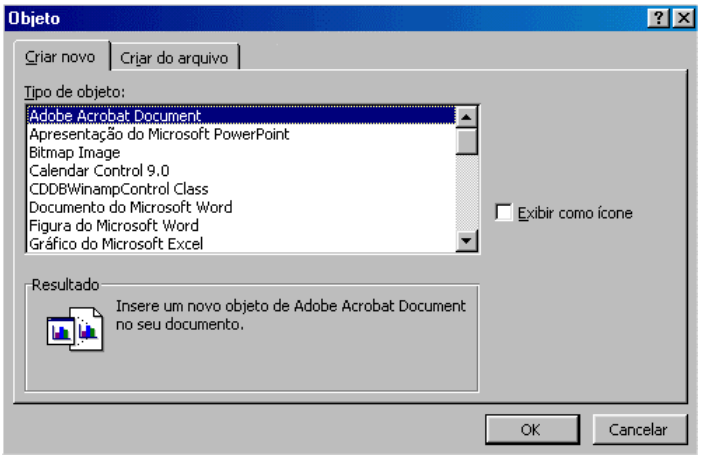

Figura 3.7. Inserção de um objeto OLE2 em um documento do Microsoft Word

Um problema da arquitetura OLE2, quando utilizada para o trabalho em grupo, é que se um membro receber um documento contendo um objeto desenvolvido em uma determinada aplicação, ele deve tê-la instalada em sua máquina para poder editar o objeto (apesar disto, o OLE2 possibilita que mesmo sem ter a aplicação de origem instalada no sistema, seja possível visualizar e imprimir o objeto). Como não há um lugar central de onde os componentes (aplicações) possam ser instanciados, um usuário pode introduzir um componente que não esteja disponível para os outros, como por exemplo um novo editor de fórmulas. Os outros usuários não poderão editar os objetos criados com este editor, mesmo que tenham um outro editor de fórmulas instalado.

O DCOM, que é uma extensão do COM (*Component Objetct Model*) da Microsoft, é uma arquitetura que provê uma interface cliente/servidor para que componentes possam se comunicar mesmo que estejam distribuídos em diversas máquinas e desenvolvidos em diferentes linguagens de programação. A interação é feita através de RPCs (*Remote Procedure Calls*).

De maneira similar, o CORBA (*Common Object Request Broker Architecture*) fornece uma infra-estrutura que possibilita a comunicação entre componentes distribuídos. O CORBA utiliza a IDL (*Interface Definition Language*) para descrever a interface pública de seus serviços. Além dos componentes poderem ser escritos em diferentes linguagens de programação, eles também podem estar executando em diferentes plataformas.

Recentemente a Microsoft lançou o .NET[14], um *framework* no qual aplicações Windows podem ser desenvolvidas e executadas. Assim como em Java, as aplicações .NET são pré-compiladas e executadas em uma máquina virtual, o que facilita a interoperabilidade em diferentes plataformas. A diferença é que o .NET não está restrito a uma única linguagem de programação, os programas podem ser escritos em qualquer linguagem compatível.

Na próxima seção, vamos analisar uma arquitetura que atende a maior parte dos requisitos de groupware descritos na terceira seção deste documento.

3.5.1. Arquitetura para Sistemas de Groupware Baseados em Componentes

Um tipo especial de componente a ser tratado neste trabalho é o componente de groupware. Componentes de groupware implementam ferramentas colaborativas com as quais os indivíduos operam nos objetos da cooperação. Estes componentes de groupware são acoplados em uma arquitetura derivada de um *component framework*. Esta arquitetura fornece a infra-estrutura através da qual os componentes irão se comunicar e acessar os objetos.

A arquitetura do groupware deve prover, portanto, mecanismos de controle de acesso e de concorrência aos objetos compartilhados, para evitar que um usuário sobreponha a ação de outro, pois um mesmo objeto compartilhado pode ser acessado simultaneamente por mais de um componente de groupware. É necessária uma forma de mapear os objetos e os componentes associados a eles.

Na área de trabalho do sistema operacional OS/2, a amarração entre um objeto e a aplicação utilizada para manipulá-lo é uma propriedade do objeto. Os objetos são criados a partir de *templates*, que incluem entre outras coisas as aplicações padrões para aquele tipo de objeto. No sistema operacional Windows 9x, o mapeamento é feito através de tarefas. O tipo de arquivo é detectado por sua extensão e, para cada tipo de arquivo, há uma série de operações possíveis e aplicações associadas a elas. Por exemplo, arquivos com a extensão HTML podem estar associados para abrir no Internet Explorer, para editar no FrontPage e para imprimir usando o Word.

Em um ambiente colaborativo, a abordagem de amarrar diretamente o objeto a uma aplicação traz algumas limitações, já que os membros do grupo normalmente efetuam tarefas diferentes sobre um mesmo objeto, dependendo do papel de cada um. Por exemplo, em um ambiente de aprendizagem, o docente precisa editar os conteúdos

---

[14] http://www.microsoft.com/net

didáticos, enquanto os aprendizes normalmente necessitam de um visualizador. Da mesma forma, podem existir diferentes aplicações para executar uma mesma tarefa em um objeto, como por exemplo, para lidar com mensagens de correio eletrônico podem estar disponíveis o Outlook e o Eudora.

Portanto, o que define qual o componente a ser utilizado será a tarefa que o usuário deseja executar sobre o objeto. Neste modelo de desenvolvimento, denominado desenvolvimento baseado em tarefas (*task-based development*), para cada objeto e tarefa há somente um componente associado. A amarração entre os componentes e objetos é feita através das tarefas. Por exemplo, a impressão de determinado objeto pode estar associada a um componente, enquanto a edição do mesmo, a outro, similarmente ao que ocorre no Microsoft Windows. A função de um componente de groupware é executar tarefas em objetos, sendo que um mesmo componente pode executar diferentes tarefas em diferentes tipos de objetos (classes).

As tarefas devem acompanhar as heranças e o polimorfismo das classes. Por exemplo, se um componente C1 executa uma tarefa T1 em uma classe A, e a classe B é uma subclasse desta, então C1 também executa a tarefa T1 em B. Da mesma forma, se C2 estende o componente C1 por herança, então C2 executa todas as tarefas de C1, mais as suas específicas. C2 também pode sobrescrever (*override*) tarefas herdadas.

O conjunto de tarefas que cada componente pode executar é solicitado pelo sistema durante a inicialização do servidor ou quando um novo componente entra no repositório. De posse da informação de quais componentes executam quais tarefas em quais classes de objetos, o sistema deve ajudar o usuário a selecionar o componente mais adequado à sua necessidade (RU2 – auxílio à escolha da ferramenta apropriada). As tarefas também servem para os componentes invocarem uns aos outros, reduzindo o acoplamento e a interdependência entre eles, já que não há chamada direta de um componente a outro. Isto possibilita a extensão e a reconfiguração dinâmica do sistema.

Conforme visto anteriormente, uma arquitetura de groupware baseado em componentes deve prover flexibilidade suficiente para que os componentes interajam entre si e para que os novos componentes possam ser incorporados e compostos. A arquitetura deve ter um repositório de objetos compartilhados, com mecanismos de acesso e gerenciamento dos mesmos.

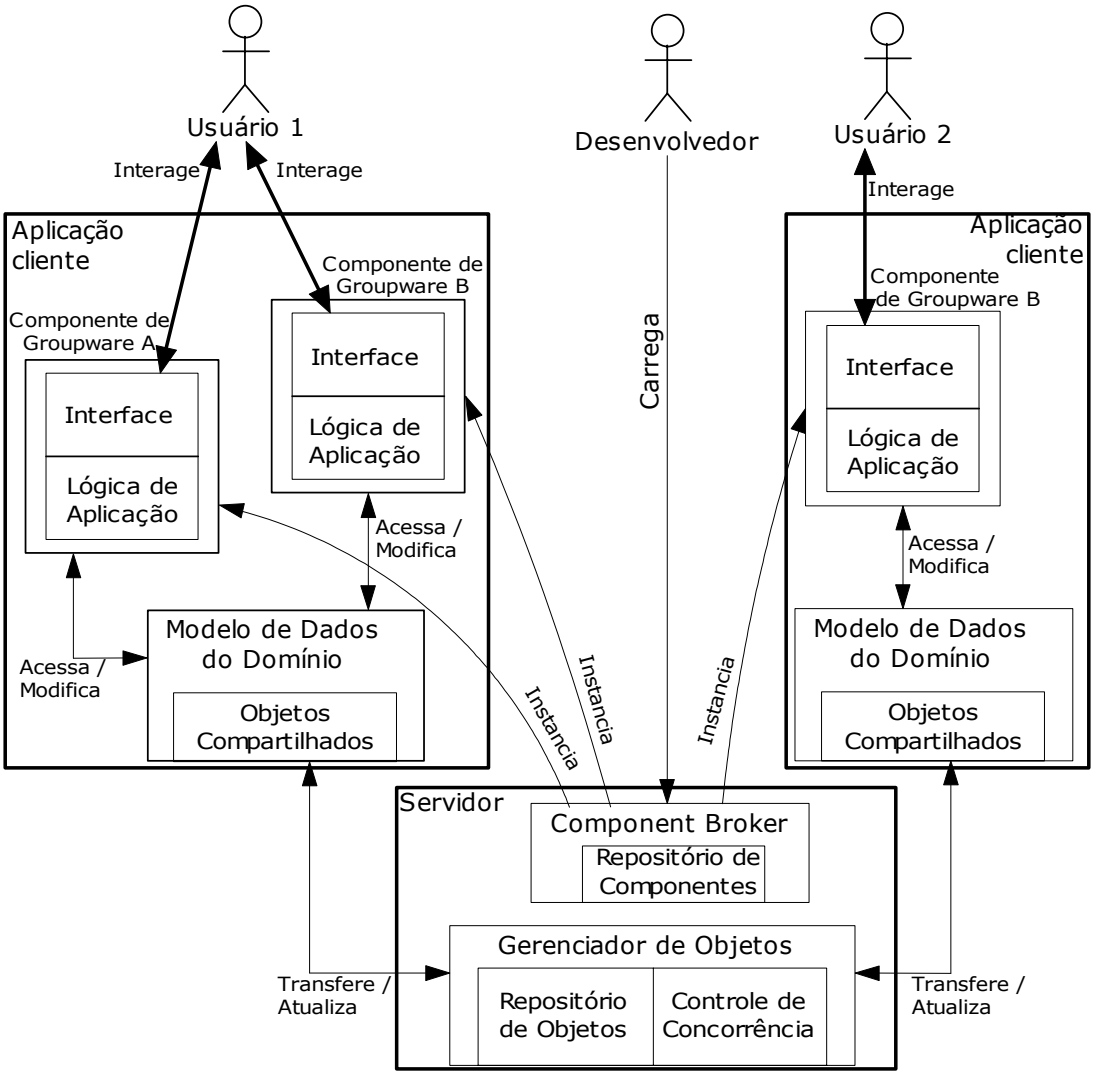

Figura 3.8. Arquitetura para um groupware baseado em componentes (proposta por [Tietze, 2001])

Na Figura 3.8, encontra-se o diagrama que representa uma arquitetura para um sistema baseado em componentes de groupware. Cada usuário utiliza uma aplicação cliente que contém os componentes de groupware. Estes componentes são instanciados a partir do repositório do *Component Broker*, localizado no servidor. Cada componente combina uma camada de interface, onde são implementadas as funcionalidades relativas à interação com o usuário, e uma camada de lógica de aplicação, onde se localizam as funcionalidades específicas do componente e de manipulação dos objetos.

A divisão da implementação do componente em interface e lógica da aplicação possibilita que ele tenha diferentes versões da camada de interface para, por exemplo, computadores pessoais, computadores de mão e celulares. Estas implementações da camada de interface utilizam a mesma camada de lógica de aplicação, e com isto, o comportamento do componente é mantido para as diferentes formas de acesso. A manutenção também é facilitada pois qualquer alteração no comportamento do

componente fica valendo para todas as formas de acesso. Esta característica favorece o RU9 (mobilidade) e RD2 (aproveitamento de modelo de dados).

Os componentes acessam os objetos através do Modelo de Dados do Domínio[15], que implementa as funcionalidades relativas ao domínio em questão e é responsável por manter as consistências semânticas entre os objetos. O Modelo de Dados faz uma cópia do objeto compartilhado localizado no Gerenciador de Objetos do servidor. A cada alteração realizada em um objeto compartilhado, o Modelo de Dados se encarrega de replicá-la no Gerenciador de Objetos, e este por sua vez, de replicá-la em outros clientes que estejam utilizando o objeto modificado.

Diversos componentes, inclusive de natureza diferente, podem compartilhar ao mesmo tempo os objetos de cooperação (RU8). A arquitetura em questão deve prover recursos para fazer a propagação de alterações e o controle de concorrência, livrando os desenvolvedores dos componentes desta tarefa (RD3). As alterações de um objeto feitas por um usuário são propagadas para todos os componentes que estejam utilizando aquele objeto, e esses se encarregam de refletir a alteração na interface com o usuário, de forma a manter a consistência de todas as instâncias dos objetos. Estas funcionalidades são implementadas na arquitetura mostrada na Figura 3.8 através do Modelo de Dados do Domínio, que se encarrega de gerenciar os objetos na máquina do cliente, e do Gerenciador de Objetos, que centraliza os objetos e mantém consistentes todos os modelos de dados. Este compartilhamento facilita a integração de diferentes componentes e reduz o tempo de projeto e implementação, visto que são aproveitados esforços anteriores (RD2).

A persistência dos objetos no Gerenciador de Objetos torna possíveis aplicações assíncronas (RU4). Já o Modelo de Dados do Domínio possibilita utilizar de forma similar objetos locais e compartilhados (RD4) e reduz o tráfego na rede, pois uma cópia do objeto compartilhado é armazenada localmente e apenas as modificações trafegam pela rede (RU11).

A utilização de componentes favorece a separação das ferramentas colaborativas dos objetos compartilhados, possibilitando fácil acesso (RU1), extensão dinâmica do ambiente (RU7) e agrupamento de ferramentas (RU10). Além disto, o carregamento sob demanda facilita a implementação dos requisitos de performance (RU11) e de escalabilidade (RD7).

A arquitetura deve também prover recursos para que os desenvolvedores possam carregar no sistema, sem reiniciá-lo, novos componentes ou novas versões de componentes existentes (RD6). Esta carga é feita no *Component Broker*, que é a partir de onde os clientes irão instanciar os componentes. O *Component Broker* se encarrega de manter persistentes e consistentes as versões utilizadas pelos clientes, além de gerenciar as instâncias que estão rodando no servidor (RD9). Ele também auxilia na seleção do componente adequado à execução de uma determinada tarefa em um objeto (RU2).

---

[15] Neste contexto, Modelo de Dados do Domínio é um elemento da aplicação cliente (veja Figura 3.8).

A utilização de uma arquitetura que gerencia o compartilhamento de objetos e a armazenagem e instanciação de componentes facilita o desenvolvedor, que por sua vez pode se concentrar no desenvolvimento do componente como se fosse uma aplicação mono-usuário utilizando objetos locais (RD1 e RD3). Um exemplo de implementação da arquitetura da Figura 3.8 é encontrado em [Tietze, 2001].

3.5.2. Revisitando o AulaNet usando a Abordagem de Engenharia de Groupware

A versão atual do AulaNet (2.0) foi desenvolvida levando-se em conta os aspectos de colaboração. O esforço agora é evoluir da arquitetura atual para uma baseada em componentes, de forma a atender um maior número de requisitos de groupware.

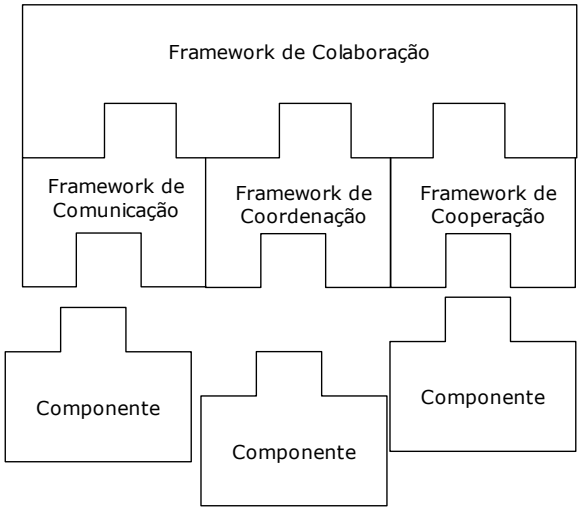

Figura 3.9. Framework de colaboração com os frameworks de comunicação, coordenação e cooperação

Na Figura 3.9 há um *framework* mais geral, denominado *Framework de Colaboração*, que implementará diversas características comuns a todos os serviços. Acoplados a ele teremos *frameworks* de comunicação, de coordenação e de cooperação. Ao se acoplar componentes ou novos *frameworks* a um destes *frameworks*, as aplicações já herdarão diversas funcionalidades relativas aos aspectos da colaboração e aos requisitos de groupware. Na figura, para efeito de ilustração, somente um componente está sendo acoplado a cada um dos *frameworks* intermediários.

Estes componentes poderão ser instanciações de *frameworks* para desenvolver funcionalidades específicas de um conjunto de ferramentas. Atualmente, os serviços de comunicação síncronos do AulaNet (Debate e Mensagem para Participantes) são instanciações do *framework* Canais de Comunicação [Ferraz, 2000]. Isto também será feito com os serviços de comunicação assíncronos (Contato com os Docentes, Conferências e Lista de Discussão), que lidam com mensagens e compartilham várias funcionalidades (abrir, remover, conceituar, alterar categoria, etc.), algumas vezes de forma diferente sobre os mesmos objetos (mensagens, categorias, eventos, etc.). Os serviços de coordenação e cooperação também passarão por reestruturações semelhantes.

Esta abordagem possibilitará um maior reuso de software, mais rapidez na integração de novos serviços ao ambiente e a possibilidade de equipes externas desenvolverem novas funcionalidades. A partir do *framework*, será possível instanciar groupware específicos para cada situação, como por exemplo, medicina, engenharia, projeto de software, e obviamente aprendizagem, bastando para tanto alterar a configuração dos componentes e a nomenclatura utilizada.

3.5.3. UML

Ao projetar e desenvolver software, é necessária uma notação comum entre os envolvidos para que eles possam documentar e debater sobre o projeto do sistema. Esta notação comum reduz a chance de má interpretação, que pode acarretar mudanças na fase de implementação, normalmente associadas a altos custos. Uma notação de projeto que se tornou padrão e está sendo largamente utilizada pela indústria é a UML (*Unified Modeling Language*). A UML é uma notação extensível que inclui definições de diagramas de modelagem para as diversas fases do projeto, desde as primeiras até as mais refinadas [Booch et al., 1998].

Para projetar componentes que sejam compatíveis com um *framework*, como o mostrado na Seção 3.5.1, pode-se valer da notação UML com algumas extensões específicas. Somente são apresentados neste documento os diagramas impactados por estas extensões. Com estas extensões, desde a fase de projeto fica claro com quais tipos de objetos se lidará, facilitando a posterior integração com a arquitetura.

Para propósitos de extensibilidade, a UML prevê o uso de *stereotypes*, que são pontos de extensão dos diagramas para serem utilizados com particularidades específicas do domínio em questão (neste caso, o domínio é desenvolvimento de groupware baseado em componentes). Estereótipos são representados nos diagramas através de um nome precedido por “<<” e sucedido por “>>”.

Uma primeira adaptação proposta nos diagramas de UML para representar aspectos específicos da engenharia de groupware é a distinção, no diagrama de classes, de quais são as classes que representam objetos compartilhados e de quais representam os componentes de groupware. Isto é feito indicando antes do nome da classe o estereótipo << shared >>, para classes que representam objetos compartilhados, e << component >> para componentes de groupware, conforme pode ser observado na Figura 3.10 [Tietze, 2001]. Ao indicar estas extensões, está implícito que as classes são derivadas do *framework*. Elas herdam os mecanismos de encaixe na arquitetura, concorrência, sincronização, etc.

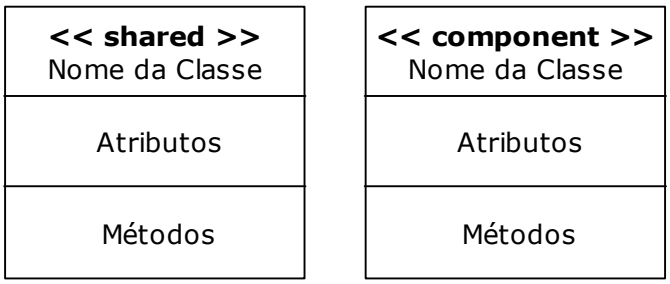

Figura 3.10. UML estendida mostrando classes que representam objetos compartilhados (à esquerda) e componentes de groupware (à direita)

Conforme visto anteriormente, a amarração entre os componentes e os objetos compartilhados é feita através de tarefas, o que gera a necessidade de representá-las no diagrama de classes. Isto é indicado através da extensão << task >> nas relações entre as classes. Observe no exemplo da Figura 3.11 dois componentes (editor de documento e visualizador de estrutura) que podem atuar sobre um mesmo objeto compartilhado (documento). As amarrações entre eles são feitas através de tarefas (editar e navegar). Note também que um objeto da classe Documento pode conter outros objetos da mesma classe (um documento composto de seções). Como o objeto Documento é um objeto compartilhado, faz-se necessário representar a extensão << slot >> antes dos nomes dos atributos[16]. Observe que os componentes possuem o método *giveTaskBindings*, que é o chamado pelo servidor durante a inicialização para saber quais tarefas o componente pode executar.

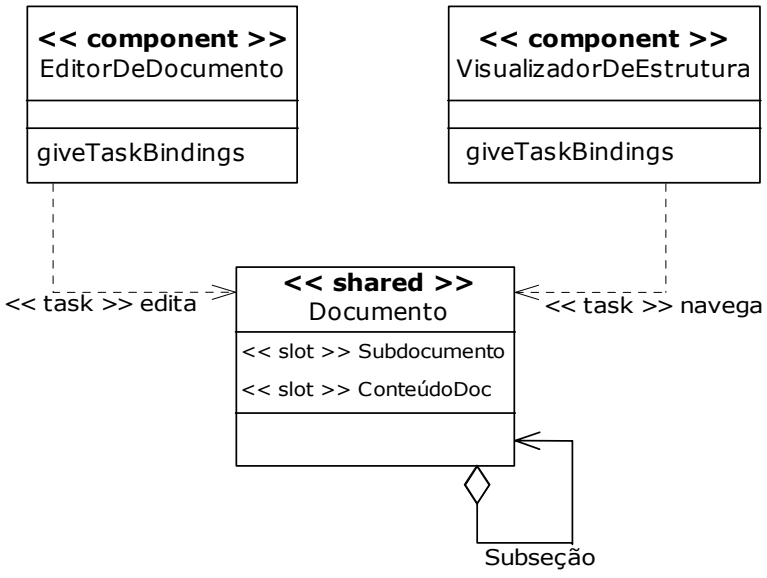

Figura 3.11. Diagrama de classes apresentando dois componentes de groupware que podem executar tarefas em um mesmo objeto compartilhado

[16] O <<slot>> faz a distinção entre os atributos regulares dos objetos e os elementos de dados replicados contidos em objetos compartilhados, que têm comportamentos diferentes e necessidades especiais de manutenção de consistência.

Para possibilitar as operações sobre um objeto compartilhado, o sistema em questão deve dar suporte ao conceito de sessão. Uma sessão comporta um grupo de usuários atuando conjuntamente (cooperando), e um mesmo usuário pode estar em várias sessões simultaneamente, já que ele pode estar rodando múltiplas instâncias da aplicação cliente em várias máquinas. Uma sessão pode ser individual ou coletiva, sendo que a primeira pode ser transformada na segunda, convidando usuários para a sessão, e vice-versa (RU6). Na Figura 3.12, aparece um diagrama UML que mostra duas sessões. Elas são representadas através de elementos da linguagem UML denominados componentes[17] marcados com a extensão << session >>.

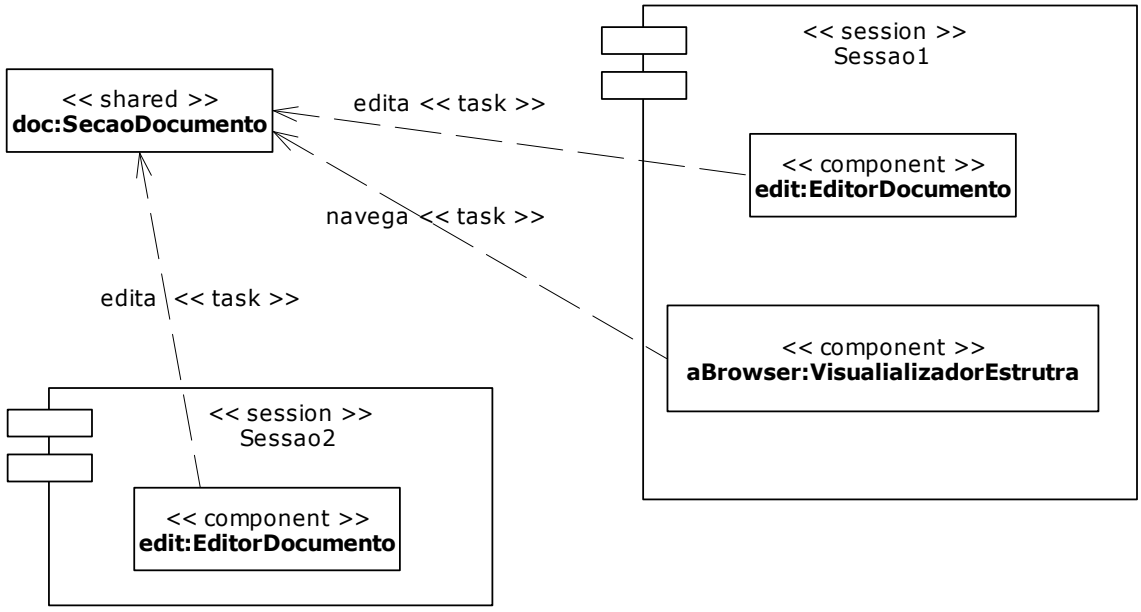

Figura 3.12. Extensão de UML para modelar sessões

Algumas operações devem ser realizadas no servidor (RD9), como por exemplo, a execução de uma instância de um fluxo de trabalho por uma máquina de *workflow*, que não está associada a nenhum cliente específico e deve continuar mesmo que ninguém esteja presente no sistema. Outros casos em que se fazem necessários componentes no servidor incluem monitoração de recursos, agendamento, processamento central, etc.

Um componente no servidor também executa um conjunto de tarefas sobre objetos compartilhados e provê uma interface com usuário, visto que algumas vezes é necessário intervenção deste. Como estruturalmente um componente de groupware localizado no servidor é equivalente a um componente de groupware localizado em um cliente, não há necessidade de extensões UML específicas para representá-lo.

[17] Componente na linguagem UML é utilizado para modelar partes físicas e substituíveis de um sistema que estão em conformidade e realizam uma série de interfaces. Normalmente são utilizados para representar executáveis, bibliotecas, tabelas, arquivos, documentos, entre outros.

Entretanto, sua sessão é mostrada em um elemento *Node*[18] que representa o servidor, conforme pode ser observado na Figura 3.13.

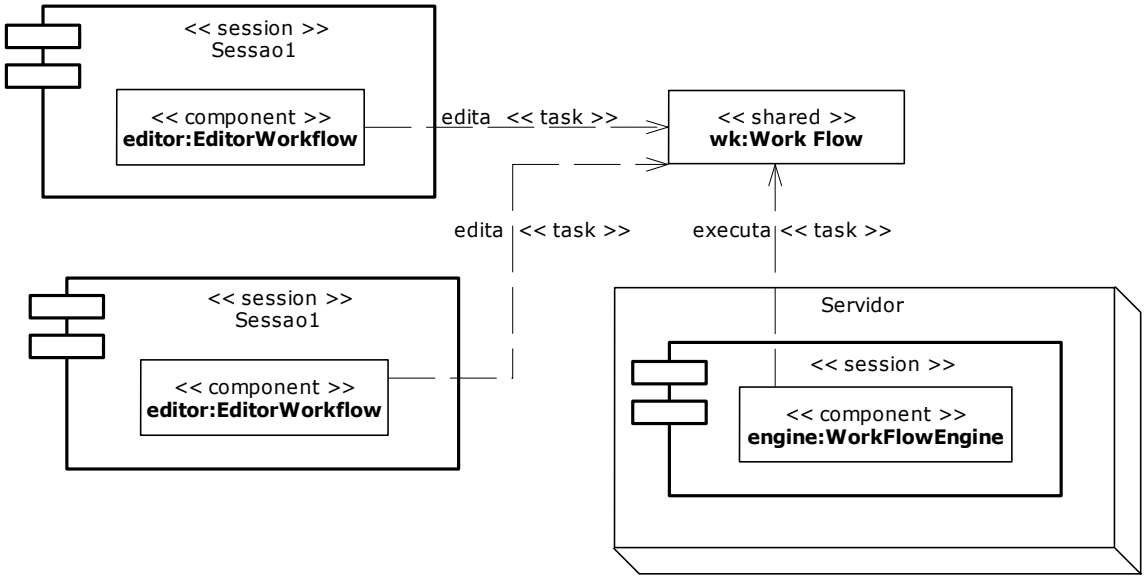

Figura 3.13. Exemplo de componente localizado no servidor

A utilização de componentes juntamente com as extensões dos diagramas de UML, favorece o aproveitamento de conhecimento e da experiência dos desenvolvedores com aplicações mono-usuários, já que boa parte dos mecanismos específicos que dão suporte às ferramentas colaborativas está pronta e é herdada para os componentes (RD1).

Nesta seção foram apresentadas arquiteturas e técnicas de UML para o desenvolvimento de groupware. Fica claro que não basta conhecimento de programação para se projetar um groupware, são necessários também conhecimentos sobre aspectos da colaboração em um grupo de trabalho e dos requisitos de usuários e de desenvolvedores.

3.6. Conclusão

Uma crescente parte do trabalho das empresas e instituições não é mais realizada individualmente, com uma pessoa trabalhando sozinha até completar as tarefas. O trabalho é cada vez mais realizado colaborativamente. Esta tendência se deve parcialmente ao aumento de complexidade das tarefas, que passam a requerer habilidades multidisciplinares, e aos novos paradigmas de trabalho, que envolvem diversos setores da empresa, ou até mesmo outras empresas, trabalhando conjuntamente nas diversas fases de elaboração de um produto ou desenvolvimento de um projeto.

---

[18] *Node* é um elemento da linguagem UML utilizado para representar um recurso computacional existente em tempo de execução e que normalmente conta com memória e capacidade de processamento. Um conjunto de componentes pode residir em um n*ode* e eventualmente migrar de um *node* a outro.

O advento das organizações virtuais, das empresas geograficamente dispersas e das parcerias entre empresas aumenta a demanda de trabalho colaborativo distribuído. A área de pesquisa CSCW visa estudar o trabalho colaborativo através de sistemas de computação, denominados groupware, que apóiem a comunicação, coordenação e cooperação entre os membros da equipe envolvida no trabalho, mesmo que estes estejam distribuídos no tempo e no espaço.

De acordo com [Morch, 1997], um sistema pode prover três níveis de modificação: customização, integração e extensão. Na customização o usuário pode modificar um conjunto de opções pré-definidas. Na integração, os usuários podem ligar componentes pré-definidos de forma a construir e modificar as ferramentas do sistema. E a extensão, fornece subsídio a modificação do sistema através de inserção de trechos código de uma linguagem de programação, como um script. Normalmente, sistemas desenvolvidos baseados em componentes favorecem o segundo tipo de modificação, isto é, a integração. Através do uso de componentes o sistema pode evoluir no tempo à medida que os processos de trabalho evoluem.

Neste trabalho, propomos uma abordagem de engenharia para o desenvolvimento de groupware, adaptando-se técnicas e metodologias da Engenharia de Software. Para embasar a primeira fase do ciclo de desenvolvimento, a análise do domínio, foi proposto um modelo de colaboração baseado nos aspectos de comunicação, coordenação e cooperação. Para a fase de elicitação de requisitos, foram listados requisitos de groupware, dos pontos de vista de seus usuários e de seus desenvolvedores. Foi feito então um mapeamento destes requisitos e dos aspectos da colaboração em ferramentas colaborativas: videoconferências, ambientes virtuais colaborativos e *learningware*.

A seguir, foram apresentadas algumas arquiteturas que dão suporte à composição de componentes interoperáveis que podem ser utilizadas para o desenvolvimento de ferramentas colaborativas. Atenção especial foi dada à arquitetura proposta por [Tietze, 2001], buscando mapear seus elementos aos requisitos de groupware. Finalmente, foram descritas extensões UML para a modelagem de groupware.

Conforme foi dito na introdução deste trabalho, este documento não se propõe a ser um livro-texto sobre engenharia de groupware. Infelizmente, como não há tal livro, ficamos impedidos de referenciá-lo. Com relação a groupware e CSCW, os interessados têm à sua disposição uma série de livros. Para citar somente alguns temos:

- “Computer Supported Cooperative Work - A book of readings” [Greif, 1988] que traz uma coleção de artigos seminais da área;

- “Computer-Supported Cooperative Work: Introduction to Distributed Applications” [Borghoff and Schlichter, 2000] que aborda o assunto sob a ótica das aplicações distribuídas;

- "Computer-Supported Cooperative Work: Issues and Implications for Workers, Organizations, and Human Resource Management" [Coovert and Thompson 2001] que traz um enfoque gerencial sobre o assunto; e

- "No more teams! Mastering the dynamics of creative collaboration" [Schrage, 1995] que oferece um guia do usuário que deseja trabalhar de forma colaborativa.

Finalmente, aos interessados em se aprofundar no trabalho dos autores, há uma série de artigos disponíveis no nosso site Groupware@LES[19]. Também é oferecida a disciplina Tecnologias de Informação Aplicadas à Educação[20] no Departamento de Informática da PUC-Rio. O objetivo desta disciplina é preparar educadores para a utilização de tecnologia de informação para aprendizagem em grupo na Web. Esta disciplina é ministrada totalmente via Internet pelo ambiente AulaNet e pode ser cursada por alunos de outras instituições, bastando para tanto matricular-se como aluno extraordinário. Toda esta experiência está relatada em [Lucena and Fuks, 2000].

3.7. Agradecimentos

Agradecemos aos professores Carlos José Pereira de Lucena, coordenador do Laboratório de Engenharia de Software (LES) e Marcelo Gattass, coordenador do Laboratório de Tecnologia em Computação Gráfica (Tecgraf) pelo apoio ao desenvolvimento deste trabalho.

Hugo Fuks e Marco Aurélio Gerosa têm bolsas do CNPq (respectivamente nº 524557/96-9 e nº 140103/02-3). O Tecgraf é prioritariamente financiado pela Petrobras.

Nossos agradecimentos aos alunos da primeira edição da disciplina de Engenharia de Groupware ministrada no primeiro semestre de 2002 na pós-graduação do Departamento de Informática da PUC-Rio: Anarosa Alves Franco Brandão, Ismael Humberto F. dos Santos, Mariano Gomes Pimentel, Juliana Lucas de Rezende e Luís Henrique Raja Gabaglia Mitchell, pela leitura e análise de uma versão preliminar deste texto. Finalmente, nossos agradecimentos a Viviane Torres da Silva pelas suas críticas e comentários sobre *framework* e UML.

3.8. Bibliografia

Bannon, L.J. and Schmidt, K. (1991) "CSCW: Four Characters in Search of a Context", In: Computer Supported Cooperative Work, Edited by Bowers, J. M. and Benford, North-Holland: Elsevier Science Publishers B. V., Holland.

Benbunan-Fich, R. and Hiltz, S.R. (1999) "Impacts of Asynchronous Learning Networks on Individual and Group Problem Solving: A Field Experiment", Group Decision and Negotiation, Vol.8, p. 409-426.

---

[19] http://www.les.inf.puc-rio.br/groupware

[20] http://www.les.inf.puc-rio.br/aulanet

Benford, S., Bowers, J., Fahlen, L., Mariani, J. and Rodden, T. (1994) "Supporting Cooperative Work in Virtual Environments", The Computer Journal, Vol. 37, N. 8, p. 653-668.

Berry, D.M. (1992) "Academic Legitimacy of the Software Engineering Discipline", Technical Report CMU/SEI-92-TR-34/ESC-TR-92-034, Software Engineering Institute, Carnegie Mellon University, USA.

Bischofberger, W.R., Kofler, T., Mätzel, K. and Schäffer, B. (1994) "Computer-supported cooperative software engineering with Beyond-Sniff", UNILAB Techical Report 94.9.1, Union Bank of Switzerland, Switzerland.

Booch, G., Rumbaugh, J. and Jacobson, I., The Unified Modeling Language User Guide, USA, Addison-Wesley, 1998. ISBN 0-201-57168-4

Borghoff, U.M. and Schlichter, J.H., Computer-Supported Cooperative Work: Introduction to Distributed Applications. Springer, USA, 2000. ISBN 3-540-66984-1

Brinck, T. and McDaniel, S.E. (1997) "Awareness in Colaborative Systems", Workshop Report, SIGCHI Bulletin.

Chappell, D. Understanding ActiveX and OLE, USA, Microsoft Press, 1996. ISBN 1-57231-216-5

Coovert, D.M. and Thompson, L.F., Computer-Supported Cooperative Work: Issues and Implications for Workers, Organizations, and Human Resource Management. Sage Publications, USA, 2001. ISBN 0-7619-05723-1

Daft, R.L. and Lengel, R.H. (1986) "Organizational information requirements, media richness and structural design", Organizational Science, 32/5: 554-571

Delvin K. and Rosemberg, D. (1996) "Language at Work: analyzing communication breakdown to inform system design", CSLI lecture notes no. 66.

Dourish, P. and Belloti, V. (1992), "Awareness and coordination in shared workspaces", Proceedings of Computer Supported Cooperative Work 1992, Toronto, Ontario, ACM Press, USA, p. 107-114.

Ellis, C.A., Gibbs, S.J. and Rein, G.L. (1991) "Groupware - Some Issues and Experiences", Communications of the ACM, January 1991, Vol. 34, N. 1, p. 38-58.

Ferraz, F. G., Framework Canais de Comunicação. Trabalho Final de Curso de Engenharia de Computação, Departamento de Informática, PUC-Rio, 2000.

Fuks, H. and Assis, R.L. (2001) "Facilitating Perception on Virtual Learningware-based Environments", The Journal of Systems and Information Technology, Vol 5., No. 1, Edith Cowan University, Australia, p. 93-113. ISSN 1328-7265

Fuks, H., Gerosa, M.A. and Lucena, C.J.P. (2002) “The Development and Application of Distance Learning on the Internet”, The Journal of Open and Distance Learning, Carfax Publishing, UK, February 2002, Vol. 17, N. 1, p. 23-38. ISSN 0268-0513

Fussell, S.R., Kraut, R. E., Learch, F.J., Scherlis, W.L., McNally, M.M. and Cadiz, J.J. (1998) “Coordination, overload and team performance: effects of team communication strategies”, Proceedings of CSCW '98, Seattle, USA, p. 275-284. ISBN 1-58113-009-0

Gerosa, M.A., Fuks, H. and Lucena, C.J.P. (2001) “Use of Categorization and Structuring of Messages in Order to Organize the Discussion and Reduce Information Overload in Asynchronous Textual Communication Tools”, Proceedings of the 7th International Workshop on Groupware - CRIWG, Darmstadt, Germany, IEEE Computer Society, USA, p. 136-141. ISBN 0-7695-1351-4

Gerosa, M.A., Fuks, H. & Lucena, C.J.P. (2001a) “Elementos de percepção como forma de facilitar a colaboração em cursos via Internet”, Anais do XII Simpósio Brasileiro de Informática na Educação, Vitória-ES, p. 194-202.

Greif, I. (Ed). Computer Supported Cooperative Work - A book of readings. Morgan Kaufmann Publishers, USA, 1988. ISBN 0-934613-57-5.

Gutwin, C. and Greenberg, S. (1999), “A framework of awareness for small groups in shared-workspace groupware”, Technical Report 99-1, Saskatchewan University, Canada.

Hagsand, O. (1996) “Interactive Multiuser VEs in the DIVE System”, IEEE Multimedia; Vol. 3, N. 1, p. 30-39.

Harasim, L., Hiltz, S.R., Teles, L. and Turoff, M. Learning networks: A field guide to teaching and online learning. MIT Press, USA, 1997. ISBN 0-262-08236-5

IEEE Standard Glossary of Software Engineering Technology, USA, 1991.

Johnson, R.E. (1997) “Frameworks = Components + Patterns”, Communications of the ACM, USA, Vol. 40, N. 10, p. 39-42.

Joslin, C., Molet, T., Magnenat-Thalmann, N., Esmerado, J., Thalmann, D., Palmer, I., Chilton, N. and Earnshaw, R. (2001) “Sharing Attractions on the Net with VPark”, IEEE Computer Graphics and Applications; Vol. 21, N. 1, p. 61-71.

Kraut, R.E. & Attewell, P. (1997), “Media use in global corporation: eletronic mail and organizational knowledge”, in Research milestone on the information highway, Mahwah, NJ: Erlbaum, USA.

Long, B. and Baecker, R. (1997) “A taxonomy of Internet communication tools”, Proceedings of WebNet - World Conference of the WWW, Internet, and Intranet, Toronto, Canada, p. 318-323. ISBN 1-880094-27-4

Lucena, C.J.P. and Fuks, H. (2000) Professores e Aprendizes na Web: A Educação na Era da Internet, Editora Clube do Futuro, Rio de Janeiro, Outubro 2000. ISBN 85-88011-01-8

Malone, T.W. and Crowston, K. (1990) "What Is Coordination Theory and How Can It Help Design Cooperative Work Systems?", Proceedings of the Conference on Computer-Supported Cooperative Work, Los Angeles, USA, p. 357-370. ISBN 0-89791-402-3

Morch, A.I. (1997) "Three Levels of End-user Tailoring: Customization, Integration, and Extension" In: Computers and Design in Context, Edited by M. Kyng and L. Mathiassen, MIT Press, USA.

Ostwald, J. (1995), "Supporting collaborative design with representations for mutual understanding", Proceedings of the Conference on Computer Human Interface, Doctoral Consortium.

Pressman, R. Software Engineering: A Practitioner's Approach, 3rd ed. McGraw-Hill, USA, 1992. ISBN 0-07-050814-3.

Putnam, L.L. and Poole, M.S. (1987) "Conflict and Negotiation", Handbook of Organizational Communication: An Interdisciplinary Perspective, Newbury Park, p. 549-599.

Raposo, A.B., Magalhães, L.P., Ricarte, I.L.M. and Fuks, H. (2001) "Coordination of collaborative activities: A framework for the definition of tasks interdependencies", Proceedings of the 7th International Workshop on Groupware - CRIWG, Darmstadt, Germany, IEEE Computer Society, USA, p 170-179. ISBN 0-7695-1351-4

Raposo, A.B., Cruz, A.J.A., Adriano, C.M. and Magalhães, L.P. (2001a) "Coordination Components for Collaborative Virtual Environments", Computers & Graphics, Pergamon Press, Holland, Vol. 25, N. 6, p. 1025-1039.

Rhodes, J. (2001) Videoconferencing for the Real World: Implementing Effective Visual Communications Systems, Focal Press, USA. ISBN 0240-80416-3.

Salomon, G. and Globerson, T. (1989) "When Teams do not Function the Way They Ought to", Journal of Educational Research, USA, 13, (1), p. 89-100.

Schmidt, K. (1991) "Riding a Tiger, or Computer Supported Cooperative Work", Proc. of the 2nd European Conf. on Computer-Supported Cooperative Work (ECSCW), p. 1-16.

Schön, D. The Reflective Practitioner: How Professionals Think in Action. Basic Books, USA, 1983. ISBN 0-465-06878-2

Schön, D. and Bennet, J. (1996) "Reflective Conversation with Materials", In: Bringing Design to Software, Edited by Terry Winograd, ACM Press, USA. ISBN 0-201-85491-0

Schrage, M. No more teams! Mastering the dynamics of creative collaboration. Currency Doubleday, USA, 1995. ISBN 0-385-47603-5

Schrage, M. (1996) “Cultures of Prototyping”, In: Bringing Design to Software, Edited by Terry Winograd, ACM Press, USA. ISBN 0-201-85491-0

Shum, S.B. and Hammond, N. (1994) “Argumentation-based design rationale: what use at what cost?”, Human-Computer Studies, USA, 40, p. 603-652.

Singhal, S. and Zyda, M. (1999) Networked Virtual Environments: Design and Implementation, Addison-Wesley, USA, 1999. ISBN 0-201-32557-8

Szyperski, C., Component Software: Beyond Object-Oriented Programming. ACM Press/Addison-Wesley, USA, 1999. ISBN 0-201-17888-5

Tietze, D.A., A Framework for Developing Component-based Co-operative Applications. Ph. D. Dissertation, Computer Science, Technischen Universität Darmstadt, Germany, 2001.

Turoff, M. and Hiltz, S.R. (1982) “Computer Support for Group versus Individual Decisions”, IEEE Transactions on Communications, USA, 30, (1), p. 82-91.

Winograd, T. and Flores, F., Understanding Computers and Cognition. Addison-Wesley, USA, 1987. ISBN 0-201-11297-3

Winograd, T., (1988) “A Language/Action Perspective on the Design of Cooperative Work”, IN: Computer Supported Cooperative Work - A book of readings, Edited by Irene Greif, Morgan Kaufmann Publishers, USA. ISBN 0-934613-57-5.